COMBINING ENTERPRISE ARCHITECTURE AND OPERATIONAL DATA TO BETTER SUPPORT DECISION-MAKING

Master Thesis by R.K.M. Veneberg

UNIVERSITY OF TWENTE.

BiZZdesign

Combining Enterprise Architecture & Operational Data to better support Decision-Making

Master Thesis by R.K.M. Veneberg

Author

Name: Roelof Klaas Marcel Veneberg
Programme: MSc. Business Information Technology
Institute: University of Twente
School of Management and Governance
Enschede, The Netherlands

Student number: s0173126
E-mail: r.k.m.veneberg@alumnus.utwente.nl

Graduation Committee

University of Twente: dr. Maria-Eugenia Iacob (First Supervisor)
University of Twente: dr. ir. Marten van Sinderen (Second Supervisor)
BiZZdesign: dr. Lianne Bodenstaff (Supervisor Company)

Date: June 6th, 2014

ACKNOWLEDGEMENTS

Over the past period writing this thesis I have dived into subjects I had not been working with thoroughly before. The subjects enterprise architecture and business intelligence were interesting and gave me the motivation to put the utmost energy in writing this thesis, to give something back to the research field and the people I worked for during the past period. Though sometimes complex and time-consuming, I have tried to explain my thoughts on the subject.

Of course I would have been nowhere without the help of my supervisors, dr. Maria-Eugenia Iacob who has been a creative source of inspiration and motivation that helped me creating ideas and writing them down on paper, dr. ir. Marten van Sinderen who made sure my formulations were accurate and tested my ideas thoroughly while motivating and handing useful information to proceed writing the thesis, and dr. Lianne Bodenstaff who was my daily source of strength during the week at the office, giving me ideas and questioning my proposed work to the level I deliver in this document. I am grateful to have worked with these people who also worked very hard on writing a paper to the EDOC 2014 conference in Ulm, Germany, where it was accepted as a full research paper.

During times outside the office, writing the thesis paid a toll to my friends and family who have had to cope with me when times were getting rough, when I was running into a deadlock writing my thesis. They gave me the motivation to go on and finish it, throughout my whole life. I would like to specially thank my girlfriend Evelien who understood the importance of finishing things up, sacrificing time with me to let me type the thesis.

I am thankful for the fellow interns at BiZZdesign, who were my daily pleasure at the office. I remember the funny and serious discussions we had, the struggles we shared all writing the thesis and going through the same phases, and most of all the weekly Tuesday sandwiches we ate. I am sure we will keep in touch after finishing my thesis. I am grateful to BiZZdesign who provided me a place to do my graduation on an interesting topic. I hope I have been a good colleague and a good working force, delivering a useful work on a topic relevant for the company and its clients. A special thanks to Henry Franken for handing me the option to do an exam in ArchiMate 2.0, which I luckily passed, Bas van Gils for his talks on data, and Boudewijn Meyboom for his support in learning how to script.

Most of all, I am thankful to God, for Whom I live, Whom I love. The strength and guidance I was given has been the most critical source for me to be able to write a thesis. Humbly, I try to share the love and message that is my inspiration for life, which I wish everyone to have.

> Οὕτως γὰρ ἠγάπησεν ὁ Θεὸς τὸν κόσμον, ὥστε τὸν Υἱὸν τὸν μονογενῆ ἔδωκεν, ἵνα πᾶς ὁ πιστεύων εἰς Αὐτὸν μὴ ἀπόληται ἀλλ᾽ ἔχῃ ζωὴν αἰώνιον.
>
> For God so loved the world that He gave his one and only Son, that whoever believes in Him shall not perish but have everlasting life.
>
> John 3, verse 16. The Bible.

I hope you will enjoy reading this thesis. If you have any questions, please do not hesitate to get into contact with me.

Yours truly,

Roel

ABSTRACT

Decision-making may be complex in large organizations dealing with many stakes and situations. Currently, decision-making is often done using business intelligence solutions, combined with data warehousing technologies storing operational data. Enterprise architecture is often used for strategy purposes and provides an overview of complex organizational architectures, showing business entities and relations.

> Data warehouses (DWs) storing ***operational data*** are used as a source for business intelligence to provide decision support (Sun & Heller, 2012), but they lack data traceability regarding enterprise processes and entities: business people need an explanation of the data to trace it back to their organization (Pettey & Van der Meulen, 2012). ***Enterprise architecture*** provides this type of traceability using familiar organizational entities and relating them in a structure (Zachman, 1987), but is not suitable as a source for business intelligence applications, since it does not store operational data (Johnson, Ekstedt, Silva, & Plazaola, 2004). ***Combining*** these two techniques hands possibilities for improving decision support to directly hand business people the data they need as evidence for decision-making, without having to trace it back to the location the data is showing something about: enterprise architecture may provide the meta data for operational data needed for business people to make decisions for their organization.

Currently the field of enterprise architecture and business intelligence lacks methodology in combining both worlds. This thesis provides a structured approach to address this gap. Going through six phases of the Enterprise Architecture Intelligence Lifecycle (EAIL) gives any organization dealing with complexity, having an enterprise architecture and operational data in-house, the chance to develop a better source of data for decision-making. The EAIL provides two ways of combining enterprise architecture and operational data in different approaches, namely by adding operational data to an enterprise architecture resulting in an 'enriched enterprise architecture' or by adding enterprise architecture meta data to an operational data source resulting in an 'enriched operational data source'.

Moreover, the thesis provides a model to store enterprise architecture, operational data and time, also known as the Concept Match Library (CML). Using the CML hands possibilities for forecasting and other analysis types, while maintaining the original data sources and adding the 'best of both worlds' into a single data source: meta data from enterprise architecture providing a context to operational data from data sources. These new technologies hope to bring more accurate and better structured data provision for decision-makers.

The EAIL is demonstrated using the Timber case: a real life case problem based on a case that was performed in the Netherlands in the pensions sector. Using a cost perspective, the EAIL was walked through all of its phases, showing the differences of normal business intelligence solutions with our results: a more concise and more accurate view on data and meta data, which was carefully selected using the EAIL.

Our work was rewritten in a paper and sent to the EDOC 2014 conference, where it was accepted as a full research paper. The EAIL and CML are presented in September, 2014 in Ulm, Germany.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

BI	Business Intelligence
CML	Concept Match Library
DB	Database
DS	Data Source
DSR	Design Science Research
DSRM	Design Science Research Methodology
DW	Data Warehouse
EA	Enterprise Architecture
EAIL	Enterprise Architecture Intelligence Lifecycle
IT	Information Technology
MO	Main Objective
OD	Operational Data
ODS	Operational Data Source
RQ	Research Question
SO	Sub Objective
UoD	Universe of Discourse
XML	Extensible Markup Language

I. INTRODUCTION

The area of Information technology (IT) these days is not only part of specific organizations in the technical sector, but has spread out over the whole spectrum of organizations in all kinds of sectors. Computers and the Internet have changed the world permanently and are now part of the 'corporate DNA'. Due to this change, organizations need to learn about IT in order to understand what the consequences of integrating IT solutions are for their business and daily work. The larger organizations become, the more complex their IT-infrastructure often becomes, due to the adding of IT hardware or software components. To stay in control of all this, structures are needed to optimize the functionality of all components that collaborate. Enterprise architecture (EA) gives organizations the possibility to create a structured overview of IT on a business level. Enriching EAs with data, like financial costs of IT, gives organizations new insights regarding their IT by relating it to certain parts of their organization. For management, information derived from EA is useful due to its relatively simple and understandable representation of rather complex IT architectures.

In time, both organizations and IT develop, creating situations where organizations inevitably need to decide what to do with their corporate IT. Decisions about investing in IT could for example mean that structures are optimized, software is improved and hardware is replaced, ideally creating a better performing business. All in all, the concept of combining enterprise architecture with operational data gives organizations options to enhance their decision support.

1.1 Problem statement

Having to decide about future investments regarding IT, management of organizations deal with complex situations that could influence their future. Due to the costly and time-consuming character of developing IT-related structures, like software programs or IT-infrastructures, making wrong decisions might have severe consequences for these organizations. Supporting management with information derived from cost analyses based on enterprise architecture (EA) structures might be a valuable asset to have when this means the correct decisions are to be taken. It is therefore interesting to know what kinds of information affect decisions made about IT, since these decisions might have an influence on the business.

Many decisions are based on costs and benefits, due to their measurable character and importance to the business. In the end, most businesses need to make a profit to survive and costly investments in IT therefore need to be done based on available information. The available information organizations have for decision support is often limited or difficult to directly link to IT; information they do have are direct figures about the costs of IT and possible benefits. Calculating the value of IT, however, still remains a difficult topic. Enriching enterprise architecture with cost-analysis information might be a solution for the management of organizations in a way that they can relate parts of the EA to costs in order to make financial predictions used for IT decision-making. However, providing organizations with quantifiable information regarding is currently often performed at their specific request by third party suppliers using non-formalized paths; ways of trying to find relevant information in order to do analyses and gain insights for decision support that have without any structure.

> Data warehouses (DWs) storing ***operational data*** are used as a source for business intelligence to provide decision support (Sun & Heller, 2012), but they lack data traceability regarding enterprise processes and entities: business people need an explanation of the data to trace it back to their organization (Pettey & Van der Meulen, 2012). ***Enterprise architecture*** provides this type of traceability using familiar organizational entities and relating them in a structure

> (Zachman, 1987), but is not suitable as a source for business intelligence applications, since it does not store operational data (Johnson, Ekstedt, Silva, & Plazaola, 2004). ***Combining*** these two techniques hands possibilities for improving decision support to directly hand business people the data they need as evidence for decision-making, without having to trace it back to the location the data is showing something about: enterprise architecture may provide the meta data for operational data needed for business people to make decisions for their organization.

The lack of a general method on the *requirements* to provide quantifiable information for enterprise managers using enterprise architecture means a void in the *approach* some third party suppliers have when they try to deliver useful decision support information. In short, no general methodology exists for those who try to combine enterprise architecture and operational data to support decision-making. This conforms to a *practical problem*, as mentioned by Wieringa (2009).

In the past decade, organizations are starting to create enterprise architectures with different stages of EA-maturity between organizations; some have just started developing, others have years of experience regarding their enterprise architecture. Our research is dependent of organizations having enterprise architecture. We will therefore take the property of having any form of enterprise architecture in an organization as a premise for our research. In order to provide a structured approach for quantifying the value of enterprise architectures based on costs, we will follow a structured approach explained in the next chapter. To start off with this approach, our *main problem* is formulated as follows:

> **How to combine operational data and enterprise architecture to better support decision-making?**

According to Wieringa (2009), this problem asks for the development of an *artefact*; the design of a solution to our problem. In Chapter 1.2, we start with explaining the objectives we try to achieve, which are the reasons for performing the research. Before we start developing the artefact, we will define the problem-related research questions, the properties of the subject that is investigated and the available resources (Wieringa & Heerkens, 2008). Chapter 1.3 discusses the research questions and motivates them. Chapter II discusses the research methodology that was used for structuring the research. Chapter 2.7 gives an overview and reading guide of our research with references to the research methodology and, lastly, Chapter 2.8 discusses the practical and theoretical relevance of the theory we will develop.

1.2 Objectives

The main objective for our research is to find a way to improve decision support combining enterprise architecture and operational data. In order to illustrate our concept, our approach is based on costs, due to the availability of cost-based information in enterprises. By conducting a literature study, we try to find a way to solve this problem for BiZZdesign, an enterprise architecture and business process management solutions provider, our main stakeholder, headquartered in The Netherlands. Wieringa (2010) explains that a *"practical problem is a problem to improve the world with respect to some stakeholder goals"*. The following 'stakeholder goals', as mentioned by Wieringa, were identified for our research. The main objective (MO) was first identified, directly reflecting our main problem in a goal. Next to the main objective, four sub objectives (SO) were identified that serve as focus areas and support the main objective.

The following main objective was identified from our main problem.

MO: To provide a decision support approach that combines enterprise architecture and operational data

To illustrate our approach, we need a form of enterprise data as an *input* to enterprise architecture to be able to eventually do a cost analysis. Not all enterprise input might be available or suitable for cost analysis. Therefore, we will try to come up with a description of what is suitable enterprise input. In order to describe the suitability, we will use 'costs' as an illustration

SO1: Description of suitable input for enterprise architecture enrichment

In order to do a cost analysis on enterprise architecture, we need to find a way to 'enrich' enterprise architectures with enterprise data, in our case enterprise input that contains cost-related information. By doing research regarding the possibilities, we try to come up with an approach that helps us in enriching enterprise architecture with cost-related input. We will validate our approach by applying case studies at companies.

SO2: To define an enterprise architecture enrichment approach

In order to perform an analysis on cost-enriched enterprise architectures, we need to come up with an overview of different cost analysis types to be able to distinguish which ones are suitable for performing analysis in order to create valuable analysis information for the enterprise. This analysis information can be used as input for business intelligence software applications.

SO3: Suitable cost analysis types for enterprise architecture enrichment

The data that was created based on cost analysis could be valuable for business intelligence applications that graphically represent input information. However, the business intelligence applications need to be set in order to know what to do with the input. Moreover, the input needs to be selected based on the criteria that it should be valuable for enterprise management and suitable to be represented by means of business intelligence software applications.

SO4: Recommendation on cost analysis output for business intelligence

Now we have several aiming points for our research, our next step is to translate these objectives into research questions and to pose sub research questions that might arise with the related research question.

1.3 Research Questions

Wieringa (2010) explains that *"research question investigation serves goals too"* and that *"as any other activity, it needs a budget"*. The objectives (or 'goals' as Wieringa states) were translated into the research questions stated below. Surely, more questions might be raised regarding the objectives we want to achieve. However, referring to Wieringa, our 'budget' is the *limited time span* this research was conducted in. Despite the limited time frame to perform this research, we believe the most relevant research questions were posed.

The first research question aims to find out which enterprise data is suitable for cost analysis as a starting point.

RQ1: What enterprise data can be used as input for EA-enrichment?

RQ1.1: What is the relation between enterprise data and costs?
RQ1.2: How to ensure data integrity for cost-related enterprise data?

The second research question aims to find out how to use the suitable enterprise data (found in RQ1) for EA-enrichment and how to perform EA-enrichment. It goes deeper into what the possibilities are regarding the information insertion into enterprise architecture and what the best ways are with a cost perspective.

RQ2: How to enrich EA with cost-related enterprise data?

RQ2.1: What are the premises for EA to be enriched with enterprise data?
RQ2.2: What techniques exist to enrich EA?
RQ2.3: What is the influence of EA-enrichment on the enterprise?

The third research question aims at finding suitable cost analysis methods that can be used for business valuation using enterprise architecture. Though various types may exist, suitable for all kinds of analysis, it is interesting to know which ones can help in determining the value of enterprise architectures, based on costs. Possibly, more than one type is useful for cost-based EA-valuation. Therefore, we investigate whether or not these can be integrated into a singular cost-based analysis for determining the value of enterprise architectures.

RQ3: Which analysis techniques are suitable for analysing EA enriched with cost-related data?

RQ3.1: Which methods for cost analysis exist?
RQ3.2: How can these methods for cost analysis be used in architectures enriched with cost-related data?

The fourth research question is about which cost analysis data to show decision-makers: enterprise managers.

RQ4: What cost analysis data derived from enriched EA can be used as input for business intelligence applications to support enterprise managers?

RQ4.1: How can enterprise managers be supported by EA?
RQ4.2: What cost-related information is needed regarding decision support for enterprise managers?

Conclusions Part I

In this part, we have mentioned the central problem in this thesis, i.e. ‘how to combine enterprise architecture and operational data to better support decision-making?’. This leads from the following:

> ***Data warehouses*** (DWs) are used as a source for business intelligence to provide decision support (Sun & Heller, 2012), but they lack data traceability regarding enterprise processes and entities: business people need an explanation of the data to trace it back to their organization (Pettey & Van der Meulen, 2012). ***Enterprise architecture*** provides this type of traceability using familiar organizational entities and relating them in a structure, but is not suitable as a source for business intelligence applications, since it does not store operational data. ***Combining*** these two techniques hands possibilities for improving decision support to directly hand business people the data they need as evidence for decision-making, without having to trace it back to the location the data is showing something about: enterprise architecture may provide the meta data for operational data needed for business people to make decisions for their organization.

We have mentioned the objectives that are to be met at the end of the thesis to be able to solve the problem. The objectives were translated into research questions in order to accurately give an answer in our conclusions. These research questions are to be answered in the following parts of this thesis.

- Page intentionally left blank -

II. RESEARCH METHODOLOGY

As a start of our research, we need to set our scope to the area our research is taken place. Our research is performed within the *information systems* area, the main focus area for organizational informatics. Organizational informatics is interested in the *place of information*, *information systems* and *information technology* in various forms of human organization (Beynon-Davies, 2002). According to Beynon-Davies, these key elements of organizational informatics are the natural consequences of the need for humans to *communicate* and *coordinate* activity (Beynon-Davies, 2009b).

Our study aims to find out *how* to combine enterprise architecture and operational data to better support decision-making. The expression 'how' in the previous sentence hints for an *approach* or a *method* to be developed. Having stated that, we acknowledge the fact that our study will mainly be *artificial*, meaning it is subject to human expression and *interpretation* and therefore is not subject to the *empirically* research area. Our research attempts to serve as practical use in the field of enterprise architecture.

2.1 History of Design Science

In 1963, the term 'design science' was introduced as '*a systematic form of designing*' by R. Buckminster Fuller (Fuller & McHale, 1965). The concept was further described as a design method and compared to the scientific method concept a year later by Gregory (1966). Though Gregory claimed that design was not a science and that it referred to *the scientific study of design*, the area of 'design science' became more popular in the years after. Gregory's claim of design science being the 'scientific study of design', however, is currently predominating. Still, there has been a recurrent concern that 'design' should not be related to 'science' (Gregory, 1966)(Willem, 1990)(Cross, 2001). The term 'scientific study of design', however, does not assume that the acts involved in designing are scientific and this view on *design science* is therefore increasingly accepted in the scientific world (Gero, 2004).

For the area of design, spreading from engineers to architects and other professions that are design-oriented, a growing pressure remains for acting and deciding based on a systematic body of evidence (Van Aken & Romme, 2009). Simon's book 'The Sciences of the Artificial' motivated the need for the development of systematic and formalized design methodologies for the different design disciplines, like architecture, engineering and computer science (H. A. Simon, 1996). Suh outlined the principles of design and constituted an exposition of the design axioms and their applications, aiming to bring a scientific approach to the design-making process (Suh, 1990).

As an answer to the need for a systematic body of evidence for the information systems research area, Hevner and Chatterjee provided a reference on Design Science Research (DSR) in Information Systems (A. Hevner & Chatterjee, 2010) in which they amongst others outlined key principles of DSR.

2.2 Design Science in Information Systems Research

In 2004, Hevner et al. explained a set of seven guidelines meant to assist researchers in the information systems research area to structure their research based on design science (A. R. Hevner, March, Park, & Ram, 2004). Hevner et al. state that design science fundamentally is a problem-solving paradigm and mentions that ‘designing useful artefacts is complex due to the need for creative advances in domain areas in which existing theory is often insufficient’. The set of guidelines according to them may contribute to IS research ‘by engaging the complementary research cycle between design science and behavioural science to address fundamental problems faced in the productive application of information technology’. Via a conceptual framework, Hevner et al. describe the boundaries of design science within the IS discipline, focusing on technology-based design. The set of seven guidelines are (1) design as an artefact, (2) problem relevance, (3) design evaluation, (4) research contributions, (5) research rigor, (6) design as a search process and (7) communication of research.

2.3 Design Science Research Model

Based on previous research within the field of design science, Peffers et al. developed a general methodological guideline, the Design Science Research Model (DSRM) (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007). This process model, depicted in Figure 1 below, describes six activities derived from other theories and placed them in order as steps for the conduct of design science research. Compared to the guidelines found in Hevner et al. (A. R. Hevner et al., 2004) and other literature, the approach described by Peffers et al. states the procedurally steps to be taken. The ordered activities stated by Peffers et al. are: (1) problem identification and motivation, (2) define the objectives for a solution, (3) design and development, (4) demonstration, (5) evaluation and (6) communication. By demonstrating four cases, Peffers et al. (2007) showed how each of them followed a process consistent with the DSRM. These four cases reflect the four possible research entry points in the DSRM, namely the *problem-centered initiation, objective-centered initiation, design and development-centered initiation* and a *client/context-centered initiation*.

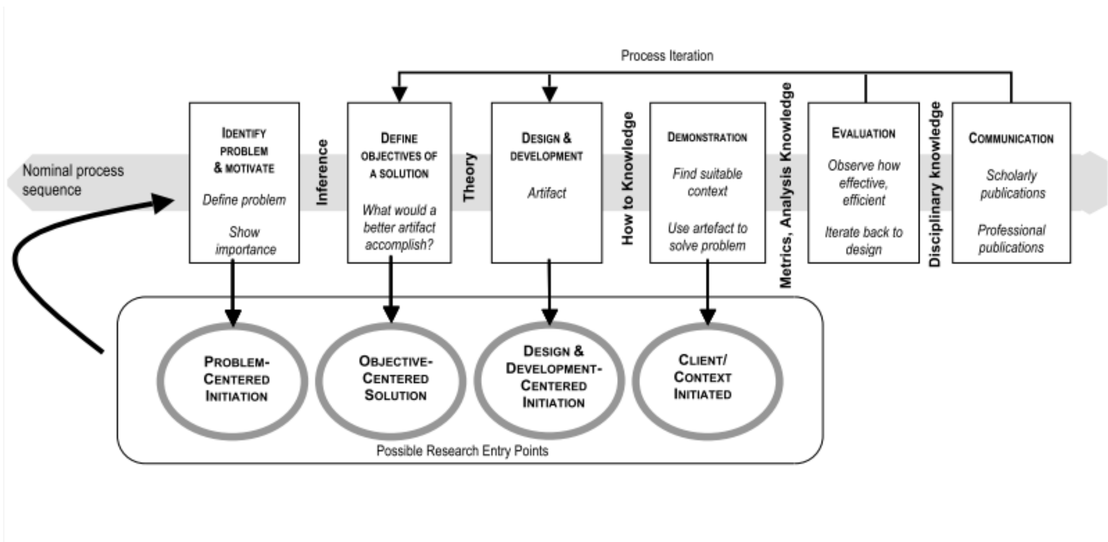


Figure 1. (Peffers, Tuunanen, Rothenberger & Chatterjee, 2007).

2.4 The Anatomy of a Design Theory

As an extension to the work of Walls, Joseph G.Widmeyer, George R.El Sawy (1992), Gregor & Jones (2007) developed an approach as an 'anatomy of a design theory' and identified eight separate components of design theories; (1) purpose and scope, (2) constructs, (3) principles of form and function, (4) artefact mutability, (5) testable propositions, (6) justificatory knowledge (kernel theories), (7) principles of implementation, and (8) an expository instantiation. The aim for the research of Gregor & Jones was to *"delineate the possible components of a design theory for IS, providing an ontological language for the discussion of these theories"* (Gregor & Jones, 2007).

Table 1. Eight components of an Information Systems Design Theory (Gregor & Jones, 2007)

Component	**Description**
Core components	
1. Purpose and scope (causa finalis)	"What the system is for," the set of meta-requirements or goals that specifies the type of artefact to which the theory applies and in conjunction also defines the scope, or boundaries, of the theory.
2. Constructs (causa materialis)	Representations of the entities of interest in the theory.
3. Principle of form and function (causa formalis)	The abstract "blueprint" or architecture that describes an IS artefact, either product or method/intervention.
4. Artefact mutability	The changes in state of the artefact anticipated in the theory, that is, what degree of artefact change is encompassed by the theory.
5. Testable propositions	Truth statements about the design theory.
6. Justificatory knowledge	The underlying knowledge or theory from the natural or social or design sciences that gives a basis and explanation for the design (kernel theories).
Additional components	
7. Principles of implementation (causa efficiens)	A description of processes for implementing the theory (either product or method) in specific contexts.
8. Expository instantiation	A physical implementation of the artefact that can assist in representing the theory both as an expository device and for purposes of testing.

2.5 Applying Design Science

Both design science approaches of Peffers et al. and Gregor & Jones were presented in 2007 and have been developed without using each other's theory. However, their ideas were based on many similar theories from earlier years. It is therefore interesting to see the differences and similarities between the two and the idea they have on how design science should be structured.

Peffers et al. use an iterative approach with different entry points, as research may have different reasons for performing the research. Gregor & Jones identified eight components that have a sequence and are logically numbered in such a way. The six activities mentioned by Peffers et al. show similarities with the eight components mentioned by Gregor & Jones. Whereas Gregor & Jones focus mainly on the *presence* of a component, Peffers et al. emphasize the *sequence* of their mentioned activities. Peffers et al. show where the possible entry points are for conducting the research *and* where the loopbacks are. Both theories are holistic regarding design science for information systems research and can be used as a guide; in our opinion it depends if the conducted research is focusing on *phases* or not.

For conducting our research, we have tried to identify the similarities and differences as a means of using the best of both theories. Since we prefer to conduct our research based on phases, we use the theory of Peffers et al. as the main thread and add components of Gregor & Jones to emphasize certain aspects of the theory. To be able to distinguish the mentioned components from Gregor & Jones from the activities of Peffers et al., we use numbers and letters as shown in Table 2. Some of the components of Gregor & Jones might be used in several activities. We chose to dedicate these components to a specific activity, unless explicitly stated different, like components (b) and (f) stated below.

Table 2. Steps for research using a mapping of Peffers et al. (2007) and Gregor & Jones (2007).

Steps for research	**Activities from Peffers et al.**	**Components from Gregor & Jones**
(1) problem identification and motivation *including* **(a)**	(1) problem identification and motivation	(a) purpose and scope
(2) define the objectives for a solution	(2) define the objectives for a solution	(b) constructs
(3) design and development *including* **(b), (c)** and **(d)**	(3) design and development	(c) principles of form and function
(4) demonstration *including* **(h)**	(4) demonstration	(d) artefact mutability
(5) evaluation *including* **(e)**	(5) evaluation	(e) testable propositions
(6) communication *including* **(g)**	(6) communication	(f) justificatory knowledge (kernel theories)

		(g) principles of implementation
		(h) expository instantiation
Note: **(b)** is used in all activities to explain concepts of theory and **(f)** is used to link all the above mentioned activities.		

2.6 Explanation of Design Science application and further steps

Having explained the approach we take to construct our research, based on theories of Peffers et al. (2007) and Gregor & Jones (2007), we briefly explain the start of our research and further steps to take. As previously stated, our study is based on solving the following problem:

> **How to combine operational data and enterprise architecture to better support decision-making?**

This conforms to the problem-centred approach stated in the DSRM (Figure 1) and *step 1* in our design science approach depicted in Figure 2. We have identified and motivated our problem in Part 1, Chapter 1.1.

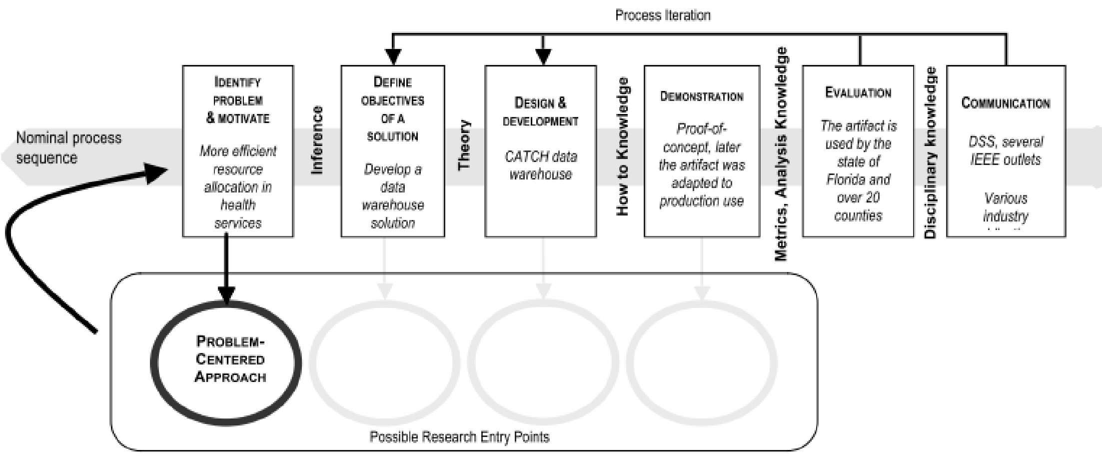


Figure 2. Application of the DSRM – Conducted Research Methodology.

According to the DSRM (Figure 2), *step 2* is to define objectives of a solution for the problem. The objectives needed are explained in Chapter 1.2 along with related research questions that can be posed in order to find answers and achieve the objectives, explained in Chapter 1.3.

Due to the use of two theories and the confusing impact this may have on our research, we will provide a thesis structure and reading guide in the next Chapter, Chapter 2.6. Chapter 2.7 will explain the practical and scientific relevance of our research, which conforms to the *purpose* part of component (a), described in Table 1. The scope of our research is given in the problem statement (Part I, Chapter 1.1). Chapter 2.9 states the used modelling languages that are used in this research. The modelling languages are a representation of component (b), described in Table 1. Part II of this research discusses the most relevant concepts needed for designing our artefact. Explaining these concepts conforms to component (b), described in Table 1. Part II is the start of *step 3*, the design and development phase. Part IV explains the first idea of combining enterprise architecture and operational data and explains the topic of decision-making in organizations. Part V explains our proposed methodology. Part VI demonstrates our methodology. Part VII evaluates on the proposed work and Part VIII hands conclusions drawn from our research.

2.7 Thesis structure and reading guide

Based on the DSRM process model, we distinguish several phases in our research to clarify the steps we take. The phases succeed one another chronologically, but research questions may be diffusedly answered throughout the thesis. For this reason, the traceability matrix is shown below in Table 3.

Table 3. Thesis structure and traceability matrix.

Part		**Applicable DSRM Phase**	**Research Questions**
I	Introduction	Problem identification & motivation	-
II	Research Methodology	Problem identification & motivation	-
III	Theoretical Framework	Problem identification & motivation Define objectives of a solution	RQ1.1 – RQ1.2 RQ2.1 – RQ2.3 RQ3.1 RQ4.2
IV	Combining Enterprise Architecture & Operational Data to better support Decision-Making	Define objectives of a solution Design & development	RQ4.1
V	Organization Lifecycle Approach to better support Decision-Making	Design & development	RQ3.2
VI	Demonstration – Timber Case Study	Demonstration	RQ3.2 RQ4.1 – RQ4.2
VII	Evaluation	Solution evaluation	
VIII	Conclusions	Communication	*All research questions*

2.8 Research relevance

Our research may serve two worlds, both the practical and theoretical world. Wieringa (2010) defines relevance as 'the suitability of an artefact or of knowledge to help achieving a goal, and applicability as sufficient incorporation of conditions of practice in a theory or in artefact behaviour'. Using this definition, we split up our research relevance and describe it for practical purposes as well as research or knowledge purposes.

A. Practical relevance for research

Our study will possibly serve as a guideline for practitioners trying combine enterprise architecture and operational data for decision-making purposes. The approach we present helps in taking steps to combine both worlds of enterprise architecture and business intelligence. Our study shows a possible execution of the explained theory by means of a cost-based approach. Also, we present a model to store *enterprise architecture*, *operational data* and *time* that may be used for analysis purposes that were not possible beforehand, like forecasting, amongst others.

B. Scientific relevance for research

For scientific research, our study aims to define methodology in an area that currently has no well-defined paths and exemplifies this by means of using a cost-based approach. Using enterprise architecture as a method to structure and enrich enterprise data, we indicate possible benefits in data quality for managers. Having said this and looking at the theory of Wieringa (2010), we could say our relevance is in *'achieving an economic goal'*, namely to lower costs in decision support for enterprises and *'improving performance'* in delivering qualitative data . Also, the model we present to store *enterprise architecture, operational data* and *time* may serve new possibilities for data analysis.

2.9 Modelling languages and notations

A. ArchiMate

ArchiMate®, an Open Group Standard, is an open and independent modelling language for enterprise architecture that is supported by different tool vendors and consulting firms. ArchiMate provides instruments to enable enterprise architects to describe, analyse and visualize the relationships among business domains in an unambiguous way (The Open Group, 2013b).

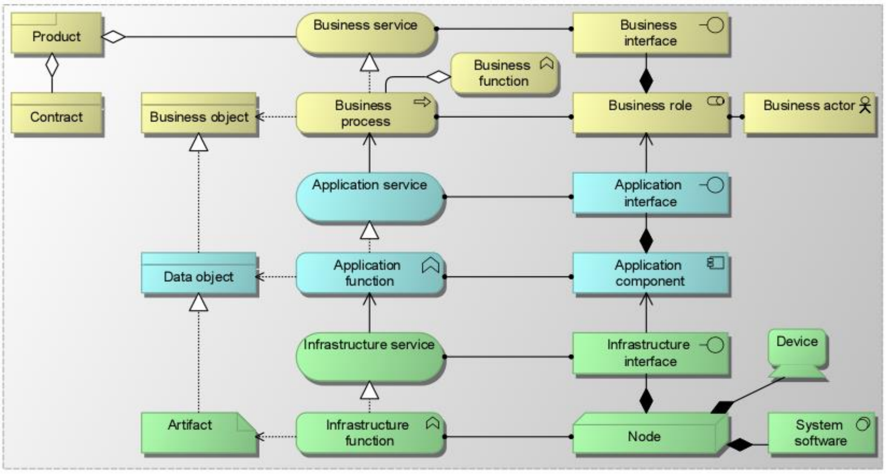


Figure 3. Example of an Enterprise Architecture with different concepts and relations.

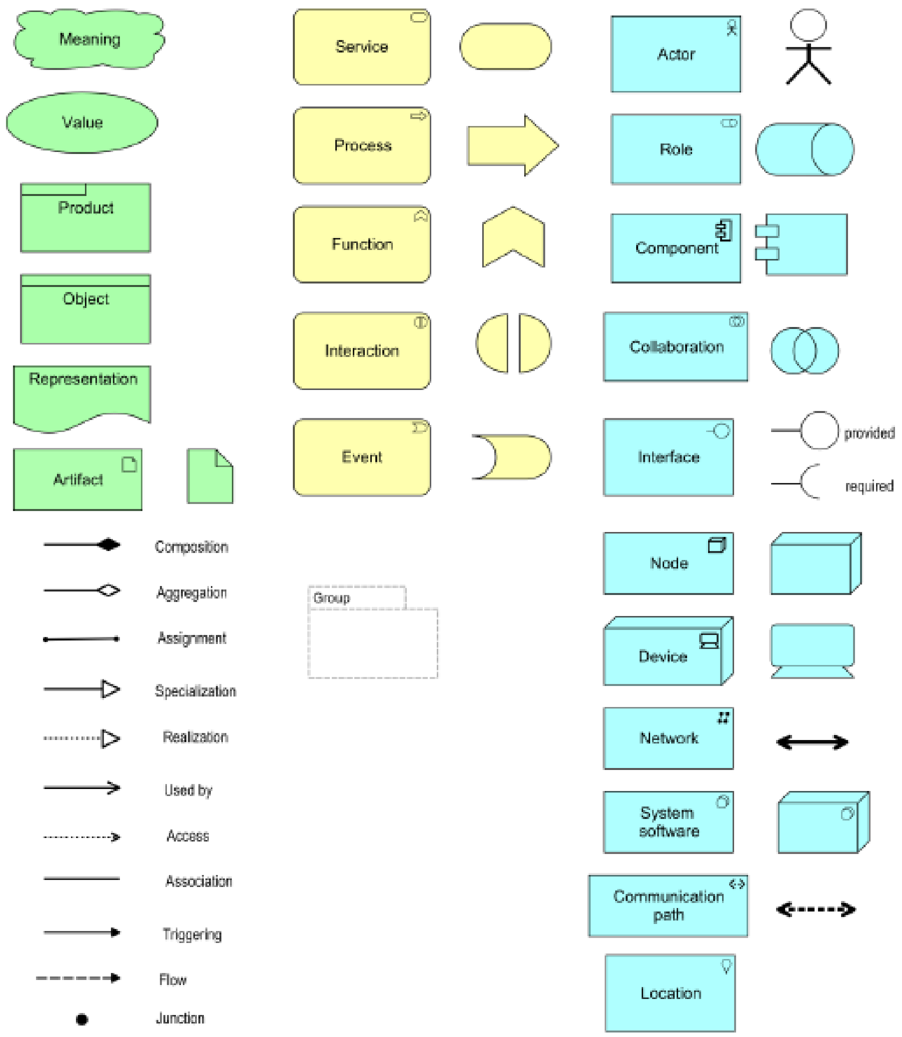


Figure 4. Core concepts and relations of the ArchiMate enterprise architecture modelling language.

B. BPMN

BPMN is a business process modelling notation. The primary goal of BPMN is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes. Thus, BPMN creates a standardized bridge for the gap between the business process design and process implementation. Another goal, but no less important, is to ensure that XML languages designed for the execution of business processes, such as WSBPEL (Web Services Business Process Execution Language), can be visualized with a business-oriented notation (Object Management Group (OMG), 2011).

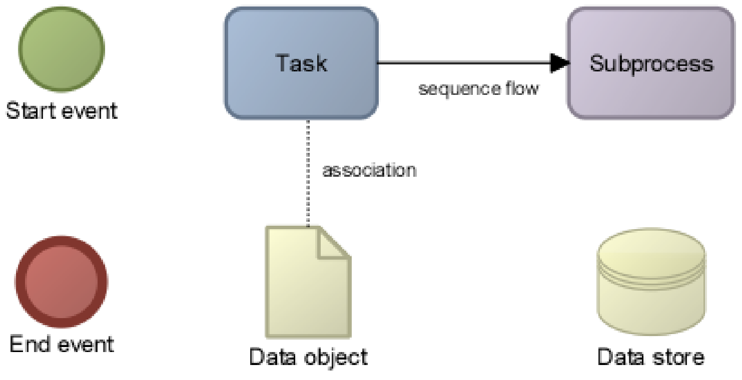


Figure 5. Used concepts of the Business Process Modelling Notation.

C. Crow's Foot

Crow's Foot is a commonly used notation for database modelling, due to its ability to give a clear view on databases that are complex (Barker, 1990). It shows attributes inside the tables and includes the primary and foreign key definitions for an entity relation diagram (ERD), which is why the notation is widely accepted for data modelling. Several data modelling suites like Microsoft Visio or online tools like Vertabelo.com have implemented the notation e.g. to create databases. Entities are represented as boxes and relations are represented as lines, as illustrated below in Figure 7. Here, different cardinalities are represented using different shapes. An example of two related entities can be read as "one *actor (can) perform(s)* zero, one or more *songs*", as illustrated in Figure 6.

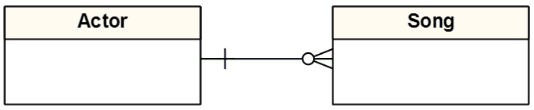


Figure 6. An example of two related entities using Crow's Foot notation.

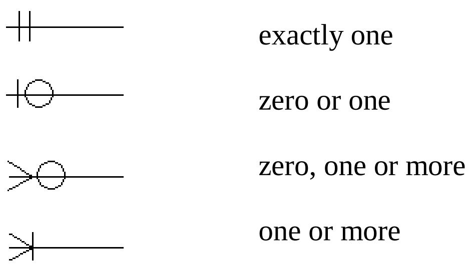


Figure 7. ERD relations – Crow's Foot notation.

Conclusions Part II

We have explained the history of design science that serves as a background for why design science was chosen as a research methodology, moreover, why the Design Science Research Methodology (DSRM) was used as a guidelines throughout this thesis. We have provided an overview of how this thesis should be read and where the research questions are addressed in this thesis. The research relevance was mentioned, serving as a reason for performing a study like ours. Lastly, we have shown and explained the modelling languages used in this thesis to be able to understand the models we use.

- Page intentionally left blank -

III. THEORETICAL FRAMEWORK

The term 'information' is used in many contexts and is often used as a buzzword rather than an understanding. This chapter tries to clarify this concept by explaining its origin and relating it to the enterprise context, given that information is a fundamental concept in our research.

3.1 Definitions of information, data and related concepts

These days, the concept information is often used in one or two ways; on one hand it might be narrowly defined as the 'transmission of bits', on the other hand it is often used as a buzzword, making it an important term with a poorly defined and understood definition (Beynon-Davies, 2009a). Using the concept information, it is difficult to distinguish to which of the ways one is referring to. Even though it remains a blurry concept, the concept is widely used and considered to be important in many contexts, like enterprise areas or consumer areas.

> *"Information is a paradoxical resource: you can't eat it, you can't live in it, you can't travel about in it, but a lot of people want it" (Stamper, 2001)*

As Stamper stated, we believe a definition of the concept of information may clarify why people think it is important enough to want it. Therefore, we will briefly explain the history of information in order to be able to understand its importance.

A. History of information

Information is a concept known to humans since the beginning of time. In fact, information is particularly associated with human communication. This communication involves *signs*, and in turn, these *signs* involve human interpretation (Beynon-Davies, 2009a). To be able to understand the concept of information, we need to go deeper into its multi-faceted nature.

Semiotics or semiology is the study of signs (Beynon-Davies, 2009a). According to Beynon-Davies, signs are seen as 'the core element of concern, serving to link issues of human intentions, meaning, the structure of language, forms of communication transmission, data storage and collaborative action'. As he explains, the world in which humans are living is revolving around *systems of signs*. Such a sign-system is described as *any organized collection of signs*. An example of such a sign-system is the every-day spoken language people use to communicate.

People have been using sign-systems for thousands of years to communicate (Beynon-Davies, 2009a). Looking at the earliest sign-systems found, the ancient Sumer token-systems with simple shaped clay tokens (Schmandt-Besserat, 1978), we see the tokens to represent concepts like livestock. The tokens later evolved into more complex shapes representing crafted goods and such, showing that the token-system evolved in time. These complex tokens had different shapes and were carved; what we see here is that objects are given meaning and meaning apparently changes in time.

The concept of information should therefore not be separated from time. If information is subject to human interpretation, and human interpretation changes in time, this affects the meaning people give to objects. Simply stated, if we dedicate a meaning to an object, we should keep in mind the period of *time* and the type of *humans* that gave the object a meaning, since culture and other facets may or may have influenced the human mind-set.

B. Information in action

Pragmatics, semantics and *syntactics* were proposed as the three branches of semiotics by Charles Morris (Morris, 1964). More than a decade later, Stamper describes these three branches in a layered model and adds a fourth branch, *empirics* (Stamper, 2001).

The four layers caught in this 'semiotic ladder' represent the concept of information as a *"socio-technical phenomenon, interposing between three different levels of system of interest to organizational informatics: activity systems, information systems and ICT systems"* (Beynon-Davies, 2009a). The semiotic ladder supports the connection of the social world on one hand with the technical world on the other hand. The ladder was based on the framework of semiotics by Stamper (Stamper, 2001). It shows the total model of the concept of communication through actions by means of signs in the *activity system* and links this to the *ICT system* via the *information system* (Figure 8).

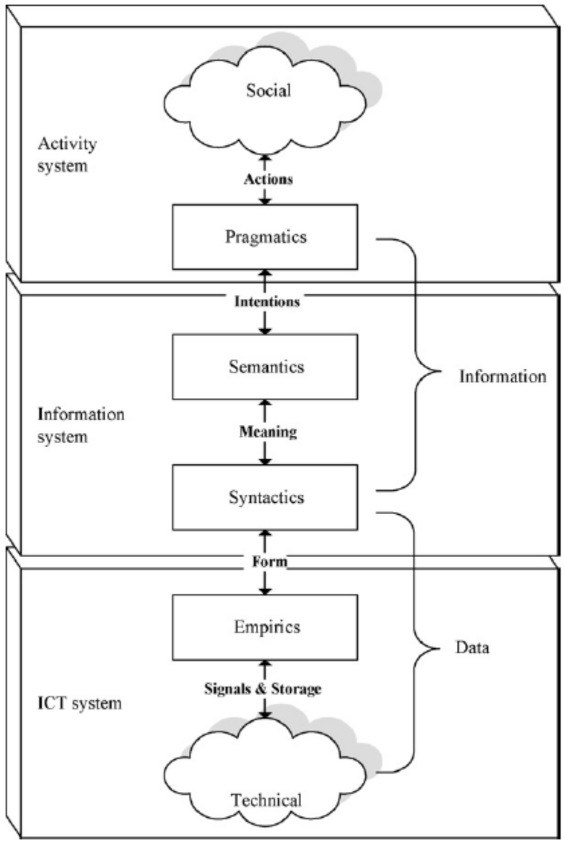


Figure 8. Levels of semiotics (Beynon-Davies, 2009a).

For our research, we are particularly interested in the link with *informatics*; the activity system shown in Figure 8 is a subtype of a social system. It is also described as a *human activity system* by Checkland & Scholes (Checkland & Scholes, 1999). The link between the social world and the technical world is described via the concepts *pragmatics, semantics, syntactics* and *empirics*. In Figure 8, we can see the difference between data and information. Whereas *information* is used for communication in the activity system as well as the information system, *data* is used for communication between the information system and the ICT system. For the full explanation of the semiotic ladder depicted in Figure 8 (Stamper, 2001), we quote its following description (Beynon-Davies, 2009a).

"Activity systems are linked to sign-systems through purposive acts of communication. ***Pragmatics*** *is concerned with such purpose of communication. Pragmatics links the issue of signs with that of intention. The focus of pragmatics is on the intentions of human agents underlying communicative behaviour. In other words, intentions link language to action.* ***Semantics*** *is concerned with the content or meaning of a message conveyed in a communicative act. Semantics is the study of the meaning of signs – the association between signs and the 'world' and hence can be considered as the study of the link between symbols and their referents or concepts".*

"**Syntactics** is concerned with the formalism used to represent a sign. Syntactics as an area studies the form of communication in terms of the logic and grammar of sign-systems. Hence, syntactics is devoted to the study of the form rather than the content of signs and sign-systems. **Empirics** is the study of the signals used to carry or code the signs of a message; the physical characteristics of the medium of communication. The area of empirics is devoted to the study of communication channels and their characteristics, e.g., sound, light, electronic transmission, etc.".

C. *Information and communication*

Beynon-Davies furthermore explains that information is related to the concept of *communication*, where signs are tools of communication (Beynon-Davies, 2009a). Shannon (1948) described in the Shannon and Weaver model that communication primarily has two agents, a *sender* and a *receiver* (Shannon, 1948). Beynon-Davies (2009b) used this concept to map the semiotic ladder and added the key elements *intentions, messages, language, signals and communication channels*. The mapping of the semiotic ladder and the communication concept is depicted in Figure 9, the communication process.

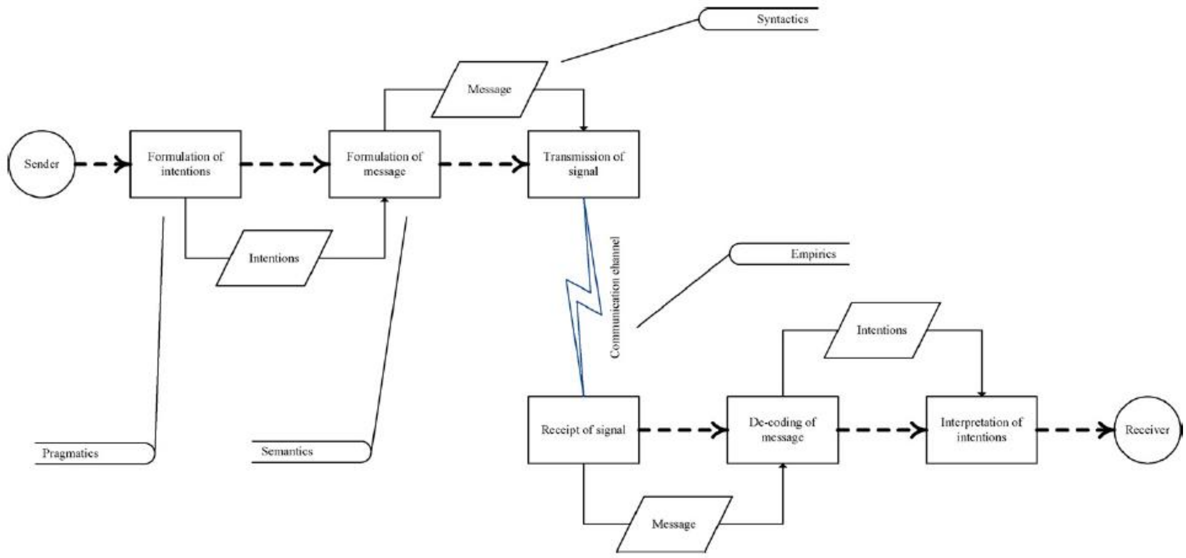


Figure 9. The communication process (Beynon-Davies, 2009a).

In Figure 9, we see the flow of concepts needed for communication going from sender to receiver. As described by Beynon-Davies and as depicted in Figure 8, the concept of information exists within the span of pragmatics and syntactics. It is called data later in the communication process going from syntactics, depicted as the concept 'message' in Figure 9, across the transmission and reception of signals (empiric level) towards the concept of message again, which is again the syntactic level.

As explained by Beynon-Davies, *"data are concerned with the form and representation of symbols in storage and transmission"*, whereas *"information is concerned with the meaning of symbols and their use within human action"* (Beynon-Davies, 2009a).

> *"Hence, information is data plus sense-making." (Beynon-Davies, 2009a).*

This definition of information is rather concise; many definitions of information exist in literature. For our research, this definition is useful for its clear distinction from data.

D. Definitions of information, data and related systems

Within the context of organizations or *enterprises*, a lot of information is being communicated. Also lots of data is stored at many locations in the enterprise. Looking back at the previous section, we have learnt there is a difference between data and information. However, a clear view on what information within enterprises actually is remains to discover.

> *"Information is concerned with the meaning of symbols and their use within human action"* (Beynon-Davies, 2009a)

Depending on the enterprise entity, many different types of enterprise information may be available. Information is subject to interpretation, as we can see in Figure 9. Whatever the intentions are from the receiver that reads the information, the intentions determine the way information is interpreted. In other words, it depends on the way we look at information to be able to give a meaning to it. Linking the concept of information to business organizations or *enterprises*, we come up with the following definition of enterprise information.

> *Enterprise information is concerned with the meaning of symbols and their use within human action in an enterprise (i.e. business organization) context.*

To be able to distinguish enterprise information from enterprise data, we use the following definition of enterprise data.

> *Enterprise data are concerned with the form and representation of symbols in storage and transmission in an enterprise (i.e. business organization) context.*

Having explained what information in an enterprise context means, we acknowledge the fact that there are *many possible forms or types of enterprise information*. It all depends on the interpretation of information, which in turn makes it possible to add certain labels like 'cost' or 'benefit' to information as a means of typifying the information. Looking at the definitions stated our research primarily entails the description regarding *enterprise data*.

Like the use of the term 'information' in all sorts of ways, the terms 'information system' and 'information and communication technology (ICT) system' and variations of both terms create confusion in the field of computer science and information systems research. Without claiming a strict definition of both terms, we feel the need to distinguish both, since we have previously explained what information is.

> In the context of signs, information systems can be described as *"a communication system used to support a given activity system",* which is *"mainly located against the semantic and syntactic levels of signs"* (Beynon-Davies, 2009a).

Information systems have *information* as an output; they use "agreed systems of signs to represent meaning". This description of information systems has a different meaning than the more common understanding that currently exists of an information system, namely the one of the ICT system.

> ICT systems represent a *"designed system of artefacts used to collect, store, process and disseminate data"*. ICT systems have *data* as an output and are *"located mainly at the technical and empirics level of signs"* (Beynon-Davies, 2009a).

3.2 Databases and data structures

Having stated the differences between data and information as well as what determines an information system and an ICT system in the previous chapter, we build on this understanding by explaining the location of data within ICT systems. For our research we aim to find cost-related enterprise data as a source for enriching enterprise architecture (EA), which is explained in the next chapter. In order to find this data, we need to know where this data is stored and how to identify it. In the previous chapter we explained the differences between information and data as well as information systems and ICT systems. This chapter explains the location of data within ICT systems.

For organizing digital data, *databases* are used to easily access, read and write data. Databases emerged in the early 60's of the last century as an answer to structure and store growing quantities of data in a time where computers emerged as successors of analogue data computing. A database is a collection of related data. However, not all related data thus makes up a database. Our meaning of a ***database*** has the following description.

> *"A database represents some aspect of the real world, sometimes called the 'miniworld' or the universe of discourse (UoD). Changes to the miniworld are reflected in the database. A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database. A database is designed, built, and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in which these users are interested." (Elmasri & Navathe, 2010)*

These mentioned characteristics of a database give an idea of how a database might look like. Database technologies can be split up into three eras of development, namely the *navigational database, relational database* and the *post-relational database* types. Many subtypes of these database types were developed over time. Currently, the relational database and post-relational databases are often used for software development. A relational database, or ***SQL (Simple Query Language)*** database, in its simplest form can be compared with a sheet of columns and rows (Figure 10). Here, columns are given names and rows (called records) are given identifiers. Each cell is called a *field*.

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

Figure 11. An example of a database storing school information (Elmasri & Navathe, 2010).

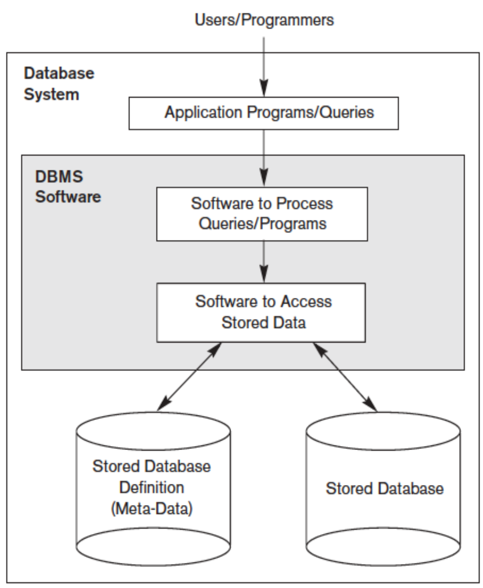


Figure 10. Simplified database system environment (Elmasri & Navathe, 2010).

In this way, all sorts of data can be stored in such a two-dimensional ‘rack’. Data can be accessed, read and written using *queries* and *transactions* using a ***database management system (DBMS)***, which is a collection of programs that enables users to create and maintain a database (Elmasri & Navathe, 2010).

It is often a suite of computer software that enables the interface between users and databases. A ***query*** is a command to the database and has a fixed set of terms to be used for *retrieving* data. A command to the database that intends to read or write data is called a ***transaction***, however, the term ‘query’ is loosely used for both types of commands as well. An application program accesses the database by sending queries or requests for data to the DBMS. For stored data to be logically structured, programmers need to be able to store logic somewhere in the database, the catalogue. This logic is also called ***metadata*** (Elmasri & Navathe, 2010).

In formal terminology, a row is called a *tuple*, a column header is called an *attribute* and the table depicted in Figure 12 is called a *relation*. By interlinking multiple relations using so-called *keys*, logic is defined between relations. Figure 11 represents a database consisting of multiple relations.

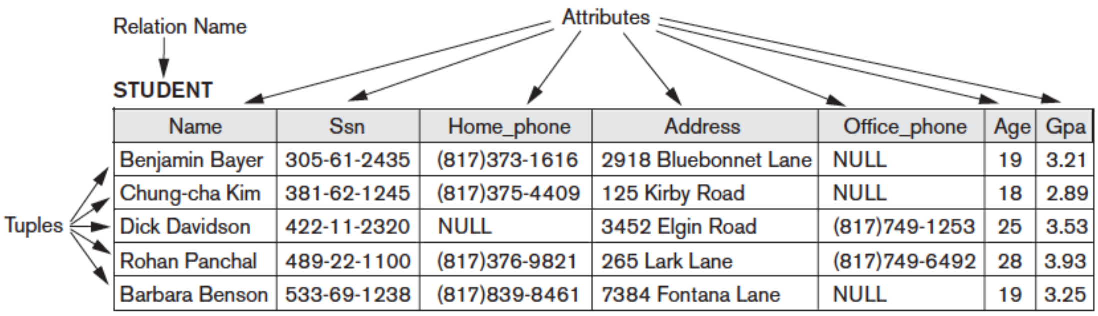


Figure 12. Graphical representation of a relation (Elmasri & Navathe, 2010).

In this graphical representation, connections are still missing. In other words, the keys are missing that define how these relations ‘relate’ to each other. In order to interrelate, each relation needs keys to be defined. This is often represented in schemas by underlining the attributes in each relation. Defining databases and their logic is done using a DBMS. Each DBMS needs a data definition language (DDL) for defining a relational database *schema*. Currently, the most used DDL is the Simple Query Language (SQL) (Elmasri & Navathe, 2010). Using *queries* or *transactions* in SQL, databases can be accessed and *manipulated* in different parts of the database that contain stored data.

Next to the previously explained ***relational database***, other database technologies were developed in time. A ***non-relational database***, also referred to as a NoSQL (No Simple Query Language) database, is the successor of the relational database. Whereas the relational database focuses on its relations notion, the non-relational database mostly focuses on its objects notion. Non-relational databases include document-oriented databases and key-value stores. They are often regarded to be faster and more flexible than relational databases. Though the term NoSQL may assume that NoSQL-based systems cannot be used in combination with SQL-based languages, the opposite is true.

Types of NoSQL databases, like the key-value stores, may be suitable for processing large quantities of data. Significant benefits of these types regarding the relational databases are often in terms of latency and throughput. This is the reason why many big data and real-time web applications nowadays use various types of NoSQL databases.

Both the relational database and the non-relational database are examples of a two-dimensional data structure or a multidimensional data structure. A ***one-dimensional data structure*** is an *array* of elements. Data is stored in memory as a contiguous sequence of elements, starting with *row 0* up till the last element stored in *row n*. Elements in these arrays can be found using indexing; these indexes are called *pointers*. Pointers themselves need not to be stored per se, but are computed at 'runtime', that is during calculations.

Looking at a relational database in its simplest form with a *row* and a *column*, this is an *array of an array*, which is a ***two-dimensional data structure***. An example could be a table with *product* as its rows and the *price* as its column. In this case, both *product* and *price* are the dimensions of the data structure. In any case, dimensions are picked as a view on relations. If a dimension has no relation with data from other dimensions, adding it is useless since the dimension will not state anything.

Dimensions might be picked from relations as illustrated in Figure 12. These dimensions can be used for multiple data structures, depending on what the data structure is intended for. For example, the relational table in Figure 10 might be used as a two-dimensional matrix, with one of the columns placed as a dimension on the y-axis and the others remain as a dimension on the x-axis. Data structures that are composed of more than one dimension are called ***multi-dimensional data structures***. These data structures are often used for their swiftness regarding querying, since it simplifies the access to the data that is queried for. Multi-dimensional data structures are also called ***cubes***. Though the verb 'cube' may presume that it only describes three-dimensional structures, cubes may in fact have more than three dimensions and still be referred to as being a cube.

In a nutshell, adding a dimension to a data structure might save up calculations. For example, querying in Figure 12 for a total of 'Gpa' for all students that are '19' of 'Age' makes it necessary to first *compare all values* in the 'Age' column to the number '19'. Then, the found records give their 'Gpa' and a total can be calculated. This querying is easier when the 'Age' column is made into a dimension on the y-axis, with all records consisting of the different ages possible. In this way, the query first finds the column '19' and then calculates the total values of 'Gpa': the comparison step was skipped.

Implementing dimensional data structures is typically done using schemas like the *star schema* or the *snowflake schema*. A ***star schema*** is a protocol for structuring data into *dimension tables*, *fact tables*, and *materialized views*. A snowflake schema is a form of a star schema. The star schema has a *dimension table* for each dimension containing a key column and one column for each level of the dimension (except the 'top' level T). ***Dimensions*** are used for two purposes: the *selection of data* and *grouping of data* and consist of levels, making up a hierarchy (Jensen, Pedersen, & Thomsen, 2010). Dimensions can be instantiated and can therefore be compared as a blueprint. Dimension instances are called '*dimension values*' or '*dimension members*', each value belonging to a particular level (Jensen et al., 2010).

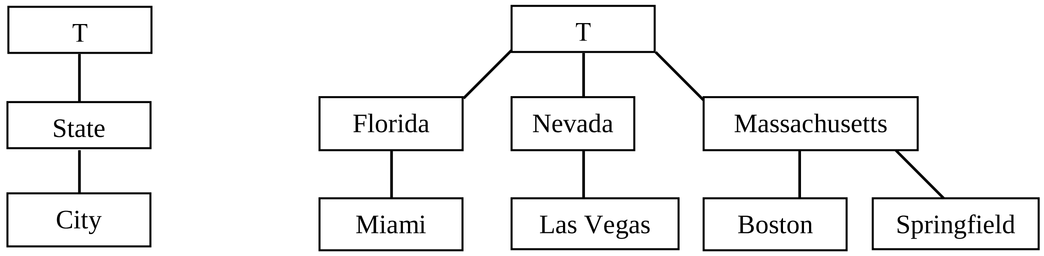


Figure 13. Schema (left) and instance (right) for a dimension 'Location'.

Dimensions may have multiple hierarchies, which may come in advantageous when representing dimensions for several situations, like representing a dimension ‘Time’ for both a fiscal and a calendar year. Multi-dimensional data structures often require ‘balanced trees’, meaning all branches of the tree should have the same height. However, possibilities exist to relax this restriction, according to (Jensen et al., 2010).

Multi-dimensional data structures contain *cells* at the dimensions’ intersections, like a spreadsheet has in the two-dimensional data structure. Such a cell has ***measures*** that are associated with it, which are numerical values that describe each measure, uniquely identifying the cell; meaning redundancy is lowered in multi-dimensional data structures. A cell can be empty, containing *null* values, or *non-empty*. These non-empty cells are called ***facts***. Measures consist of two components, namely the *numerical property* of a fact (e.g. total cost price) and a *formula*, which is often a simple aggregation function (e.g. SUM). Formulas are defined and stored in the metadata of the structure (Jensen et al., 2010).

The star schema uses the dimensions as *columns* in a ***fact table***. These columns are identified in the fact table by *foreign keys*, which refer to the *primary keys* of the respective dimension. ***Snowflake schemas*** use the same principle, but ‘bring in more depth’ to the dimensions. For example, a snowflake schema may have sub dimensions, e.g. a dimension ‘Time’ consisting of a Day-table, Month-table and Year-table. In this dimension, the Day-table contains references to the Month-table and the latter to Year-table. The Day-table will store the most granular data from this dimension, since it is the most specific. Figure 14 shows a star schema with dimensions *Time*, *Location* and *Fruit* and the fact table in the middle containing foreign keys to all dimensions. Figure 15 shows a snowflake schema with the same dimensions: it is essentially the same as the star schema, but specifies the dimensions more.

Having explained the basic concept of data storage, the concept of a database and the underlying data structures, we go on explaining the next phase of data processing: the data warehouse.

DayID	Day	MonthID
4545	5601	15
4545	5601	10

FruitID	Fruit
1201	Peach
1202	Apple

FruitID	DayID	CityID	Sales
1201	4545	5601	15
1202	4545	5602	10

CityID	City	StateID
5601	Miami	12
5602	Las Vegas	13

Figure 14. A star schema with dimensions *Fruit, Location* and *Time*.

MonthID	Month	MonthID
4545	5601	15
4545	5601	10

DayID	Day	MonthID
4545	5601	15
4545	5601	10

FruitID	Fruit
1201	Peach
1202	Apple

FruitID	DayID	CityID	Sales
1201	4545	5601	15
1202	4545	5602	10

CityID	City	StateID
5601	Miami	12
5602	Las Vegas	13

StateID	State
12	Florida
13	Nevada

Figure 15. A snowflake schema with dimensions *Fruit, Location* and *Time*.

3.3 The Era of Data Warehousing

Traditional databases have been the standard for operational environments ever since data needed to be stored digitally. Operational environments assume that at any moment in time there is only one correct definition of the structure of data (Inmon, 2005). However, it is valuable to track data and look at information over *time*. In order to see what happened in the past period to make predictions about what will happen in the future, the concept of *time* was added to data when *data warehouses* were introduced around 1983. According to Inmon (2005), trying to describe the time of operational environments, *"prior to 1983 there were applications [..], but someone looked up and decided that there was a need for information, not data"*. According to him, a period came with *"a need to look across the organization"* and not just a small fragment or as he calls it "*a tiny application area"*. From this period on, data warehouses in their simplest form were born and with it the concept of ***atomic data*** – data that cannot be sub divided. Atomic data, data at its lowest level of detail, provides the base for further data transformations. Being able to store atomic data in a data warehouse hands new possibilities regarding analysis and ***decision support***.

> A ***data warehouse*** is *"a subject-oriented, integrated, time- varying, non-volatile collection of data that is used primarily in organizational decision making"*. (Inmon, 2005)

The main reason for using a data warehouse is to be able to see data over time, being able to apply all kinds of analyses to this data in order to *support decision-making*.

> ***Data warehousing*** is *"a collection of decision support technologies, aimed at enabling the knowledge worker (executive, manager, analyst) to make better and faster decisions"*. (Chaudhuri & Dayal, 1997)

Going from an *decentralized situation* with data stored in ***operational source systems*** across the organization towards a *centralized situation* with data stored in a data warehouse brings in a new problem, namely how to transfer this data to the new environment and how to make sure this data is transferred correctly, i.e. make sure the data has the correct meaning it is intended to have.

William H. (Bill) Inmon et al. introduced the *Corporate Information Factory (CIF)* in 2001 as an approach to data warehouse design (Inmon, Imhoff, & Sousa, 2001). This approach was later on referred to as the ***top-down*** design approach. In this approach, the data warehouse is at the centre of the CIF and provides the 'spine' towards business intelligence.

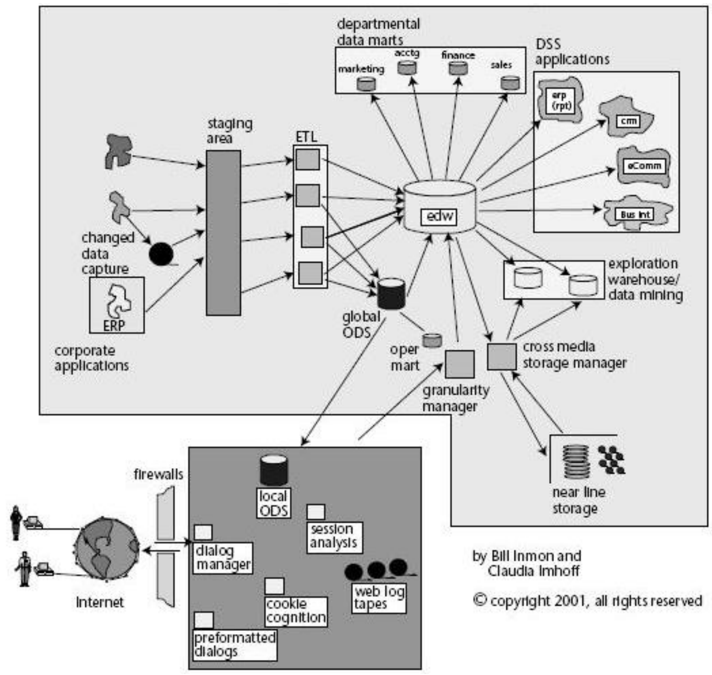


Figure 16. Corporate Information Factory (Inmon et al., 2001).

The texture of a data warehouse, consisting of its interlinking elements, is different in several approaches. We will discuss the specific parts of several approaches later on in this research. For now it is enough to know that different approaches towards data warehouse design exist.

In a situation with all kinds of interlinked operational source systems, organizations are in web in which it is *difficult to manage data*. Moreover, the more operational source systems are added, the harder it becomes to keep an overview and track where data is flowing. These situations already tended to happen decades ago, making it useful to transfer the data into a more central environment, like a data warehouse.

Doing all the things needed to prepare the data for entering the data warehouse is called 'entering the ***data staging area***'. The data staging area of the data warehouse is both a *storage area* and a *set of processes* commonly referred to as ***extract-transformation-load*** **(ETL)** (Kimball & Ross, 2002).

> *"**ETL technology** allows data to be pulled from the legacy system environment and transformed into corporate data. The ETL component performs many functions, such as logical conversion of data, domain verification, conversion from one DBMS to another, creation of default values when needed, summarization of data, addition of time values to the data key, restructuring of the data key, merging of records and deletion of extraneous or redundant data."* (Inmon, Strauss, & Neushloss, 2008)

ETL technology basically helps in *transforming* data coming from operational source systems into 'corporate data' flowing into the data warehouse. The transformation of data is usually performed inside an ***operational data store (ODS)***.

> *"The **ODS** is the place where online update of integrated data is done with online transaction processing (OLTP) response time. The ODS is a hybrid environment in which application data is transformed (usually by ETL) into an integrated format. Once placed in the ODS, the data is then available for high-performance processing, including update processing. In a way the ODS shields the classical data warehouse from application data and the overhead of transaction integrity and data integrity processing that comes with doing update processing in a real-time mode."* (Inmon et al., 2008)

An ODS is an optional part of a data warehouse (Inmon, 2005). It often contains 'current' data, i.e. data not older than a month, whereas a data warehouse contains 'historic' data, which is data that goes much further than a month to even decades of years. Typically, an ODS has a set time boundary of storing data, whereas the data warehouse has not.

Mentioned in the clarification of the ODS stated above, ***Online Transaction Processing (OLTP)*** is a concept that is best explained within the context of the earlier mentioned DBMS. OLTP refers to a 'class of information systems', which is a grouping of information systems using the same technology, namely transaction-oriented technology. Elmasri & Navathe (2010) explained the concept of OLTP as follows:

> *"A multiuser DBMS, as its name implies, must allow multiple users to access the database at the same time. This is essential if data for **multiple applications** is to be integrated and maintained in a single database. The DBMS must include concurrency control software to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct."*

The ‘multiple applications’ mentioned in this quote are also referred to as ***OLTP applications***. These applications are depicted in Figure 10 as operational source systems. The ODS might be implemented somewhere within the data staging area, but its functionality might also be used elsewhere. ETL technology is used between the different areas to manipulate data as explained earlier.

As shown in Figure 17, the data staging area is *“everything between the operational source systems and the data presentation area”* (Kimball & Ross, 2002). We use this image to explain several areas data is flowing through. Figure 17 is however used in a different approach, referred to as *bottom-up*. This approach is discussed in the next chapter.

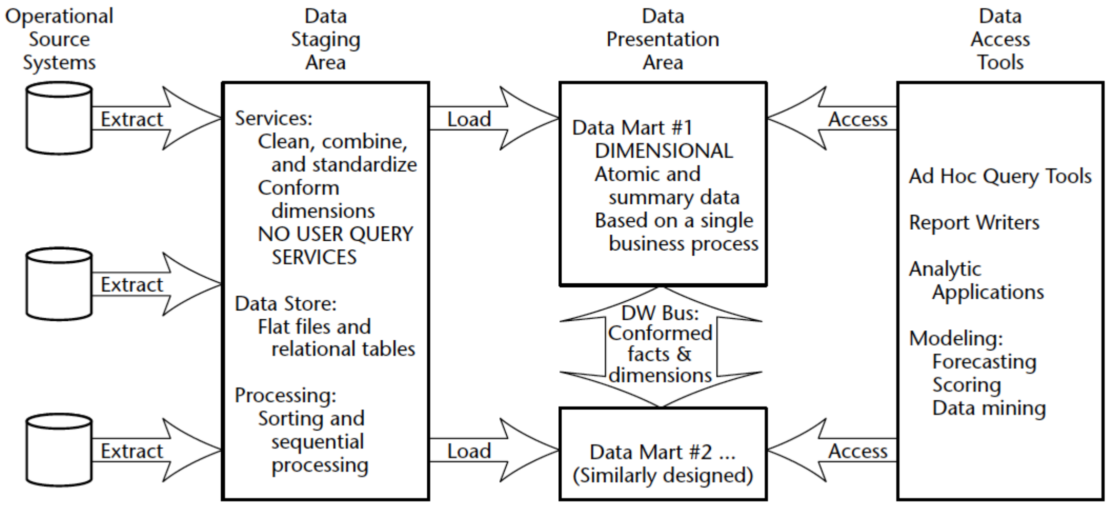


Figure 17. Basic elements of the data warehouse (Kimball & Ross, 2002).

Before information goes into the data staging area, it saves time to create a consistent landscape of operational source systems and applications. Integrating these systems is referred to as ***enterprise application integration (EAI)***. EAI is a complex process due to the different character of the source systems involved; they are often built for different purposes and use different techniques and terminology. Finding a technique to let the systems communicate is not enough; the data that is processed should become a ‘consistent source package’.

Using the ETL process in the staging area, EAI is an *effort* to ‘tidy up the mess’ before entering the central data warehouse. Due to the fact that in time systems have evolved, and with it the technology they consist of, it might be hard to integrate these systems. In fact, it is possible that systems will not completely integrate at all. When it seems impossible to match data between systems, it might mean that *data will be lost* when practicing EAI. Therefore, managers that are in charge of an EAI effort should know exactly what is needed in the data warehouse before they decide to evolve into a data warehouse environment.

After having moved from the operational source systems, through the data staging area, data is passed to the data presentation area. Kimball & Ross explain that the ***data presentation area*** is *“where data is organized, stored, and made available for direct querying by users, report writers, and other analytical applications”*. This area is what Inmon et al. (2001) might call the *‘enterprise data warehouse’* and what we would consider to be the actual data warehouse, the place where all the data is centrally stored. Additional components for the data presentation area are ***data marts*** and ***OLAP (Online Analytical Processing)*** technologies, which we will elaborate on in the next chapters.

Now that we have centrally stored corporate data, there are various options to apply on this rich dataset. ***Data access tools*** are *"the variety of capabilities that can be provided to business users to leverage the presentation area for analytic decision-making"* by means of *querying*, which is *"the whole point of using the data warehouse"* (Kimball & Ross, 2002).

3.4 Modern Day Data Warehousing

As explained, the concept of data warehouses has been introduced around several decades ago, but never stopped evolving until today. The definition of a data warehouse given by Inmon (2005), mentioned earlier, was stated back in 1992. This definition might give the idea that data warehouses are just *"mere copies of the operational system of record stored on a separate hardware platform"* (Kimball & Ross, 2002).

Though many data warehouses might be developed as a means to separate and isolate the operational environment from the data warehouse environment for performance reasons, Kimball & Ross (2002) claim that developing these 'pseudo data warehouses' is a *"disservice to data warehousing"*. According to them *"these imposters [..] don't acknowledge that data warehouse users have drastically different needs than operational system users"*.

Until now we have explained all kinds of concepts using the *top-down* approach towards data warehouse design as explained by Inmon et al. (2001). There is, however, an opposite approach, also known as the *bottom-up* approach towards data warehouse design, as explained by Kimball & Ross (2002).

Basically Kimball & Ross (2013) state that data warehouses should not be created to serve the operational system users, but to serve a different type of user; the ***business user***. Another way to look at the elements or concepts relating to data warehousing is represented the *Kimball DW/BI architecture* (Figure 18). Kimball & Ross (2013) use the concepts of *back room* and *front room* to express their view on data warehousing. In this perspective, all we have explained so far regarding data warehousing can be brought under the *source transactions* and *back room* areas.

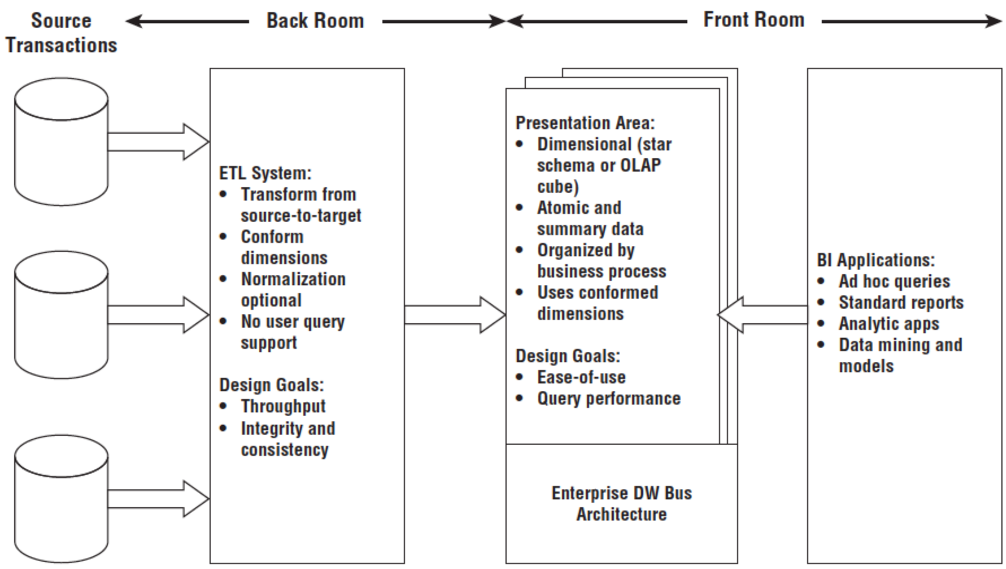


Figure 18. Core elements of the Kimball DW/BI architecture (Kimball & Ross, 2013).

This DW/BI architecture design is a way of looking at data warehouses. It is, however, not per se *the* way to build enterprise architecture. Both William H. (Bill) Inmon and Ralph Kimball are considered as ‘thought leaders’ in data warehousing. They are also discussed as ‘opponents’ in data warehousing due to their different views on how a data warehouse should be built. Whereas Inmon uses a *top-down approach*, with the data warehouse as the central hub in his ‘Corporate Information Factory’ as a result of merging data from the operational source systems, Kimball & Ross use the *bottom-up approach*, where the focus is on creating what is needed to *serve the business user*, like first creating *subsets of a data warehouse*, also known as ***data marts***, as a means to provide reporting and analytical capabilities used by ***OLAP (Online Analytical Processing)*** technologies.

3.5 Data warehousing architectures

Having explained two approaches for data warehouse design, we broaden our scope by relating these approaches to *data warehouse architectures*. Data warehouse architecture is not the same as a data warehouse design approach. Where an ***approach*** describes the *activities* that have to be performed and their *sequencing*, an ***architecture*** describes *component parts*, their *characteristics* and the *relationships* among the parts (Watson & Ariyachandra, 2005).

In time, many approaches have mingled and influenced one another due to new insights and lessons learnt from practice: an era of *architecture hybrids* has started. However, most of these hybrids are primarily based on five data warehouse architectures, namely (1) independent data marts, (2) data mart bus architecture with linked dimensional data marts, (3) hub and spoke, (4) centralized data warehouse (no dependent data marts), and (5) federated (Watson & Ariyachandra, 2005). Below we briefly describe the five different architectures to understand the concepts and relate them to both the approach described by Inmon et al. (2001) as well as Kimball & Ross (2002). The architectures were derived from Watson & Ariyachandra (2005).

A. Hub-and-Spoke architecture

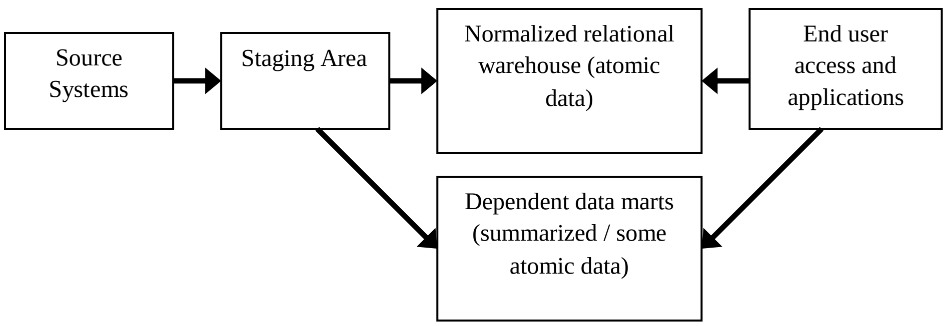


Figure 19. Hub-and-Spoke architecture.

The hub-and-spoke architecture (Figure 19) is developed in an iterative manner, each subject area at a time. In this architecture, atomic data is stored in the 3rd normal form (3NF) inside the ***central data warehouse (CDW)*** depicted as ‘normalized relational warehouse’ in Figure 19. Using the CDW as a source, data marts are developed *“for departmental, functional area, or special purposes (e.g., data mining) and may have normalized, denormalized, or summarized/atomic dimensional data structures based on user needs”* (Watson & Ariyachandra, 2005). Most users query the dependent data marts instead of querying the CDW directly. The Corporate Information Factory (CIF) described by Inmon et al. (2001) can be seen as an example of a hub-and-spoke architecture (Watson & Ariyachandra, 2005).

B. Data Mart Bus with linked Dimensional Data Marts architecture

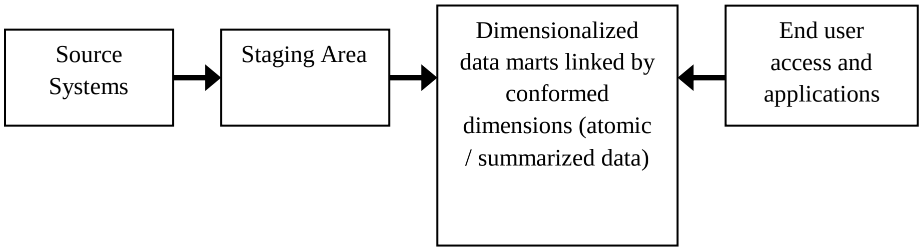


Figure 20. Data Mart Bus with linked Dimensional Data Marts architecture.

The data mart bus with linked dimensional data marts architecture (Figure 20) starts with a single data mart that is being built for a specific business process using conformed *dimensions* and conformed *facts*. In time, more data marts are being developed that make use of these dimensions, resulting in an integrated mart structure and an enterprise overview on the data. Atomic data as well as summarized data are maintained in a *star schema* to provide the dimensional view (Watson & Ariyachandra, 2005). Kimball's DW/BI architecture, described by (Kimball & Ross, 2013) may be considered as an example of this architecture.

C. Independent Data Mart architecture

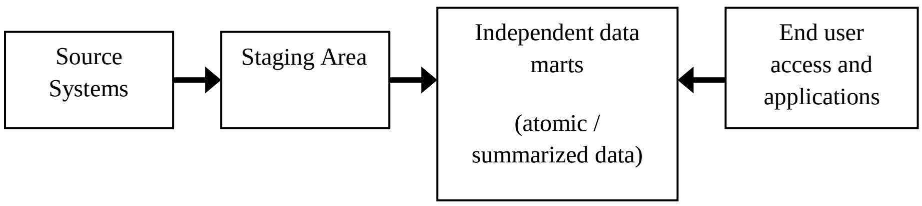


Figure 21. Independent Data Mart architecture.

The independent data mart architecture (Figure 21) is an architecture in which different data marts are separately created for multiple organizational units, like departments and divisions. Here, data marts do not use the data from other data marts. The separate data marts are developed for specific needs and therefore might use different definitions and measures of data, hence there is no *"single version of the truth"* across these data marts, which makes it difficult to analyse data (Watson & Ariyachandra, 2005).

D. Centralized Data Warehouse architecture

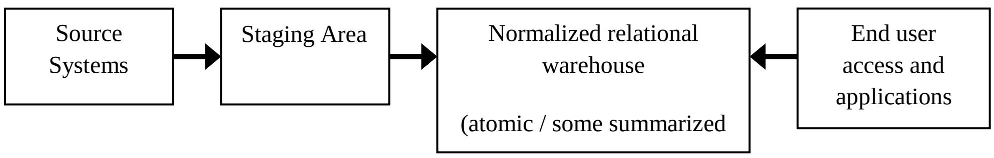


Figure 22. Centralized Data Warehouse architecture.

The centralized data warehouse architecture (Figure 22) has the same characteristics as the hub-and-spoke architecture, except no dependent data marts exist in this structure. It may be seen as the basic form of a data warehouse structure, where the central data warehouse takes care of queries coming from the end user. Data inside the central data warehouse is atomic, normalized and relational and some summarized data is available.

E. Federated architecture

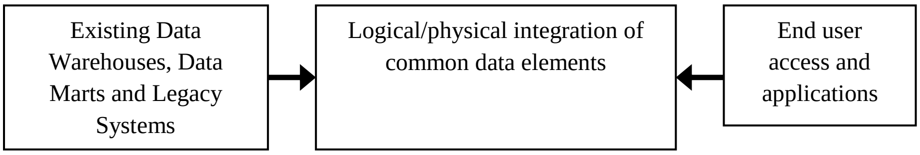


Figure 23. Federated architecture.

The federated architecture (Figure 23) is an additional structure that builds on existing structures like operational source systems, data warehouses, data marts and legacy systems. Based on a set of business requirements, data from the existing systems is being integrated in either a logical or physical way *"using shared keys, global metadata, distributed queries and other methods"* (Watson & Ariyachandra, 2005). The integrated data is used for querying by the end users.

Next to the mentioned data warehouse architectures, several other types and hybrids of the above mentioned architectures exist. In most cases, organizations determine the best option regarding their situation and add parts of other architectures to suit their requirements. Examples of hybrid architectures are *centralized data warehouse (CDW) with ETL architecture, centralized data warehouse (CDW) with Operational Data Store (ODS)* and *data warehouse (DW) with Integration Bus* (Asadullaev, 2009).

3.6 Data Warehouse Related Terminology

For our research, we use different concepts to illustrate our thoughts when answering our research questions. Experts in literature have described these concepts thoroughly, which makes it unnecessary to come up with our own definitions. For now, it is good to get an idea of the data warehouse related terminology before we proceed with our study.

We have previously explained that ***data marts*** are subsets of data warehouses. They are used for specific purposes when the total set of data found in the data warehouse is not needed, but a small part is. Data marts are often developed for parts of an organization, like departments.

> *"A data mart is the access layer of the data warehouse environment that is used to get data out to the users. The data mart is a subset of the data warehouse that is usually oriented to a specific business line or team. Data marts are small slices of the data warehouse. Whereas data warehouses have an enterprise-wide depth, the information in data marts pertains to a single department"* (Inmon, 2005).

Looking at a data perspective, *"the departmental or data mart level of data is shaped by end-user requirements into a form specifically suited to the needs of the department"* (Inmon, 2005). Hence, the end-user determines the needs for such a data mart. The data inside a data mart differs fundamentally from data inside a data warehouse. Whereas a *data warehouse* contains *'granular'* data with the highest level of detail, the *data mart* contains *shaped* data, suitable for the end-user needs. The data mart operates isolated and addresses the department's *analytic* requirements (Kimball & Ross, 2013). Data marts have different characteristics from data warehouses; they are faster, since they were tailored for specific needs.

> *"These data marts enable faster roll out, since they do not require enterprise-wide consensus, but they may lead to complex integration problems in the long run, if a complete business model is not developed"* (Chaudhuri & Dayal, 1997).

If data marts are used as explained in the *independent data mart architecture*, organizations may face difficulties when trying to integrate multiple data marts in time. This is because the data inside these data marts do not contain 'a single version of the truth' due to the different measures and definitions of data.

When data is stored somewhere in the organization, e.g. in a data mart, ***OLAP (Online Analytical Processing)*** technology can be used to represent data in a multidimensional view. The term "Online" or "On-Line" refers to fast and interactive query response that is implied with the technology. Whereas OLTP focuses on the previously explained 'transactions', OLAP focuses on data analyses. In a nutshell, ***OLAP systems*** *"provide fast answers to queries that aggregate large amounts of so-called detail data to find overall trends, and they present the results in a multidimensional fashion"* (Jensen et al., 2010). We have previously explained multi-dimensional data models. OLAP models are also referred to as ***cubes***.

OLAP systems are labelled in three broad categories, namely: "systems based on *relational database management technology*, called ***ROLAP*** systems, systems utilizing *non-relational, multidimensional array-type technologies*, called ***MOLAP*** systems, and *hybrid systems that combine these technologies*, called ***HOLAP*** systems" (Jensen et al., 2010). OLAP technology is often the source for business intelligence applications, which are used for decision support.

> *"Business intelligence (BI) is an umbrella term that includes the applications, infrastructure and tools, and best practices that enable access to and analysis of information to improve and optimize decisions and performance"* ("Gartner IT Glossary - Business Intelligence (BI)," 2013)

As explained by Gartner in the definition above, the term business intelligence is an 'umbrella term', which often comes down to the fact that nobody exactly knows the definition and therefore define one themselves. Comparing this definition from Gartner to Forrester's definition may give us some overlapping concepts on which they agree upon what defines business intelligence.

> *"Business intelligence (BI) is a set of methodologies, processes, architectures, and technologies that transform raw data into meaningful and useful information. It allows business users to make informed business decisions with real-time data that can put a company ahead of its competitors. Traditionally, core features like reporting and analytics have been the focus of BI technology choices, but as those features get commoditized, a whole new set of possibilities has emerged"* ("Forrester Research - Topic Overview: Business Intelligence," 2013).

Clearly, both Forrester and Gartner agree that the term business intelligence covers multiple concepts used for decision support. In Forrester's definition, we may get a grasp of what separates *business intelligence* from *data warehousing* technology, namely that business intelligence is a set of concepts *'that transform raw data into meaningful and useful **information**'*. This is exactly what differs DW from BI: whereas data warehousing is all about data, business intelligence is about transforming the data into what we might consider as *information*. Looking back at what we have previously explained about *data* and *information*, business intelligence adds *pragmatics, semantics* and *syntactics* to data. However, the field of business intelligence is rapidly changing and technologies are merged into applications like business intelligence suites. To capture the essence of business intelligence and still be able to cope with this evolving field, we conform to Kimball's definition of business intelligence (BI).

> *"**Business intelligence (BI)** is the range of capabilities provided to business users to leverage the presentation area for analytic decision making"* (Kimball & Ross, 2013).

When building a data warehouse, developers should bear in mind that eventually end users will use the DW as a source for business intelligence. These end users might use several techniques to retrieve this data; one of them is 'ad hoc querying'. ***Ad hoc querying*** means *"accessing data with any meaningful combination of values for the attributes in the dimension or fact tables"* (Elmasri & Navathe, 2010). Ad hoc querying is performed using data access tools, explained earlier. Data warehouse designers and developers should try to imagine the possible queries end users might perform in order to fulfil the goal of the data warehouse, which is *querying* (Kimball & Ross, 2002).

Reporting is another technique that might be used by end users. Reports are useful regarding several purposes, like tax preparation, and are defined by predefined queries that make up the format of the report. As stated by the 'father of the relational theory' E.F. Codd, *"Reporting must be capable of presenting data to be synthesized, or information resulting from animation of the data model according to any possible orientation"* (Codd, Codd, & Salley, 1993). According to him, *"this means that the rows, columns, or page headings must each be capable of containing/displaying from 0 to N dimensions each, where N is the number of dimensions in the entire analytical model"*. Codd is referring to an analytical model used as a basis for reporting.

Analytical models are designed to analyse data and find *patterns*. These models use predefined algorithms to define and predict behaviour of *objects*, which can be everything in an organization as long as it can be identified in data. Analytic models are used in the field of ***data mining***, which is *"the process of analysing large amounts of data in search of previously undiscovered business patterns"* (Inmon, 2005).

Having explained the principles of data warehousing and related concepts, we go on with a different concept, namely Enterprise Architecture (EA).

3.7 Enterprise Architecture (EA)

Enterprise architecture is a relatively young field of practice and formally exist less than 20 years (Paras, 2013). Modern day enterprises are challenged with changes in their environment, like *governmental legislation, power shifts* in the value chain, the *changing role of IT* in organizations and so on (Greefhorst & Proper, 2011). According to Greefhorst & Proper, these challenges impact the *design* of an enterprise; from the *products* and *services* an enterprise offers to its clients, via the business processes that enable them, and the information systems that support these processes, towards the underlying IT infrastructure. Modelling the characteristics of an enterprise, from the business layer, via the application layer (or information systems layer) towards the technical infrastructure layer, makes it possible to provide views for all kinds of stakeholders that might benefit from the insights provided. Enterprise architecture enables the translation of business stakes towards execution, handing a base for future *decision-making.*

> *"**Enterprise architecture (EA)** is the organizing logic for business processes and IT infrastructure, reflecting the integration and standardization requirements of the company's operating model. The enterprise architecture provides a long-term view of a company's processes, systems, and technologies so that individual projects can build capabilities – not just fulfil immediate needs."* (Ross, Weill, & Robertson, 2006)

Enterprise architecture is becoming increasingly important. The government of the United States of America even requires governmental agencies to appoint a CIO with 'sound and integrated information technology architecture' as a responsibility to have. This requirement was enacted in the Clinger-Cohen act (Greefhorst & Proper, 2011). As such, Greefhorst & Proper explain that enterprise architecture (EA) is an *"instrument to direct an enterprise's future direction, which also serving a coordination and steering mechanism toward the actual transformation of the enterprise"*. In other words, EA brings *structure* and *coordination* for decision-making to an enterprise.

The principle of EA has been adopted in research for over the past decades and is being implemented at many, often large, organizations. The field of EA currently has an increasing amount of standards, like TOGAF (The Open Group Architecture Framework) and ArchiMate. These standards help in the maturation of the EA field and give hands on for EA implementation in organizations. Overall, the standards often represent three different kinds of 'layers', namely *strategic*, *tactical* and *operational* and their interrelation. To practise the field of EA, organizations need to be able to transform their enterprise characteristics into enterprise architecture. In doing so, a ***modelling language*** is needed to illustrate concepts that reflect parts of the real enterprise in a 'blueprint' of the enterprise, consisting of the interrelated concepts like *actors, processes, services, systems,* etc. Such a modelling language hands a base for ***applications*** that make manipulation possible, show views on the enterprise architecture for decision support and for *methods* to be applied to. These ***methods*** may incorporate steps for target situations, modelled using a language like ArchiMate. Methods, applications and languages in the field of EA are therefore interrelated; methods (like TOGAF) need a way of *representing* enterprise concepts (using a *language*) and the language itself is only useful when it is used (by *applications* and *methods*). So to state, modelling *languages* are used by *applications*, which may be used in steps of a *method*. We acknowledge the fact that enterprise architectures may be drawn by hand, but using applications makes it possible to easily modify an EA. In a way, drawing an EA by hand is a form of application, namely *'applying'* a language. However, the ease of using a language in *applications*, like Software AG Alfabet or BiZZdesign Architect, makes it possible to modify and show different views without having to put large amounts of time in it.

ArchiMate is an open standard, belonging to the Open Group consortium. ArchiMate is described as an open and independent modelling language for enterprise architecture. ArchiMate is one of several architecture-modelling languages, like ARIS, ebXML, etc. Most languages are proprietary and are based on or make us of concepts from other meta-modelling languages like UML (Jonkers, Lankhorst, Buuren, Hoppenbrouwers, & Bonsangue, 2004). As an illustration of the concepts that will be described in our research **we will use ArchiMate**, however, **other languages may also be suitable** to depict these concepts. The Open Group is a global consortium that enables the achievement of business objectives through IT standards and consists of more than 400 member organizations.

ArchiMate uses different concepts in its framework, first described in 2004 by Jonkers et al. (2004).

These concepts reflect organizational structures, like actors, processes, objects and others.

The concepts can be placed in a matrix with nine cells, consisting of two dimensions, *layers* and *aspects*. These layers and aspects are sub divided in respectively *business layer, application layer, technology layer* and *passive structure, behaviour structure* and active structure (The Open Group, 2013a), as depicted in Figure 24. This 'ArchiMate Framework' gives an overview of all the known concepts in ArchiMate and their characteristics.

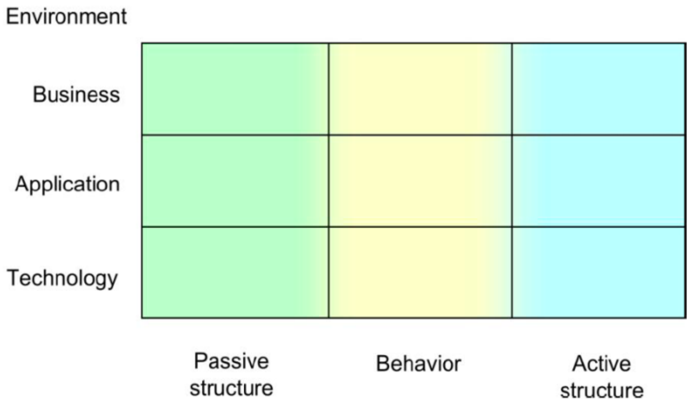


Figure 24. ArchiMate Framework (The Open Group, 2013a).

Layers reflect the part of the enterprise (or environment) a concept can be placed in, for example, some concepts in the business layer are actors, products and services.

> *"The **Business Layer** offers products and services to external customers, which are realized in the organization by business processes performed by business actors.*
>
> *The **Application Layer** supports the Business Layer with application services that are realized by (software) applications.*
>
> *The **Technology Layer** offers infrastructure services (e.g., processing, storage, and communication services) needed to support applications, realized by computer and communication hardware and system software."* (The Open Group, 2013a)

Aspects can be described as the 'type' or a 'structure' of a concept. They show what a concept is capable of doing, namely *performing* (active structure) some *activity* (behaviour) on an *object* (passive structure).

> *"An **active structure** element is defined as an entity that is capable of performing behaviour. [..]A **behaviour** element is defined as a unit of activity performed by one or more active structure elements. [..]A **passive structure** element is defined as an object on which behaviour is performed."* (The Open Group, 2013a)

Figure 25 shows how frequently used concepts in ArchiMate can be placed on the ArchiMate Framework depicted in Figure 24. Here, we see the layers shown in colours, however, in ArchiMate colours have no exact definition but are used for different clarifications regarding modelling. The active structure is shown in the two right 'columns', behaviour is shown in the middle two 'columns' and the passive structure is shown in the left two 'columns'. The colours represent from top to bottom respectively the business layer, application layer and the technology layer.

The figure can also be seen as a simple form of enterprise architecture. Enterprise architecture in the working field may be more complex, using different types of relations we have not discussed, and show less or more variety in concepts as we have depicted, but nevertheless enterprise architecture exists of interrelated concepts in its core.

ArchiMate also hands possibilities for extensions. In fact, it was extended with the Motivation Extension and the Implementation and Migration Extension. The Motivation Extension describes the way enterprise architecture can be aligned to its context, using additional concepts like *stakeholders, drivers* and *assessments*. The Implementation and Migration Extension adds concepts that may be used to model implementation programs and projects in order to support program, portfolio and project management. The extension also adds a 'plateau' concept to support migration planning (The Open Group, 2013a)

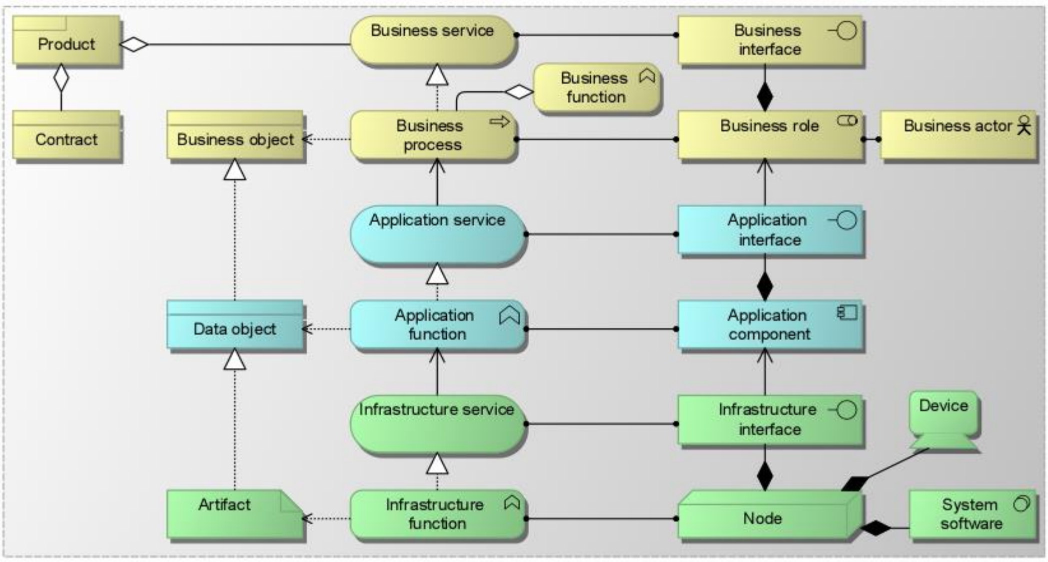


Figure 25. Frequently used concepts in ArchiMate 2.1 related to the ArchiMate Framework.

As we have briefly mentioned before, TOGAF is an Open Group *standard* that consists of several concepts related to *architecture*. TOGAF is an ***architecture framework***. Simply stated, *"TOGAF is a* *tool for assisting in the acceptance, production, use, and maintenance of architectures. It is based on an iterative process model supported by best practices and a re-usable set of existing architectural assets"* (The Open Group, 2011). ISO/IEC 42010:2007 defines ***'architecture'*** as:

> *"The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution"* (IEEE, 2007)

This definition is embraced by TOGAF; however, TOGAF also uses different definitions of architecture depending on the view that is taken. The core of TOGAF is in its Architecture Development Method (ADM), which defines a recommended sequence of phases and related activities that are involved when developing enterprise architecture (The Open Group, 2011). This ADM can be seen as the steps that are to be taken when starting from scratch.

Figure 26 shows the correspondence of ArchiMate with TOGAF's ADM. The core elements depicted in Figure 25 are shown as part of phase B, C and D in the ADM. Phases E, F and G correspond with the Implementation and Migration Extension of the ArchiMate language. Phase A and H, together with the phase 'Requirements Management' and the 'Preliminary Phase' correspond with the Motivation Extension. Altogether, ArchiMate (including its extensions that correspond with the design and evolution principles) corresponds with the IEEE definition of architecture, making it a suitable language for enterprise architecture.

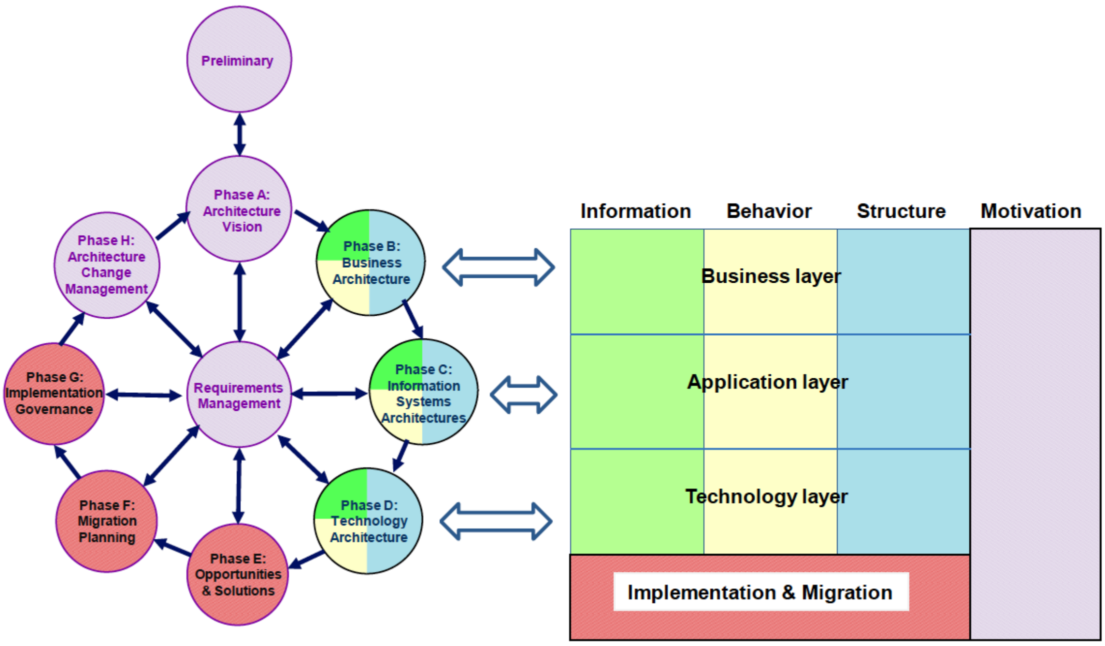


Figure 26. Correspondence between ArchiMate (right, including extensions) and TOGAF (left) (The Open Group, 2013).

3.8 Costs & Cost Terminology

Costs are a hot topic in organizations when times are rough. Cutting costs may e.g. help organizations surviving in heavy competition. Costs are used for ***accounting***, which provides both financial and non-financial information aiming to let decision-makers make the right decisions for their organizations (Drury, 2007).

> Accounting is defined by the American Accounting Association as *"the process of identifying, measuring and communicating economic information to permit informed judgements and decisions by users of the information"* (American Accounting Association. Committee to Prepare a Statement of Basic Accounting Theory., 1966).

Accounting may be split up into two different forms, namely financial accounting and management accounting. ***Financial accounting*** focuses on reporting to external parties, like banks, investors, governmental organizations, and suppliers. Business transactions are measured and recorded to provide financial statements based on 'generally accepted accounting principles' (GAAP). ***Management accounting*** measures, analyses, and reports financial and non-financial information to help managers make decisions in order to fulfil the goals of an organization (Horngren, Datar, & Rajan, 2012). Below, an overview of major differences between management and financial accounting is shown (cf. Figure 27).

	Management Accounting	Financial Accounting
Purpose of information	Help managers make decisions to fulfill an organization's goals	Communicate organization's financial position to investors, banks, regulators, and other outside parties
Primary users	Managers of the organization	External users such as investors, banks, regulators, and suppliers
Focus and emphasis	Future-oriented (budget for 2011 prepared in 2010)	Past-oriented (reports on 2010 performance prepared in 2011)
Rules of measurement and reporting	Internal measures and reports do not have to follow GAAP but are based on cost-benefit analysis	Financial statements must be prepared in accordance with GAAP and be certified by external, independent auditors
Time span and type of reports	Varies from hourly information to 15 to 20 years, with financial and nonfinancial reports on products, departments, territories, and strategies	Annual and quarterly financial reports, primarily on the company as a whole
Behavioral implications	Designed to influence the behavior of managers and other employees	Primarily reports economic events but also influences behavior because manager's compensation is often based on reported financial results

Figure 27. Major differences between management and financial accounting (Horngren et al., 2012).

Cost accounting concerns the cost accumulation for inventory valuation to meet the requirements of external reporting and internal profit measurement, whereas ***management accounting*** relates to the provision of appropriate information for decision-making, planning, control and performance evaluation. The distinction between cost accounting and management accounting is extremely vague regarding the topic of decision-making and the two terms are often used interchangeably (Drury, 2007), both meaning the same subject. In this thesis, we will refer to the term 'management accounting', where we include the meaning of cost accounting as well.

Costs are measured for *profit measurement and inventory valuation, decision-making, performance measurement* and *control* (Drury, 2007). In good times, organizations tend to focus on selling as much as they can, while in difficult times costs are high on the corporate agenda. The term 'costs', however, is rather ambiguous. Is it the price that is paid for something that has a value; is it a cash outflow; perhaps a financial term to describe an event that affects corporate profitability? Many different types of costs exist, and organizations shift their focus on them at different times (Horngren et al., 2012).

> Accountants describe a ***cost*** as *"a resource sacrificed or forgone to achieve a specific objective"* (Horngren et al., 2012). It often reflects a monetary measure (Drury, 2007).

All businesses are concerned about *revenues* and *costs*(Horngren et al., 2012), since they need to make profit. Costs may have occurred in time, referred to as ***actual costs*** or may be predicted or forecasted to occur, referred to as ***budgeted cost***. Costs refer to things that reflect a value; anything for which a measurement of costs might be desired. Such a 'thing' is called a ***cost object***. Examples of cost objects are *products, services, projects* and *activities* (Horngren et al., 2012). Organizations may have different forms of resources, like products and services that represent value for the company. Appointing a monetary measure to such a resource and sacrificing it to achieve a specific objective will create costs. Logging these monetary measures in any form will create ***cost data***. When this data is being interpreted, the cost data will become ***cost information*** (cf. Figure 8 and Figure 9).

The term 'cost' often goes with an adjective to describe the type of cost for a specific context (Drury, 2007). Costs may be classified using such adjectives. As explained, costs refer to cost objects. Referring to cost objects requires ***cost assignment***, which is the appointing of costs to cost objects. Cost assignment can be divided into ***cost tracing*** and ***cost allocation***, explained later.

Different classifications or *cost types* exist, like ***direct costs*** and ***indirect costs***. Direct costs of a certain cost object relate to the particular cost object and can be *traced* to it in an 'economical feasible (cost-effective) way' (Horngren et al., 2012). For example, the cost of steel for a car (cost object) can be traced to the specific vehicle, likewise labour costs. These costs are called direct costs. Indirect costs of a certain cost object relate to the cost object, but cannot be traced to it. For example, the people who manage the planning for the building of a vehicle do not directly spend their time on building the vehicle. Therefore, it cannot be directly traced to the cost object, the vehicle. Their salaries are indirect costs for the vehicle. Assigning indirect costs to cost objects, like a vehicle, is called ***cost allocation***. Assigning direct costs to cost objects is called ***cost tracing*** (Horngren et al., 2012).

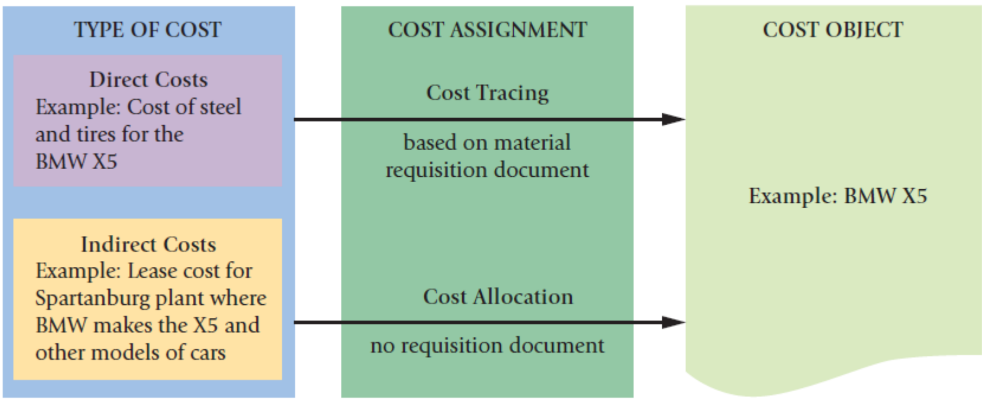


Figure 28. Cost assignment to a cost object (Horngren et al., 2012).

As explained, costs are important for decision-making. Many different types of costs may be identified, depending on the context referred to. Horngren et al. (2012) provided an overview of cost classifications (cf. Figure 29). These cost classifications are examples that may serve decision-making and evaluation performance.

1. Business function
   a. Research and development
   b. Design of products and processes
   c. Production
   d. Marketing
   e. Distribution
   f. Customer service
2. Assignment to a cost object
   a. Direct cost
   b. Indirect cost
3. Behavior pattern in relation to the level of activity or volume
   a. Variable cost
   b. Fixed cost
4. Aggregate or average
   a. Total cost
   b. Unit cost
5. Assets or expenses
   a. Inventoriable cost
   b Period cost

Figure 29. Examples of cost classifications (Horngren et al., 2012).

Next to these classifications, other classifications may be identified, like *prime costs*, *semi-variable costs*, *relevant costs*, *avoidable and unavoidable costs*, *opportunity costs*, *incremental costs*, and *marginal costs* (Drury, 2007). All of these classifications identify the cost for a specific context. In an organization, costs are recorded in a cost and management accounting system, in order to submit e.g. to tax regulation. For accounting, it is important to say something about 'where the cost comes from' or 'where the cost belongs to'. Sometimes costs can directly be traced (direct costs), but sometimes this is done by allocating them to cost objects (indirect costs). According to Drury (2007), *"a cost and management accounting system should generate information to meet the following requirements".* According to Drury (2007), three ways of constructing accounting information exist.

1. To allocate costs between cost of goods sold and inventories for internal and external profit reporting and inventory valuation;
2. To provide relevant information to help managers make better decisions;
3. To provide information for planning, control and performance measurement.

First of all, ***cost accounting*** is used for cost allocation to products, between costs of goods sold and inventories, in order to meet the requirements for external and internal financial accounting inventory valuation and profit measurement. Secondly, *relevant* costs are used for ***decision-making*** by providing information in such a way that it is understandable for managers and that decisions can be fairly made. Thirdly, ***responsibility accounting and performance management*** are used for the process of *"translating goals and objectives into specific activities and the resources that are required, via the short-term (budgeting) and the long-term (planning) processes, to achieve the goals and objectives"*.

For each of these three requirements, costs are presented in a different manner. For our research, we focus on the first two requirements , and explain how enterprise architecture could play a role in later chapters (cf. Figure 30).

Cost allocation (Chapter 3.10)	Decision-making (Chapter 3.11)	Planning, control & performance management (not in this thesis)

Figure 30. Coverage of management accounting topics in this thesis (grey indicates coverage).

According to Drury (2007), there is a need for a cost accumulation system for generating relevant costs for *decision-making*. The three main reasons he mentions are:

1. “Many indirect costs are relevant for decision-making”;
2. “An attention-directing information system is required that periodically identifies those potentially unprofitable products that require more detailed special studies”;
3. “Product decisions are not independent”.

Such a ‘system’ is explained in Chapter 3.10 and explains the topic of ‘cost allocation’. The above stated reasons also explain the direct link between the topic of ‘cost allocation’ and ‘decision-making’, namely that cost allocation serves decision-making.

In the following chapters, the topic ‘cost allocation’ is explained and its relation to decision-making (cf. Chapter 3.10 *Cost Allocation & Methods*). In the subsequent chapter, the topic ‘decision-making’ is discussed with respect to cost information (cf. Chapter 3.11 *Relevant Cost Information for* Decision-Making).

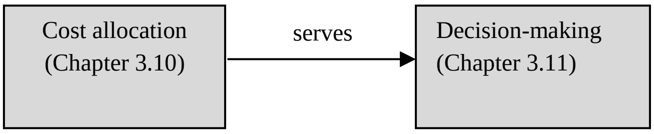


Figure 31. Relation between cost allocation and decision-making.

In the next chapter, we will give an overview of several cost analysis types and methods that may be used for either *cost allocation* or *decision-making.* Due to the focus of this thesis, we will not cover the topic of *planning, control & performance management.*

3.9 Cost Analysis Types & Methods

Many types of cost analysis exist. Often, cost analysis includes *revenues* as a variable. Mutschler (2008) has given an overview of existing *evaluation approaches*. These approaches discuss topics like *"costs of an investment, assumed profit, its impact on work performance, business profit performance* and *the achievement of enterprise objectives"* (Mutschler, 2008). Regarding information technology evaluation, four main cost-oriented approaches may be used, namely *zero base budgeting, cost-effectiveness analysis, target costing* and *total cost of ownership.* Surely, many more cost analysis types exist that include revenues, e.g. for calculating profit. Static measures like *return on investment (ROI), payback period (PP), accounting rate of return (ARR)* and *break-even analysis* exist that involve revenues as well. Dynamic measures like *net present value (NPV)* and *internal rate of return (IRR)* are mentioned (Mutschler, 2008). Next to IT evaluation, these methods may generate information that serves as input for decision-making.

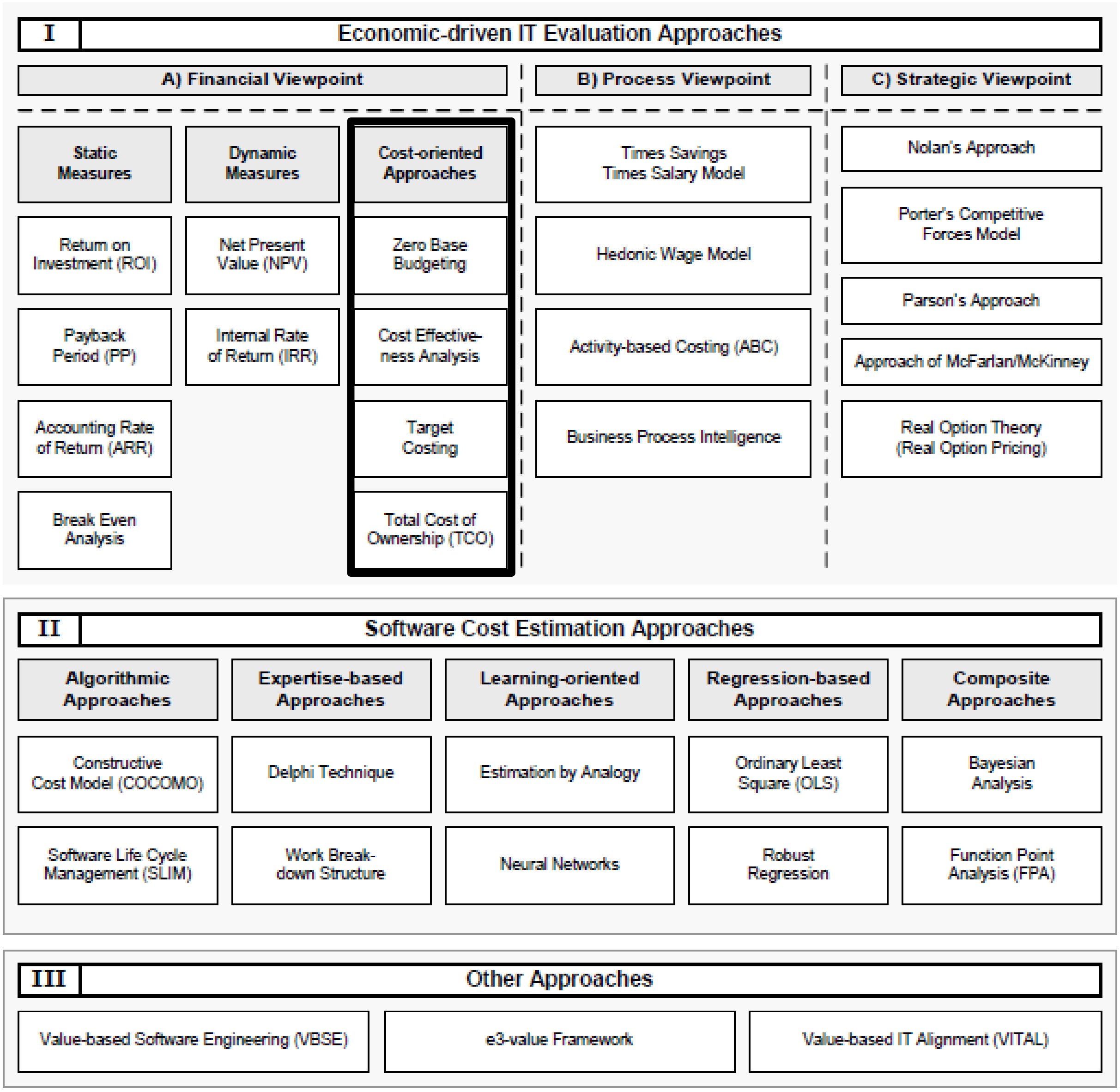


Figure 32. Overview of economic driven IT evaluation, software cost estimation and other approaches (Mutschler, 2008).

Next to the overview shown in Figure 32, several other types of cost methods exist. At this point, we need to differentiate between two topics, namely *cost accumulation for inventory valuation* and *information for decision-making*. The first topic relates to *cost assignment*, the second to *cost analysis*. Surely, cost allocation provides new information that may serve as input for decision-making as well. As explained by Drury (2007), these topics are inter-related and no clear distinction can be made as far as the topic *'decision-making'* concerned. However, we have previously explained that cost allocation information serves decision-making. The topic of *planning, control & performance management* may also serve decision-making, but is not included in this thesis.

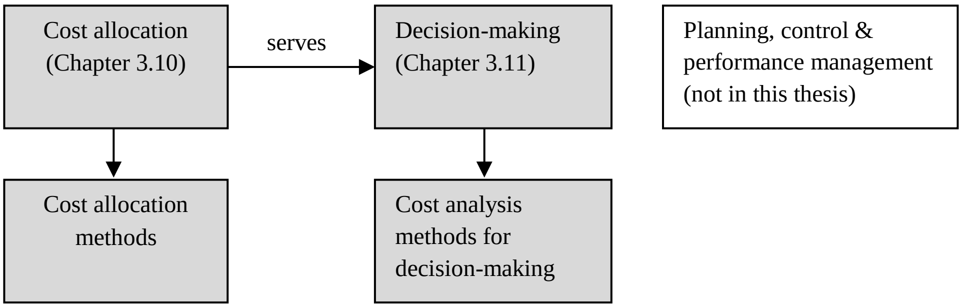


Figure 33. Overview of topics in relation to cost allocation methods and cost analysis methods for decision-making.

3.10 Cost Allocation & Methods

As explained in the previous chapter, measuring costs may serve several purposes. In this chapter, we explain how cost allocation is done. In the previous chapter, we have stated that assigning direct costs to cost objects is called ‘cost tracing’. Assigning indirect costs to cost objects is called a ‘cost allocation’.

> A cost allocation is *“the process of assigning costs when a direct measure does not exist for the quantity of resources consumed by a particular cost object”* (Drury, 2007).

Ideally, costs are linked with the ‘thing’ that caused it. This thing may be a process, a department, products, services, or any other organizational subject that is able to ‘create costs’ (Horngren, Sundem, Stratton, Teall, & Gekas, 2006). However, often costs are not directly linked to the cost object (also known as ‘cost objectives’). According to Horngren et al. (2006), costs are allocated for three main reasons, namely:

1. To obtain desired motivation;
   Using cost allocations may, for example, give employees (e.g. managers) the reasons to carefully use resources and thereby save costs.
2. To compute income and asset valuations;
   When costs are allocated, these costs may be used to measure inventory and cost of goods sold. This may serve *financial accounting* purposes, but also *planning, performance evaluation*, and *control.*
3. To justify costs and obtain reimbursements.
   Sometimes it is difficult to explain why a certain cost is what it is. Producing products may involve many costs that need to be made in order to make it possible to create them. Performing a cost allocation may justify these costs.

Horngren et al. (2006) names three types of cost allocations, namely: (1) *allocation of joint costs to the appropriate responsibility centres*, (2) *reallocation of costs from one responsibility centre to another* and (3) *allocation of costs of a particular organizational unit to its outputs of products or services*. The types are fundamentally the same in what they establish, but differ in the *type* of cost to be allocated, *how* the establishment is done and *where* the costs should be allocated to.

Two types of *systems* to use for assigning indirect costs to cost objects exist, namely ***traditional costing systems*** and ***activity-based costing (ABC) systems***. Next to these’ indirect costing systems’, used for cost assignment, ***direct costing systems*** exist. Direct costing systems do not address the assignment of indirect costs to cost objects, but instead only assign direct costs to cost objects, which is a disadvantage if a large proportion of an organization’s total costs are indirect costs (Drury, 2007). Analysis results would then reflect an unrealistic situation of the organization, making the data presented unsuitable for decision-making. This is why in this thesis we do not cover the topic of direct costing systems (cf. Figure 34). Direct costing

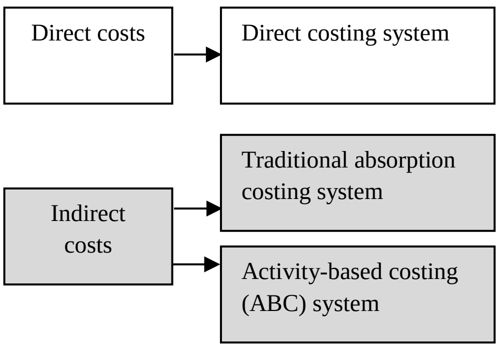


Figure 34. Choosing a costing system (grey indicates coverage in this thesis).

systems are only recommended if indirect costs make up a small percentage of an organization's total costs (Drury, 2007).

Traditional costing systems are used since the early 1900s and are even still used today. They rely extensively on arbitrary cost allocations. ABC systems emerged in the late 1980s; they are focused on cause-and effect cost allocations (Drury, 2007). Both systems identically assign direct costs to cost objects. Assigning direct costs to cost object may be done, e.g. using *time sheets* or *job cards*. For direct labour, time spent on creating a product is then directly written down and directly inserted into either a traditional costing system or an ABC system. For direct materials, the source document is a *materials requisition* (Drury, 2007). Details of the materials used for the manufacturing of a product are then recorded on the materials requisition.

A simple way to assign indirect costs to cost objects in traditional costing systems is done by using a single overhead rate for the entire organization. Indirect costs are also called *overheads*; the terms 'blanket overhead rate' or 'plant-wide rate' are used when referring to the single rate used for the entire organization.

A 'two-stage allocation process' could also be used to *assign indirect* costs to cost objects; it may be applied to both traditional as well as ABC systems. Traditional absorption costing systems differ from activity-based costing systems in the following ways. Both use the ***'two-stage allocation process'***, but in a different way. A traditional absorption costing system *"allocates overheads to production and service departments in the first stage, and reallocates service department costs to the production departments in the second stage"*, whereas ABC-systems *"assigns overheads to each major activity (rather than departments)"*.

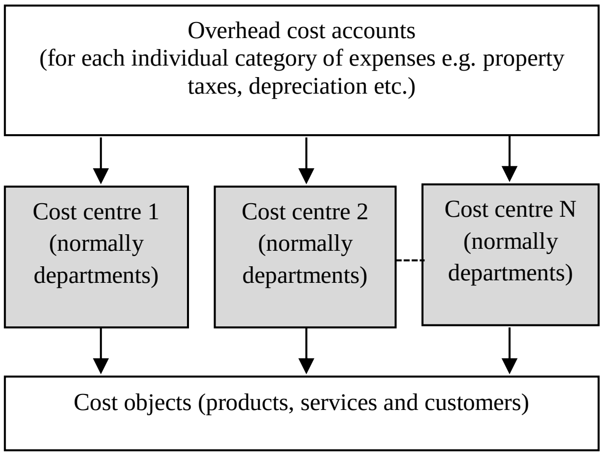


Figure 35. Two-stage allocation process for traditional costing systems (Drury, 2007).

In the first stage, traditional absorption costing systems tend to use departments for pooling costs (cost centres) and ABC-systems use activity-based cost centres (activity cost pools) (Drury, 2007). A ***cost pool*** is *"a group of individual costs that is allocated to cost objectives using a single cost driver"* (Horngren et al., 2006).

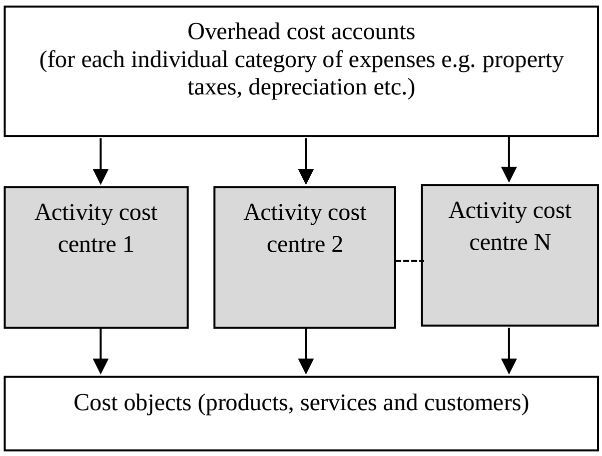


Figure 36. Two-stage allocation process for activity-based costing systems (Drury, 2007).

Activities are aggregations of many different tasks, like 'schedule production', 'move materials', 'inspect items', etc. Some activities in ABC-systems may be identical to traditional costing systems' cost centres, like 'purchasing' departments. In general, activity-based costing systems will have a larger number of cost systems (Drury, 2007).

In the second stage of the two-stage allocation process, costs from cost centres are allocated to products or other chosen cost objects. Traditional costing systems *"trace overheads to products using a small number of second stage* ***allocation bases*** *(normally described as overhead allocation rates)"* (Drury, 2007), like direct labour and machine hours. ABC systems use the term ***'cost driver'*** instead of the term 'allocation base'. Unlike traditional costing systems, ABC systems use a variety of different cost drivers to allocate costs in the second stage, like 'the number of purchase orders for the purchasing activity'.

The major differences between traditional costing systems and ABC systems are that within the two-stage allocation process, ABC systems heavily rely on *a greater number of cost centres* and *a greater number and variety of second stage cost drivers* (Drury, 2007).

ABC systems use both ***volume-based cost drivers*** and ***non-volume-based cost drivers***, whereas traditional costing systems only use volume-based cost drivers (Drury, 2007). Using volume-based cost drivers assumes a product's consumption of overhead resources is directly related to units produced. This means that the assumption exists that the overhead consumed by products highly correlates with the number of units produced. For traditional costing systems, typical volume-based cost drivers are *units of output, direct labour hours* and *machine hours*. For costs that do not increase or decrease with volume of products produced, *non-volume-based cost drivers* may be used for cost allocation. For example, in some cases a *machine set-up* is more related to 'set-up costs' than the products produced on the machine. Here, the cost driver should be the *number of machine setups* rather than the *amount of products produced*. We might say, ABC systems recognize that overheads are caused by other factors next to *volume*. Sometimes it's good to use ***consumption ratios*** as a cost driver. These reflect *"the proportion of each activity consumed by a product"* (Drury, 2007).

Table 4 shows an example of a cost allocation of 1 million overheads using direct labour hours as the allocation base for the traditional system and number of batches processed as the cost driver for the ABC system. The example was derived from the book of Drury (2007). In this example, the variable 'overheads allocated' are the amounts of currency calculated based on the consumption ratios related to the products. The ratios are shown as percentages.

Table 4. Example of a cost allocation (Drury, 2007).

	Traditional costing system		ABC system	
	Product 'high volume' (currency)	Product 'low volume' (currency)	Product 'high volume' (currency)	Product 'low volume' (currency)
Direct costs	310 000	40 000	310 000	40 000
Overheads allocated	300 000 (30%)	50 000 (5%)	150 000 (15%)	150 000 (15%)
Reported profits/(losses)	(10 000)	60 000	140 000	(40 000)
Sales revenues	600 000	150 000	600 000	600 000

In the example presented in Table 4, the traditional costing system represents misleading information. Namely, a small loss is reported for product 'high volume'. The information presented gives the impression that if product 'high volume' would be discontinued, the 'overheads allocated' would drop by 300 000 in currency. What may be concluded here is that the decision-maker should focus on the more profitable product 'low volume'. In reality, however, such a decision might be disastrous, since low volume products are made in small batches and require more labour hours and other overhead costs. The effect of deciding wrongly would be peaking overhead costs, which is the last thing the organization wants.

As shown in the previous example, ABC systems are more accurate in providing information for decision-making. According to Drury (2007), designing an ABC system involves four steps:

1. Identifying the major activities that take place in an organization;
2. Assigning costs to cost pools/cost centres for each activity;
3. Determining the cost driver for each major activity;
4. Assigning the cost of activities to products according to the product's demand for activities.

Here, the first two steps relate to the first stage in the 'two-stage allocation process', whereas the final two steps relate to the second stage. Identifying the major activities taking place in an organization is a matter of judgement and is likely to be influenced by factors like the total cost of the activity centre; it should be significant enough to justify a separate treatment. Activities having the same cost driver can be put together in a single activity cost centre. Activities with the same product consumption ratios could use the same cost driver to assign costs to products.

Assigning costs to activity cost centres involves the determination of the amount the organization is spending on each of its activities. Some resources are directly attributable to specific activity centres, but others (e.g. heating costs) are indirect or even jointly used by multiple activities. These costs should be assigned to activities on the basis of cause-and effect cost drivers (Drury, 2007). If allocations are arbitrary, they should not be used. Also, if costs are unaffected by a decision e.g. to stop developing a product, they should not be assigned to products.

Selecting appropriate cost drivers for assigning the cost of activities to cost objects involves selecting a cost driver for each activity centre. Here, cost drivers are called *activity cost drivers*. To select a suitable cost driver, a few factors should be born in mind. First of all, the cost driver should provide a good explanation of costs in each activity cost pool, second it should be easily measureable and the data should be fairly easy to retrieve and to be identified with the products. Costs of the measurement need to be taken into account; if costs are too high, it might not be feasible to select this activity cost driver. Instead, another driver may suffice. *Transaction drivers* are all drivers that measure the number of times an activity is performed. These drivers are the least expensive to determine, but may be inaccurate since they assume that the same resources are used each time an activity is performed. If the conditions do not change considerably, a transaction driver may be feasible to determine. *Duration drivers* are drivers that measure the amount of time required to perform an activity, e.g. *set-up hours* or *inspection hours.* Assigning the cost of the activities to products involves the application of the cost driver rates to products. For this reason, the cost driver must be measureable. For example, if 'utilization rate based on processor usage' is a cost driver, the rate should be easily obtained, e.g. from a server process that keeps track of which activity uses the processor in a computer. Two types of allocation methods exist, namely *methods for reallocating inter-service department costs* and *methods of allocating joint costs*. The latter is, as we will explain in this chapter, not relevant for decision-making.

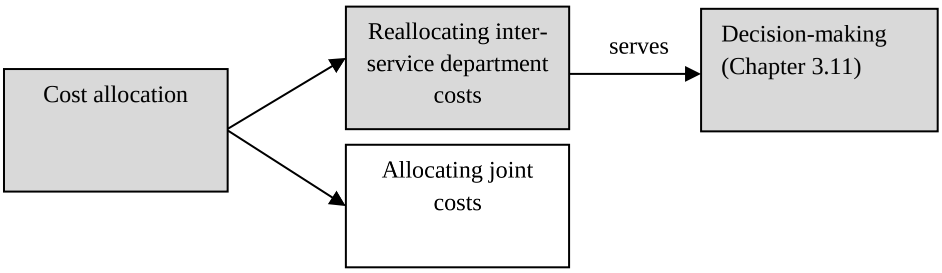


Figure 37. Relevance of allocation methods for decision-making.

Methods of Reallocating Inter-Service Department Costs

In this part, four methods of allocating service department costs will be explained. We refer to the examples as illustrated in Drury (2007), Chapter 3. Table 5 and Table 6 represent the situation of a certain production company. As we can see in Table 5, the production company has several departments, split up into *production* and *service* departments. Production costs may be directly traced to the products produced by each department, but the costs incurred in the service department need to be reallocated, since the services they produce are consumed by the production departments.

Table 5. Totals of overheads analysed to production and service departments.

		Currency
Production department	X	48 000
	Y	42 000
	Z	30 000
Service department	1	14 040
	2	18 000
		152 040

Table 6 represents the apportioning of expenses of service departments to the production departments. For each production department, a percentage of both service department 1 and service department 2 is given. This percentage refers to the portion of service consumed by the production departments. The service department do not consume their own services produced, which is why they have no share percentage in the apportioning of service costs.

Table 6. Expenses of service departments apportioning.

	Production departments			Service departments	
	X	Y	Z	1	2
Service department 1	20%	40%	30%	-	10%
Service department 2	40%	20%	20%	20%	-

Four different methods of allocating service department costs exist. Each of the methods have a different purpose and are suitable for different situation. These methods are:

1. Repeated distribution method
2. Simultaneous equation method
3. Specified order of closing method
4. Direct allocation method

For convenience, we explain the methods based on the book of Drury (2007). The examples illustrated in the book were reused here as well.

Repeated distribution method

The *repeated distribution method* repeatedly allocates department costs in the specified percentages *"until the figures become too small to be significant"* (Drury, 2007). For each service department percentages are determined to allocate costs from other departments. Multiplying percentages with costs makes up one iteration. Iterating is done as long as costs are still significant. Outcomes of iterations are summed to show the total allocated costs per department (cf. Drury (2007), Chapter 3). Table 7 shows the example as explained. Calculation of allocation costs is done per line; e.g. in line 2, 20 percent of the costs of service department 1 is allocated. 20 percent multiplied by 14 040 makes up 2 808. For calculating the costs per line for each service department, percentages of the previous line from the other service department are used to calculate the costs for the respective service line. For example, in line 3, the costs of service department 1 are 20 percent of the costs shown in service department 2. 20 percent of 19 404 is 3 881.

Table 7. Example of a 'repeated distribution method' application (Drury, 2007).

		Production departments			Service departments		
	Line	X	Y	Z	1	2	Total
1	Allocation as per overhead analysis	48 000	42 000	30 000	14 040	18 000	152 040
2	Allocation of service department 1	2 808 (20%)	5 616 (40%)	4 212 (30%)	(14 040)	1 404 (10%) ---------------- 19 404	
3	Allocation of service department 2	7 762 (40%)	3 881 (20%)	3 880 (20%)	3 881 (20%)	(19 404)	
4	Allocation of service department 1	776 (20%)	1 552 (40%)	1 165 (30%)	(3 881)	388 (10%)	
5	Allocation of service department 2	154 (40%)	78 (20%)	78 (20%)	78 (20%)	(388)	
6	Allocation of service department 1	16 (20%)	31 (40%)	23 (30%)	(78)		
7	Allocation of service department 2	4 (40%)	2 (20%)	2 (20%)	-		
8	Total overheads	59 520	53 160	39 360			152 040

Simultaneous equation method

In the *simultaneous equation method*, equations are initially stated and solved. An example of such equations are: $x = 14040 + 0.2y$ and $y = 18000 + 0.1x$, with x = total overhead of service department 1 and y = total overhead of service department 2. The variable x is resolved to 18000 and y is resolved to 19800. Then, these overhead costs are e.g. allocated to the production departments, like in the repeated distribution method (cf. Drury (2007), Chapter 3). The equations are built up as follows: a fixed part that represents the costs of the respective service department, as shown in Table 5, and a variable part that represents the service department percentage, as shown in Table 6.

Table 8. Example of a ‘simultaneous equation method’ application (Drury, 2007).

	Line	X	Y	Z	Total
1	Allocation as per overhead analysis	48 000	42 000	30 000	120 000
2	Allocation of service department 1	3 600 (20%)	7 200 (40%)	5 400 (30%)	16 200
3	Allocation of service department 2	7 920 (40%)	3 960 (20%)	3 960 (20%)	15 840
4	Total overheads	59 520	53 160	39 360	152 040

Specified order of closing method

In the *specified order of closing method* (also referred to as the ‘sequential method’), overhead costs are allocated to e.g. production departments in a certain order. Service departments that do the largest amount of work for other service departments are ‘closed’ first, the service department with the second largest portion of work for other departments are ‘closed’ second, and so on. This method differs from the repeated distribution and simultaneous equation methods in accuracy for clerical convenience. The totals allocated in this method differ strongly from the totals in the repeated distribution method and the simultaneous equation method. There may be strong reasons to use this method for clerical convenience if a close approximation to an alternative accurate calculation is reached (cf. Drury (2007), Chapter 3).

Table 9. Example of a ‘specified order of closing method’ application (Drury, 2007).

		Production departments			Service departments		
	Line	X	Y	Z	1	2	Total
1	Allocation as per overhead analysis	48 000	42 000	30 000	14 040	18 000	152 040
2	Allocate service department 2	7 200 (40%)	3 600 (20%)	3 600 (20%)	3 600 (20%)	(18 000)	
3	Allocate service department 1	3 920 (2/9)	7 840 (4/9)	5 880 (3/9)	(17 640)	-	
4		59 120	53 440	39 480	-	-	152 040

Direct allocation method

In the *direct allocation method* service department costs are reallocated only to production departments, ignoring inter-service department service reallocations (Drury, 2007). The method is used when inter-service reallocations are rather insignificant. In these cases the method may bring simplicity in allocating costs (cf. Drury (2007), Chapter 3).

Table 10. Example of a ‘direct allocation method’ application (Drury, 2007).

		Production departments			Service departments		
	Line	X	Y	Z	1	2	Total
1	Allocation as per overhead analysis	48 000	42 000	30 000	14 040	18 000	152 040
2	Allocate service department 2	3 120 (2/9)	6 240 (4/9)	4 680 (3/9)	(14 040)	-	
3	Allocate service department 1	9 000 (4/8)	4 500 (2/8)	4 500 (2/8)	-	(18 000)	
4		60 120	52 740	39 180	-	-	152 040

Summary – Methods of Reallocating Inter-Service Department Costs

The principle of cost allocation has been explained along with reasons why cost allocations may be in favour to be performed. Three types of cost allocation exist, namely: (1) *allocation of joint costs to the appropriate responsibility centres*, (2) *reallocation of costs from one responsibility centre to another* and (3) *allocation of costs of a particular organizational unit to its outputs of products or services*. Two basic *systems* have been explained, namely traditional costing systems and activity-based costing systems. The two-stage allocation process was explained, which is used in both traditional as well as activity-based costing systems. The differences between traditional costing and activity-based costing (ABC) systems are that ABC systems assign costs to activity cost centres and traditional costing systems assign costs to departments in the first stage of the two-stage allocation process. Moreover, ABC systems use a greater number and variety of second-stage allocation bases. Traditional costing systems often use arbitrary allocations, whereas ABC systems often use cause-and-effect allocation bases. ABC systems are in favour nowadays for organizations (Horngren et al., 2006). The mentioned allocation methods are used in different situations and depend on the type of organization.

Methods of Allocating Joint Costs

Thus far we have discussed cost allocations for products and services. For *joint* and *by-products*, different allocation methods exist. A ***joint product*** is realized *"when a group of individual products is simultaneously produced, and each of these products has a significant relative sales value"* (Drury, 2007). The outputs are then referred to as 'joint products'. In a single process where multiple products are produced, there may be two products created intentionally, called joint products, but other products that have a minor sales value when compared to the joint products are called ***by-products***. For such a production process, labour and overhead (indirect services) may be the input, together with raw materials. The process itself is called a ***joint process***. Once the joint process is done, both the joint products as well as the by-products split off at some time. This point is called the ***split-off point***. The production process for joint and by-products, as described by (Drury, 2007), is shown in Figure 38.

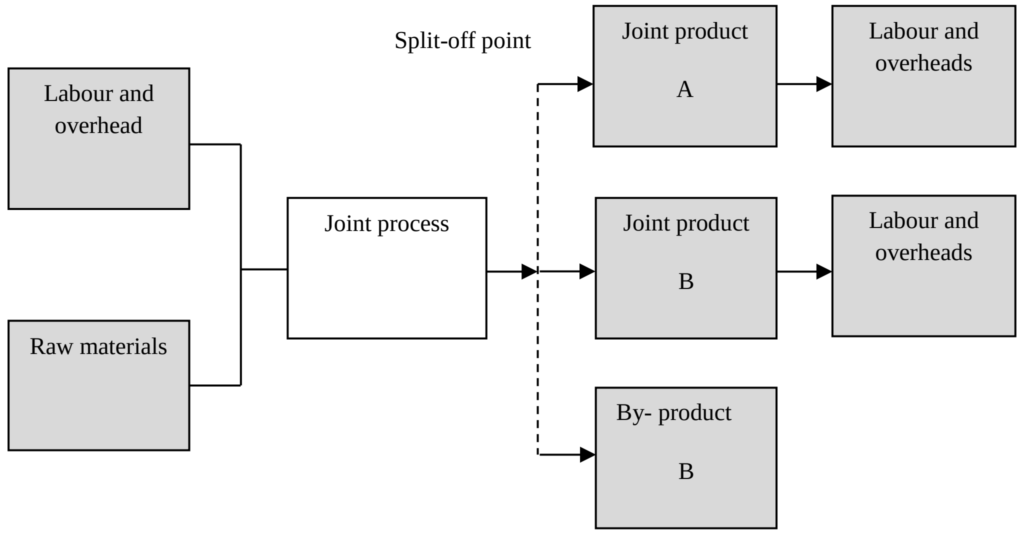


Figure 38. Production process for joint and by-products (Drury, 2007).

Allocating joint costs is mostly done in two ways, using:

1. Methods based on physical measures (such as weight, volume, etc.)
2. Methods assumed to measure the ability to absorb joint costs, based on allocating joint costs relative to the market values of the products.

Drury (2007) describes four methods that may be used for allocating joint costs. The methods may be used in different situation under different conditions. It is up to the person responsible for allocating the joint costs to determine which method should be used. The methods are:

1. Physical measures method
2. Sales value at split-off method
3. Net realizable value method
4. Constant gross profit percentage method

Physical measures method

The *physical measures method* uses a simple cost allocation of joint costs in correspondence with volume. Here, the rate is calculated for each product based on the percentage the respective product is part of the total amount of products (or units) produced.

Table 11. Example of 'physical measures method' application.

Product	Amount of products	Proportion to total	Joint costs allocated (currency)	Cost per product (currency)
X	20 000	2/10	100 000	5
Y	30 000	3/10	150 000	5
Z	50 000	5/10	250 000	5
Total	100 000		500 000	

Sales value at split-off point method

The *sales value at split-off point method* is used when the assumption can be made that when the products are sold, the allocated costs should also be higher. In a way, this method has more to do with apportioning profits and losses rather than allocating costs. The sales value at split-off point method allocates joint costs *"based on their ability to absorb joint costs"* (Drury, 2007). However, it may be criticized due to its assumption that prior costs are determined by sales revenue. For example, if a product that is unprofitable and has a low sales revenue is allocated with a small share of joint costs, this may give the impression it is profitable, while in fact is not.

Table 12. Example of a 'sales value at split-off point method' application.

Product	Amount of products	Sales value (currency)	Percentage of sales value to total (%)	Joint costs allocated (currency)
X	20 000	400 000	40	200 000
Y	30 000	300 000	30	150 000
Z	50 000	300 000	30	150 000
Total	100 000	1000 000		500 000

Net realizable value method

The *net realizable value method* is used to estimate the sales value at the split-off point. Since it is likely that joint products will be processed individually after they were realized after the split-off point, an estimation of what their values are may still be lacking. The net realizable value at split-off point may be calculated by deducting the further processing costs at the point of sale from the revenues. In Table 13, an example of an application of the method is presented. Here, the costs beyond split-off point are given, together with the sales value. The estimated net realizable value at split-off point is calculated by deducting the costs beyond split-off point from the sales value. The proportion to total of each product's estimated net realizable value at split-off point is calculated by calculating the ratio for each product with respect to the total. The joint costs allocated are calculated by deducting the costs beyond split-off point from the estimated net realizable value at split-off point. Profit is calculated by deducting the estimated net realizable value at split-off point from the sales value. Calculating the profit for each product is done by multiplying the proportion to total with the total profit. Gross profit is calculated by dividing the profits by the sales values for each product.

Table 13. Example of a 'net realizable value method' application.

Product	Sales value (currency)	Costs beyond split-off point (currency)	Estimated net realizable value at split-off point (currency)	Proportion to total (%)	Joint costs allocated (currency)	Profit (currency)	Gross profit (%)
X	400 000	70 000	330 000	41.25	247 500	82 500	20,06
Y	300 000	60 000	240 000	30.00	180 000	60 000	20,00
Z	300 000	70 000	230 000	28,75	172 500	57 500	19,17
Total	1000 000	200 000	800 000		600 000	200 000	20,00

Constant gross profit percentage method

The *constant gross profit percentage method* is used when the assumption is made that 'since the products are all from the same productive process, they should all earn the same percentages'. In Table 13, the gross profit percentages are different. This method *"allocates joint costs so that the overall gross profit percentage is identical for each individual product"* (Drury, 2007).

Table 14. Example of a 'constant gross profit percentage method' application.

	Product X (currency)	Product Y (currency	Product Z (currency)	Total (currency)
Sales value	400 000	300 000	300 000	1000 000
Gross profit (20%)	80 000	60 000	60 000	200 000
Cost of goods sold	320 000	240 000	240 000	800 000
Less separable further processing costs	70 000	60 000	70 000	200 000
Allocated joint costs	250 000	180 000	170 000	600 000

Summary – Methods of Allocating Joint Costs

Four methods of allocating joint costs have been discussed, namely the *physical measures method, sales value at split-off method, net realizable value method* and the *constant gross profit percentage method*. Depending on the organization's situation, a method should be selected to allocate the joint costs. For decision-making, however, joint product costs should *not* be used due to their irrelevance for future situations; only costs that change as a possible result from a decision are classed as relevant.

General Summary

Regarding cost allocations, several *cost allocation types*, *costing systems* and *cost allocation methods* exist. The two-stage allocation process may be used for both traditional costing systems as well as ABC systems. Nowadays, ABC systems are often used due to their accuracy of allocating costs. For decision-making, allocations for joint costs are irrelevant and should not be used when presenting information (Drury, 2007). Depending on the organization, a recommendation is made to use an ABC system, select the appropriate method depending on the situation the organization is in and perform the allocation for the organization, which may be used for several purposes like *obtaining the desired motivation, computing income and asset valuations* or to *justify costs and obtain reimbursements*.

3.11 Relevant Cost Information for Decision-Making

Providing cost information may be interesting for decision-making for several reasons. We have explained the role of costs for internal and external reporting, i.e. creating overviews of the costs e.g. for departments, services and products that may be used to justify costs on formal reports like balance sheets and income statements. These reports are ***recurrent*** and show the financial state of an organization. For a balance sheet this state represents the value on a certain moment in time, for an income statement the state represents the value (both costs and revenues) over a period of time. ***Non-recurrent*** situations, which are situations where organizations need to be provided with information that is tailored specifically to the needs, may occur at all times. For example, if an organization is faced with heavy competition, the organization may want to follow a different strategy to overcome the competition. These non-recurrent decisions are also called '***special studies***' (Drury, 2007). Examples of common special studies are: *special selling price decisions, product-mix decisions when capacity constraints exist, decisions on replacement of equipment, outsourcing (make or buy) decisions* and *discontinuation decisions* (Drury, 2007).

> For decision-making only *relevant costs* should be used. These represent the incremental costs relating to a decision. Costs that will be unaffected while making decisions are therefore irrelevant (Drury, 2007).

Relevant costs are *expected future* costs, according to Horngren et al. (2012). According to them, determining the costs that are relevant for decision-making should be done based on the following criteria:

1. *They should occur in the future*; decisions deal with future situations. Costs that do not influence these situations are therefore irrelevant.
2. *Differ among the alternative courses of action*; costs that may happen in the future, but are constant for all future situations do not matter for decision-making, since whatever choice is made, the costs will be the same anyway.

For organizations it is important to have ***quantitative information*** for decision-making, e.g. for making estimations and calculations to compare whether a decision will turn out well. Thus far, we have focused only on quantitative information. Next to quantitative information, ***qualitative information*** is important for decision-making. Though difficult to measure, like the risk of something happening in the future, it is important to take qualitative information into account when making decisions (Drury, 2007). For example, an organization may decide to develop a new product in half a year, which needs different labour skills. The new product may be profitable, but current employees may feel that they need to make room for new people, making them feel less motivated. Though it is difficult to predict if such situations will happen, it should be up to decision-makers to decide and estimate the chance of happening. When deciding upon the information presented, organizations should bear in mind the costs that incur when a second best option is not chosen. For example, if an organization decides to create 200 products A that incur 150 000 total costs in currency (e.g. pounds) with 150 labour hours to be spent, the organization should also bear in mind that this may take more resources. The resources spent extra could also be used for creating 100 products B, which has lower costs. Let's say that with the same amount of extra labour hours spent on creating product A, 100 products B could be created at a cost of 50 000. Product B has production costs (500 in currency, versus 750 in currency for product A). However, more products of product type A can be sold in the market, though product B may be sold in a smaller amount. Deciding which option is best needs to take into account the costs that are sacrificed when not doing the other option. Table 15 shows the two decisions that can be made. For decision I, only the costs are relevant that distinguishes this decision from the other. The differences is the amount and type of products created: for our situation, decision I involves 200 units of product A. Decision II involves 133 units of product A and 100 units of product B. All is produced under within the same capacity constraint, i.e. 150 labour hours. What distinguishes decision I from decision II is the difference in opportunity costs. If we choose decision I, we have higher costs, because the organization cannot sell 100 units of product B, priced at 800 in currency. This situation is likewise for decision II; the organization cannot sell 67 units of product A, priced at 900 in currency. The management of the organization may make decision II, due to its lower costs for the organization.

Table 15. Example of two scenarios for decision-making.

	Decision I	Decision II
Total costs Product A (200 units, 750 costs in currency per unit)	150 000 (200 units, 150 labour hours)	100 000 (133 units, 100 labour hours)
Total costs Product B (100 units, 500 costs in currency per unit)	-	50 000 (100 units, 50 labour hours)
Total production costs	150 000	150 000
Opportunity costs (of not creating and selling product B (sales price 800))	80 000 (100 units Product B, 50 labour hours)	
Opportunity costs (of not creating and selling product A (sales price 900))		60 300 (67 units Product A, 50 labour hours)
Total relevant costs under opportunity-cost approach	80 000	60 300

Horngren, Datar, & Rajan (2012) describe a *five-step decision-making process*, which may be used as a decision-model for decision-makers (e.g. managers, boards of directors, etc.).

1. Identify the problem and uncertainties
2. Obtain information
3. Make predictions about the future
4. Make decisions by choosing among alternatives
5. Implement the decision, evaluate performance, and learn

In the first step, the problem is identified, like ‘should the manufacturing process be reorganized to obtain lower manufacturing costs?’. Uncertainties are included as well, like the uncertainty of knowing how employees will respond to the choice made.

In the second step, information is obtained , like the current costs being made for manufacturing. Also, interviews may be done to retrieve information about the uncertainties, e.g. to obtain information on whether employees would feel negative about a reorganization.

In the third step, predictions are made based on the information provided by the second step. Calculations are done and scenarios may be created to show the options for decision-making.

In the fourth step, decisions are made about which scenario would be most beneficial for the organization. Here, both quantitative as well as qualitative information should be presented.

In the fifth step, the decision made in the fourth step is implemented and, once implemented, evaluated upon how it worked out for the organization. Once it has been evaluated, the evaluation information is taken into account for future decision-making. For example, if the outcome would be less beneficial than expected, the managers might learn that decisions should be taken differently.

Cost Analysis Types & Methods for Decision-Making

Looking at the third step of the 'five-step decision-making process' (Horngren et al., 2012), information is obtained and used for calculation, e.g. to predict future outcomes that may result from decision-making. Cost information may be available, but may also be calculated based on available data. Regarding costs, many types of methods, systems and approaches exist. We have named three main topics regarding management accounting (internal reporting), namely *cost allocation*, *decision-making* and *planning, control & performance management*, based on research performed by (Drury, 2007), of which we have put our focus on cost allocation and decision-making.

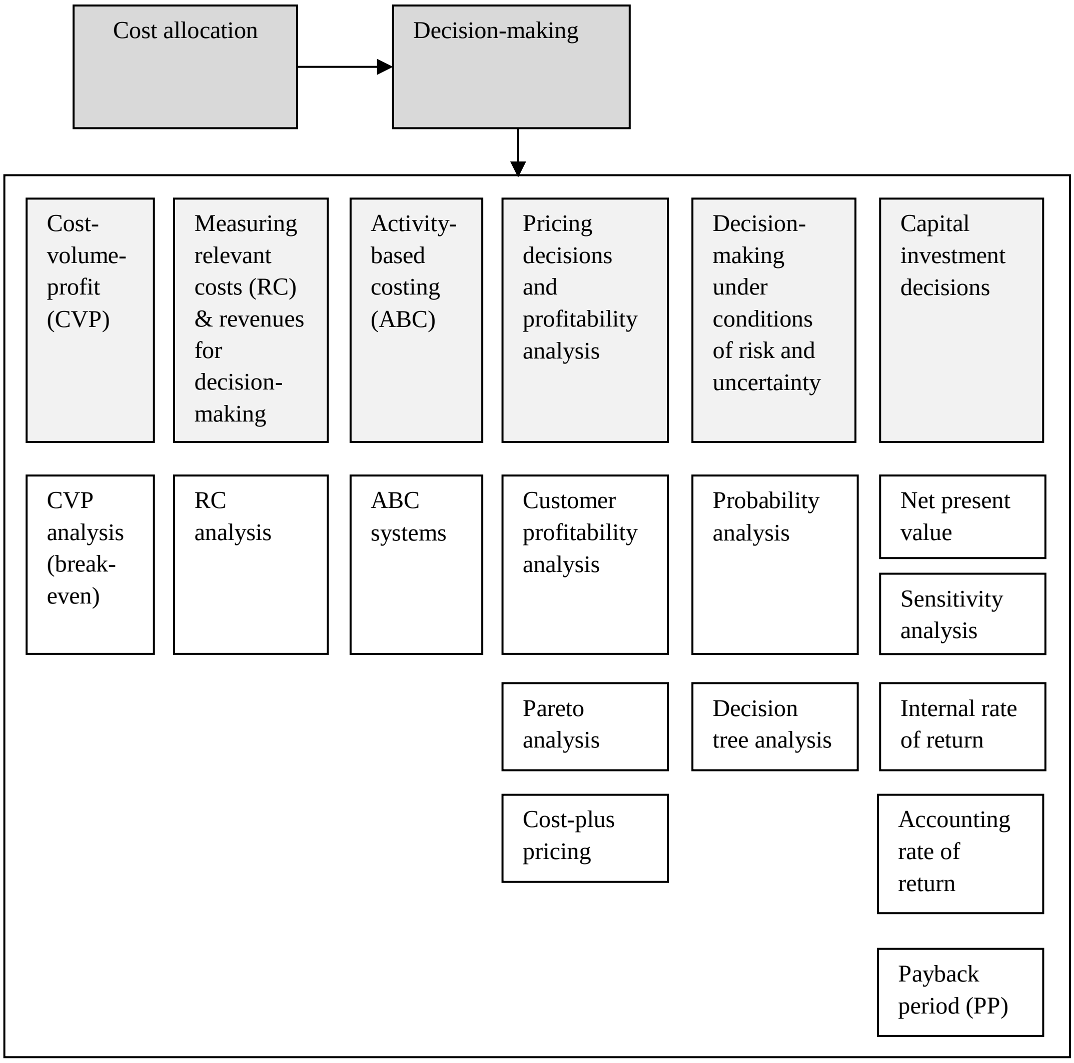


Figure 39. Overview of popular cost analysis techniques and systems, categorized in decision-making topics.

For decision-making, several quantitative techniques may be used for calculating with cost data for analysis purposes. As we have explained, information derived by performing cost allocation may serve decision-making. Information for decision-making itself may be split up into several topics, shown in Figure 39. Cost analysis, sometimes referred to as economic analysis, is a term used in many different contexts. For evaluation purposes, three different types of cost analysis may be identified, namely *cost allocation*, *cost-effectiveness analysis* and *cost-benefit analysis* (Sewell & Marczak, n.d.). These *cost analysis types* include systems and methods for which some of the cost analysis techniques mentioned in Figure 39 may be used. For example, the net present value (NPV) and sensitivity calculations are used in the cost-benefit analysis type .

Figure 40. Spectrum of cost analysis types for evaluation purposes.

General Summary

In this chapter we have discussed the topic of cost information for decision-making. For decision-making, only relevant costs should be used. Relevant costs may be determined based on the criteria that they *should occur in the future* and that they should *differ among the alternative courses of action*. Depending on the decision-making need, a cost analysis may be performed to provide the decision-maker (e.g. a manager or a CIO) the information needed. We have provided different overviews on the topic of cost analysis. We have explained that many different techniques for cost analysis exist that may serve as ways to provide information for decision-making; these types are evaluative. Next to *evaluation*, the topic of *planning, control & performance management* exists. However, this topic was not discussed in this thesis due to our focus on the so-called ‘special studies’ that address non-recurrent situations like having to react on competition in the market, whereas *planning, control & performance management* involve recurrent activities.

Conclusions Part III

In this part of the thesis, we have provided a literature review on different topics. First of all, we started with the concept of *information,* its relation to *data* and related concepts like *communication*. We have discussed how data is digitally stored using concepts like *databases* and *data warehouses,* amongst other technologies. These topics were addressed to clarify how organizations store their data. We have discussed how these data storages could be structured using data warehousing architectures, since many large organizations use a form of data warehousing in their daily operations. We have briefly introduced the topic of *enterprise architecture* and explained its main subjects. All these topics deal with storing and structuring data. Costs, if digitally stored, become cost data. For decision-making, costs are important, since organizations need to keep control over their resources before going bankrupt. We have provided several overviews of methods, systems and techniques related to the topics costs and decision-making. In the next part of this thesis, we will explain how enterprise architecture and operational data may be used to better support in decision-making.

- Page intentionally left blank -

IV. Combining Enterprise Architecture & Operational Data to Better Support Decision-Making

Having explained both the field of enterprise architecture as well as the origins of enterprise information and, more specifically, enterprise data, this chapter will explain how enterprise architecture and enterprise data (or operational data (OD)) are related.

4.1 Connecting operational data to enterprise architecture

As previously described, each organization deals with different situations that influence the organization's structure. Organizations may for example need to adapt to changes in regulation and change parts of their corporate entity in order to stay compliant. However, there are many more situations we might think of when discussing organizations. In any case, organizations are faced to make decisions about certain aspects. In doing so, they need to keep in mind their corporate entity and therefore the way decisions may affect their organization, e.g. changes in IT systems as a result of a strategy decision. Decision-making is often done based on the *available information* an organization has to offer for their board of directors. As previously explained, this information is derived from data, which is saved into storage facilities like databases and data warehouses. The rise of enterprise architecture as the 'fundamental management information system' in an organization (D. Simon, Fischbach, & Schoder, 2013) means it is increasingly conquering its place in the 'board room', where decisions are being made. The central place EA is deriving in the corporate world means organizations might start thinking about what they can do with it, since they might have all kinds of data sources being *described* by the enterprise architecture, but a dedicated connection to the data sources are not always implemented.

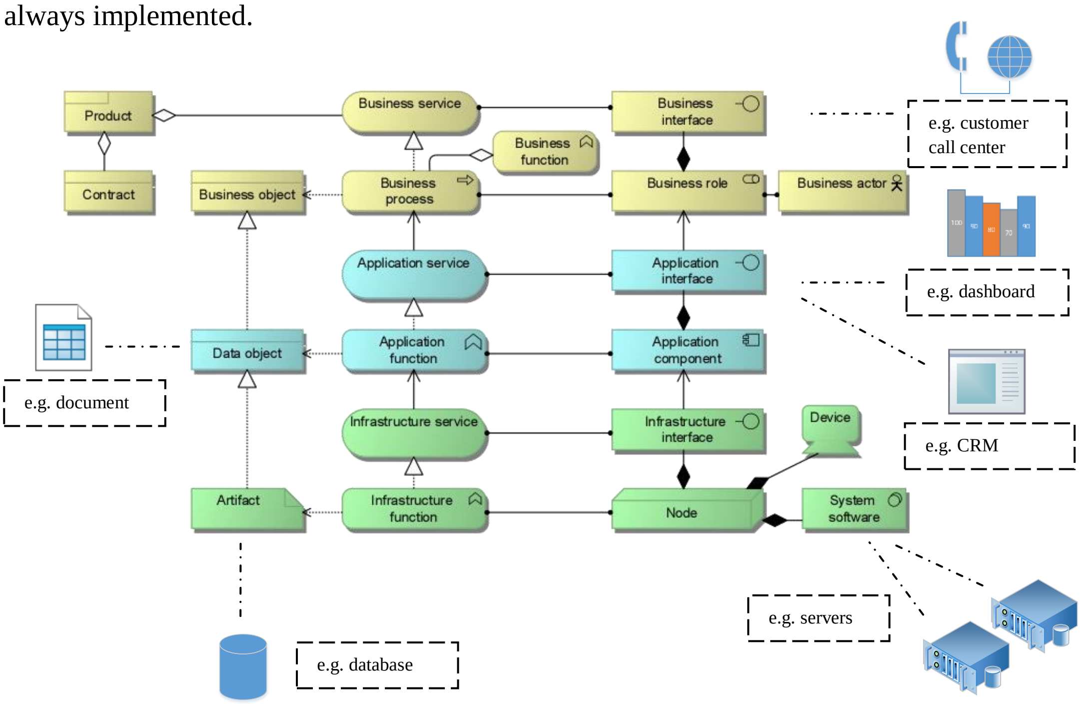


Figure 41. Some Operational Data concepts and their relation with EA.

Such a connection could be an interesting way to align the ‘physical’ operational data with the ‘structured description of the enterprise’ that EA brings. Due to the considerable number of organizations ***facing difficulties bringing strategy to execution***, as well as the ***lack of structure*** and ***transparency*** the corporate strategic management deals with (D. Simon et al., 2013), corporate strategic management could be helped to overcome their difficulties when enterprise architecture and operational data can be combined in a way that they ***bring structure*** and ***transparency***.

Surely, enterprise architecture in itself might bring structure and, to a certain extent transparency as well, however, enterprise architecture remains loosely coupled with operational data, which often is the de facto standard source for business intelligence. The question rises how operational data (OD) and enterprise architecture (EA) should be combined to better support decision-making.

Figure 41 shows how some operational data (OD) concepts and their relation with enterprise architecture could be made. Having a connection between OD concepts (e.g. databases, servers, data warehouses) and EA might hand new options to see the operational data within an enterprise architecture’s structure. However, other benefits might exist and even other combinations might be possible between EA and OD.

The connection between EA and OD in itself is, however, already challenging. Currently the standard for enterprise architecture is the ArchiMate language, as explained in Chapter 2. However, the structure of how the language should be implemented in applications (e.g. BiZZdesign Architect) is, until now, not standardized. Implementations of ArchiMate could be done in a markup language like XML (Extensible Markup Language), which is the language that is used in BiZZdesign Architect to define the structure of EA and save it as a model package with extension .XMA (BiZZdesign, 2013). Luckily, XML is widely accepted as a format in the IT world, making it feasible to connect EA and OD using XML related transformation and exchange techniques.

The focus in our research, however, is not on the technical implementation of a connection establishment between EA and OD. The way *how* an implementation should be done and *what it adds* is more interesting and hands freedom to the technical side of realizing the concept. In the next chapter, we will describe how EA and OD can be combined in order to better support decision-making.

Having explained the fundamentals of operational data, enterprise architecture and related concepts, we turn our view on these technologies by looking at what the end users might need. This chapter explains how operational data and enterprise architecture can be used in several scenarios.

4.2 On Decision-making in Organizations

In time, ***data warehouses*** and related techniques have been a way to structure and cleanse large amounts of data as a means of providing a 'clean library of *data'*. This data that is stored in different types of data warehouses is then used to derive *information* out of it, as we have previously explained the difference between data and information and their relation to one another. This information is valuable for decision-making; enterprise top-level management (e.g. Chief Information Officers (CIOs), Chief Executive Officers (CEOs) etc.) and even lower-level management may benefit from insights delivered by business intelligence using data sources like data warehouses. In any case, the people that are using data warehousing and related technologies often end up in using ***business intelligence*** technologies to represent and analyse the data inside these data warehouses. ***Enterprise architecture*** is increasingly being embraced, both by researchers as well as practitioners, as a *representation of the current state* of the organization as well as a *planning tool* (Kappelman, Mcginnis, Pettite, & Sidorova, 2008). Most organizations that use EA are dealing with large and complex IT landscapes, making it hard to keep track of what is going on. EA hands possibilities to model the IT landscape and its relation to business, like a blueprint. However, this is just the *baseline*, the current state of the landscape. For organizations that are heading towards a different state of the organization, e.g. by introducing new products or services in a few years, it is possible to create a *target* situation. Adding more states, like temporary states (called *plateaus* in ArchiMate) of the IT landscapes before reaching the target, would make up a roadmap. Using these different states gives organizations the opportunity to see differences between situations and hands information for *decision support*. Traditional organizations that gradually grow from small to large in size might initially not consider using EA. However, when situations tend to become more complex, enterprise management might consider using EA to solve their problems. In any case, these large organizations often already have a *data warehouse* environment, in whatever form. Adding enterprise architecture as a 'different world' to the existing 'traditional organization' might be challenging. Shortly stated, enterprise architecture can bring the 'helicopter view' to an organization with options to further specify and relate concepts towards the actual hardware and software environment.

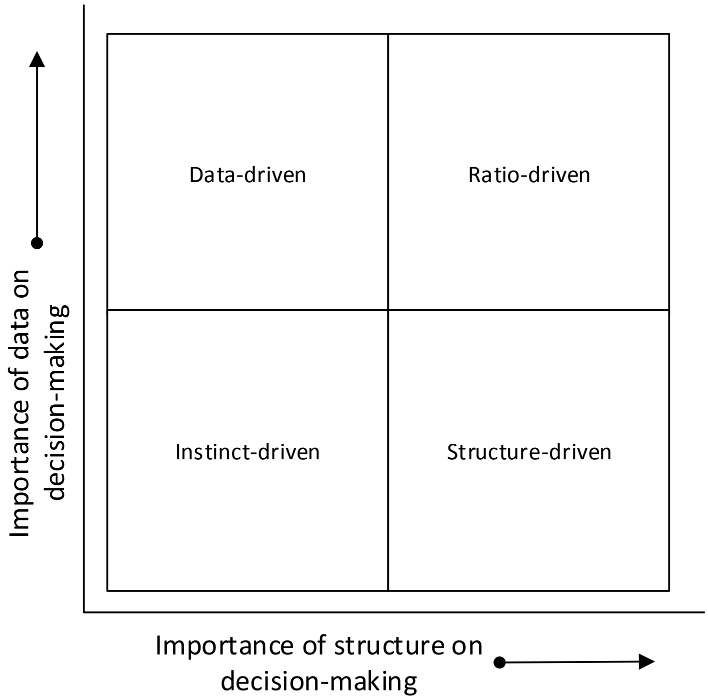
In a report of IBM (2010) on dealing with complexity, more than 1500 CEO spread over 60 countries and 33 industries were asked what challenges they find hardest to cope with. Most CEOs mentioned ***complexity*** the hardest challenge they deal with, i.e. *"a world that is substantially more volatile, uncertain and complex"*. These CEOs are doubtful about the future, and are uncertain about whether they are able to cope with an increasing complexity or not. However, still there are ***decisions to be made*** in this complex world. According to IBM's report, there are standouts that manage to cope with complexity. These organizations often *"take more risks, find new ideas, and keep innovating in how they lead and communicate"*. When asking for external factors that have the biggest impact on organizations, CEOs mentioned *market factors*, ***technological factors***, *and macroeconomic factors* as the top three most important factors. According to IBM's report, *"it is no longer sufficient, or even possible, to view the world within the confines of an industry, or a discipline, or a process, or even a nation. Yet the emergence of advanced technologies like* **business analytics** *can help uncover previously hidden correlations and patterns, and provide greater clarity and certainty when making many* ***business decisions"*** (IBM, 2010).

Looking at these business concerns, we know that technological factors play a large role in decision-making. With advanced analytic technologies, like many business intelligence solutions, we are able to help management in supporting their decision-making. The question, however, remains what to provide them with. This depends on the situation an organization is in, which we will discuss in this chapter.

4.3 Personas – Corporate Management Archetypes

For organizations having to decide upon certain situations, *different types of corporate management* may be categorized. Corporate management is a management team consisting of people and people have a mind-set with which they approach situations they need to decide upon. Depending on the mind-set a corporate management member has, like a CIO (Chief Information Officer) or a CEO (Chief Executive Officer), the management takes different decisions. In a way, these management members could be categorized in the way they approach decisions.

Many categories (or classifications) could be made to *archetype the managers*. These corporate management ***personas*** are useful when thinking about decision-making, since the way a person thinks determines the way a person decides and therefore the way an organization responds to situations. Depending on whether the right decision was made, the organization performs well or not. Thus, the type of corporate management personas has an effect on the way an organization performs over the years, depending on the situations that occur.



There are multiple archetypes that can be determined for corporate management. For our research we are focusing on two dimensions, data and structure. A corporate management archetype describes the mind-set of a member of the organization's management. It the way of thinking itself and consists of the principles one has regarding decision-making.

Assuming that the available amount of correct data has a positive effect on decision-making as well as the structure (or overview) one has regarding the concern that has to be decided upon, we have determined the following personas. The ***instinct-driven*** (adventurer), the ***data-driven*** (data-man), the ***structure-driven*** (strategist) and the ***ratio-driven*** (rationalist) persona.

The ***adventure-driven*** persona, or adventurer, is a person who decides upon 'gut-feeling'. He instinctively knows what to decide when a problem or concern shows up. This person is not dependent on any specific form of input for decision-making, but this does not mean his decisions are wrong or might not turn out right. The decisions that are made could be improved by providing accurate input, which may change the view the adventurer had on a situation at first.

The ***data-driven*** persona, or data-man, is a person who decides upon the data that is given (e.g. using business intelligence (BI) applications). Using the data shown to him, for example visualized in BI-dashboards, the data-man is able to determine the action that is needed when a problem or concern shows up. The data-man is able to translate the data in his mind into an action-plan that can be executed to solve the problem. The data-man does not rely on a conceptual view of the situation, but instead ‘reads numbers’.

The ***structure-driven*** persona, or architect, is a person who needs an ‘overview of concepts’ and is relying on relations that can be drawn between these concepts (which is a structure). Based on the overview he has and the structure he sees, he can strategically decide upon concerns. The architect does not rely on the data presented (e.g. in numbers, etc.), but tries to see the whole picture and relate things that are important for the specific concern that needs decision-making.

The ***ratio-driven*** persona, or rationalist, is a person that needs both structure as well as data for his decision-making. While confronted with specific concerns, the rationalist needs to see the ‘total picture’ (structure) that might be affected by decision-making, e.g. certain processes that change, customers that may be affected, and so on. To base his decisions upon facts, the rationalist needs data that is tailored to the specific concern. A combination of both structure and data is needed to receive enough information, which is the result of the interpretation of data, for his final decision-making.

4.4 Scenarios – Personas in Fictive Situations

Organizations operating in different areas have different needs regarding information that should be provided to support decision-making. We have briefly discussed the era of technology in this document where we focused on the concepts of *data* and *information* as the main threads, while keeping in mind that *"information is data plus sense-making"* (Beynon-Davies, 2009a). In the words of Inmon (2001), these 'corporate information factories' produce large amount of data in their lifecycle, but not all necessarily in the same way. In this chapter we will explain four *scenarios* organizations may have faced in their lifecycle. These scenarios are ***archetypes***; in most cases organizations will have a *hybrid* or a slightly different form of situation they deal with compared to the scenarios described. The scenarios are based on the topics we have explained in this document, namely data sources and enterprise architecture, and always have an initial situation with a typical concern. To illustrate the concepts of *data* and *information* in these scenarios, we will use *monetary costs* and *value*, since they are assumed to play a role in every enterprise, be it a large or a small role. Shown in the figures below, we see that an organization transitions from a 'current situation' to a 'new situation' and between these two situations decisions are made; this is the case for all organizations and all managements in every moment of the day. It is the way of thinking and the available information that determines the decision-making, which may result positively or negatively for an organization.

A. Scenario 1: the adventure-driven management

The organization is being managed by people that decide upon instinct, people who do not need information to be able to decide upon situations. In this scenario, the ***adventurer*** plays a role. In a situation where a particular concern is on the management agenda, the adventurer decides upon 'gut-feeling'. By making the decision upon the particular concern, the organization heads for a new situation.

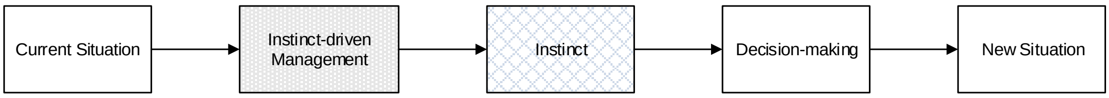


Figure 42. Instinctive decision-making.

B. Scenario 2: the data-driven management

The organization is being managed by people that decide upon data presented, people who need numbers and other quantitative data to be able to decide upon situations. In this scenario, the ***data-man*** plays a role. In a situation where something needs to be decided upon, he bases his decision on the data provided. By doing so, the organization heads for a new situation.

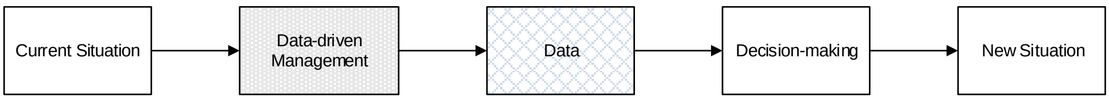


Figure 43. Data-driven decision-making.

C. Scenario 3: the structure-driven management

The organization is being managed by people that need to be able to see a relation between certain aspects in the organization, people who need the 'total picture' of important stakes that need to be considered when making a decision. In this scenario, the ***strategist***, plays a role. He is the one who decides upon overseeing the overall view on the problem with which he is able to make predictions or to relate impacts that certain choices may have on a specific concern.

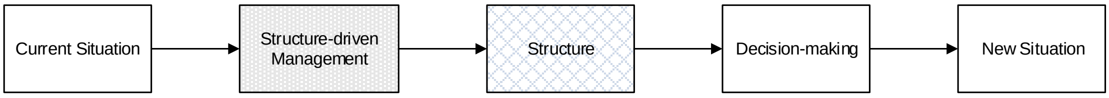


Figure 44. Structure-driven decision-making.

D. Scenario 4: the ratio-driven management

The organization is being managed by people that need all the information they can get, decide upon a concern after careful observation of the situation and need to understand what is happening. In this scenario, the ***rationalist*** plays a role. He is a person who wants to see the total picture of important stakes (and their relation) regarding a specific concern while using data (e.g. analytics) to see and compare upon decision options. In this scenario, both structure and data are combined to provide a full scope on a specific concern.

Figure 45. Ratio-driven decision-making.

E. Decision-making accuracy

For all of the scenarios described above, there is no way to state that one organization is better at decision-making than the other. The mentioned scenarios may not cause issues when deciding upon concerns and therefore do not determine the success of an organization. However, the accuracy of determining the actual cause may be improved if both the data-driven management and structure-driven management share their worlds and combine the 'business-view' (structure) and the 'data-view' (operational data). Currently, the field of enterprise architecture provides the structure needed for the structure-driven management. For the data-driven management, business intelligence applications provide the representation of data needed to make decisions. If we are able to combine both worlds, we believe decision-making can be made more accurate, since concerns may be outpointed accurately (e.g. in an enterprise architecture) while combining it with data e.g. for quantification of a specific concern.

Conclusions Part IV

In this part, we have discussed several topics. We started off by explaining in concept how *enterprise architecture and operational data* could be combined. We explained the need of a methodology to combine both worlds, since currently a method is lacking. Furthermore, we briefly touched upon how currently both data warehousing and enterprise architecture are being used by decision-makers (e.g. managers and CIOs). We have illustrated different corporate management archetypes that illustrate how people may think in an organization. These archetypes serve as a parallel to the worlds of data warehousing, business intelligence solutions and enterprise architecture. Lastly, we have illustrated ways how these corporate management archetypes think in order to show where possible gains may be to support them.

- Page intentionally left blank -

V. Organization Lifecycle Approach to better support Decision-Making

The Enterprise Architecture Intelligence Lifecycle

The scenarios we previously described are situations any organization may recognize. As explained, they are *fictive* and *extreme* situations; though some may face the exact situation, many will have hybrid forms of the mentioned scenarios or other forms that have not been mentioned. In a way, decision-making is putting strategy to practice. The decision to head for a point in the future hints that the future situation seems to be better, since why would an organization head for a situation that would not be beneficial? To support their strategy, organizations need to set directions. In order to do so, they need to decide based on the data or information they have.

> In an empirical study on 'effective information delivery and effective information use from a senior executive's perspective', it was confirmed that "*the organization's information systems' **ability to deliver integrated information** to people in the organization's hierarchy and processes **positively influences** the effective organizational use of information to support business activities and **strategies***" (Kettinger, Zhang, & Chang, 2013).

Knowing that an organization's ability to deliver integrated information to people and processes positively influences the effective use of information for decision-support, we are interested in *how* we can combine and integrate existing sources of information, like enterprise architectures and data sources, to positively influence decision-support. The previously described scenarios are a way to illustrate the current situations of such organizations and how they can be helped in improving their 'effective organizational use of information to support business activities and strategies', as described by Kettinger et al. (2013). We will describe two approaches for organizations to improve their information systems' integrated information delivery ability. Both approaches are ways to lead management towards a more ratio-driven decision-making, which we consider to be beneficial for accurate problem-solving.

> To illustrate our ideas, we will make use of the openly available ***ArchiSurance*** case, concerning the insurance company ArchiSurance, which is the result of a merger of three previously independent companies based in different metropolitan areas, namely *Home & Away*, *PRO-FIT* and *Legally Yours* (Jonkers, Band, & Quartel, 2012).

The approaches we describe can be integrated with the *'EA-based decision-making method'* of Priyanto (Priyanto, 2013). Some adjustments were done, but the main idea he describes remains. However, since we believe the end result is determined what happens 'in the middle' of the process, we describe a generic approach and two specific approaches both having a different focus on data delivery. The first approach focuses on enterprise architecture and attempts to hand possibilities to 'enrich' it with operational data. The second approach focuses on operational data sources and attempts to hand possibilities to 'enrich' it with structure derived from enterprise architecture. Both approaches are based on the same principles and share a core of activities.

5.1 The Enterprise Architecture Intelligence Lifecycle (EAIL)

As explained in the scenarios chapter, organizational decision-making is done by different types of management, all with different needs. The approach we suggest is based on the ratio-driven scenario and combines both structure and data for accurate decision-making. The approach, we refer to as *Enterprise Architecture Intelligence Lifecycle (EAIL)* consists of six phases an organization goes through when deciding upon a specific concern. The phases are: I-Explore, II-Measure, III-Enrich, IV-Visualize, V-Decide & Change, and VI-Evaluate.

In all phases, several activities need to be performed in order to proceed to the next phase. Our approach includes 'agile' characteristics, like simplicity, focus on the client to be satisfied and frequent communication with the client to synchronize opinions and progress. The approach can be seen as a lifecycle approach; it may be used during the entire existence of an organization, as long as motivation is present and a concern may be determined for the organization.

Figure 46. Enterprise Architecture Intelligence Lifecycle (EAIL).

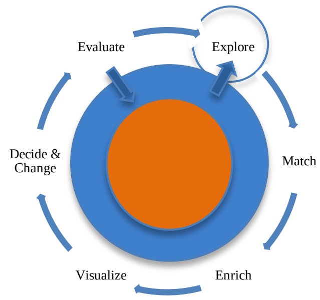
Johnson, Johansson, Sommestad, & Ullberg (2007) have described a 'method for enterprise architecture analysis', explaining three phases: *assessment scoping, evidence collection* and *analysis*. The phase 'assessment scoping' discusses the *problem* (in our approach referred to as 'concern') and discusses *assessment criteria* (in our approach referred to as 'variables'); the translation of a concern into measurable criteria that can be assessed. The phase 'evidence collection' is about *creating data models* for different scenarios and about *gathering evidence*, which in our approach is explained in the 'enrich'-phase. The phase 'analysis' is discussed in the 'enrich'-phase of our approach as well. As additional phases and to look back on results provided using our approach, we have included a 'decide & change'-phase in which a decision is being made, a *solution is developed* and *implemented* that may be *evaluated* upon solving the initially determined concern. This evaluation is done in the 'evaluate'-phase of our approach. Whereas Johnson et al. (2007) discuss different scenarios for future decision-making, used for comparison in decision-making, our approach does not determine future scenarios due to the *complexity of impact* different choices may have on an organization. Our approach attempts to present the optimal combination of enterprise architecture and data sources to support decision-making. Johnson et al. discuss that the importance of selecting and collecting 'evidence' as input for their scenarios become more important as the model grows larger (e.g. more complex, i.e. difficult to manage manually). Our approach hands possibilities to overcome this issue. Next to the study of Johnson et al. (2007), Horngren et al. (2012) provided a five-step decision-making process, mentioning these steps: *(1) identify the problem and uncertainties, (2) obtain information, (3) make predictions about the future, (4) make decisions by choosing among alternatives, (5) implement the decision, evaluate performance and learn.* These steps of Horngren et al. (2012), together with the phases provided by Johnson et al. (2007) provided the inspiration for developing the EAIL.

The focus in our approach is to minimize the organization (client)'s concern(s) or to respond to its motivation. An organization starts in a *current situation* where *motivation* is a driver for going towards a new situation; the organization wants to go to a *better situation* where the concern is addressed. The organization decides to respond to this motivation by initiating the lifecycle. Both enterprise architecture and (operational) data source(s) are needed to be able to combine for improving decision support. They are the premises that need to be met to use our approach.

Premise I: presence of an enterprise architecture (EA)
Premise II: presence of (an) (operational) data source(s) (DS)

For all activities mentioned in the phases we acknowledge that sidesteps might be useful, like dividing tasks between an internal organization's actor (e.g. a data specialist) and a consulting firm's actor (e.g. an enterprise architect). Also, it might be useful to define some 'interim steps', which may lead to splitting activities in smaller activities. It may be good to specify these activities that differ from our approach to avoid misunderstandings.

Phase I – Explore



In the first phase, *explore*, we assume the organization has consulted another firm to act upon their motivation. The first phase starts by performing the first activity, i.e. determining the concern. Once the activity is performed, the next activity is initiated, until the phase ends.

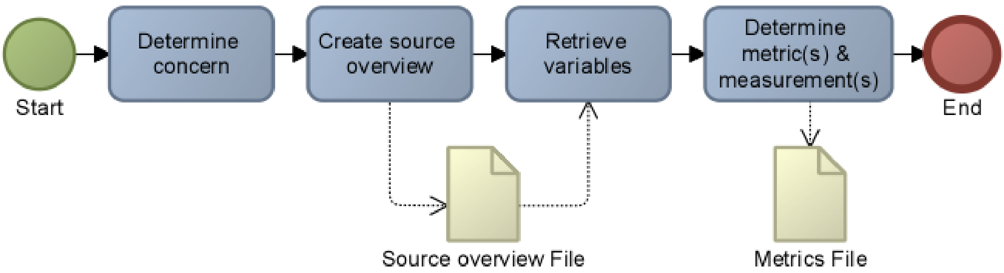


Figure 47. Processes in Phase I - Explore.

I-a Determine concern

In this phase, the consulting firm may use several techniques to point out the exact problem, the *root cause*. Finding the root cause may eventually help in determining a useful solution to solve the initial concern. In this activity, *determine concern*, the organization may also define a *measureable goal*, which is *"an end state that the stakeholder intends to achieve"* (The Open Group, 2013a). For example, if the concern is that profit has lowered, the measureable goal could be to lower costs. In any case, the consulting firm tries to find out what the exact concern or problem is in order to know what to look for. It is critical that the consulting firm is able to make the concern tangible either by finding the root cause *or* by determining a measureable goal.

Table 16. Description table of the 'determine concern' activity.

Phase I-a	***Determine concern***
Activity description	The determination of a concern.
Possible actor(s)	Stakeholder and Enterprise Architect.
Input	Business initiation request (motivation).
Usable techniques	Interview techniques (Kvale & Brinkmann, 2009), root cause analysis (Rooney & Vanden Heuvel, 2004), etc.
Output	Determined *concern*.

I-b Create source overview

When the concern has been determined, the consulting firm explores the organization (or request an exploration performed internally by the organization itself) to find out what the current states of the enterprise architecture and operational data sources are and to *create a source overview*. This source overview may be a global interpretation of the enterprise architecture and its data sources, but preferably is a concise document that can be exchanged or referred to. It serves as an inspiration for finding measurable variables.

Table 17. Description table of the 'create source overview' activity.

Phase I-b	***Create source overview***
Activity description	An exploration activity performed by the Enterprise Architect and Data Specialist to find suitable data in the organization in order to support the solving of the previously determined concern.
Possible Actor(s)	Enterprise Architect and Data Specialist.
Input	Determined *concern* (possibly translated in ‘business request’).
Usable techniques	Interview techniques (Kvale & Brinkmann, 2009), requirements analysis techniques (Hay, 2011).
Output	An overview of available data sources and a description of the data that is stored in these data sources (possibly outputted in a source overview).

I-c Retrieve variables

Having created a source overview, the consulting firm tries to translate the concern, measureable goal *or* root cause into variables that can be measured and found in the data sources. These variables are the smallest parts upon which we may later determine a metric. The more variables we are able to retrieve, the more options we have in determining a suitable metric. Examples of variables are units or rates like *processing time* or *workload rate*.

Table 18. Description table of the 'retrieve variables' activity.

Phase I-c	***Retrieve variables***
Activity description	The retrieving of ‘variables’ from the organization out of which a metric exists.
Possible Actor(s)	Stakeholder and Enterprise Architect.
Input	An overview of *available data sources* and *a description of the data* that is stored in these data sources (possibly outputted in a source overview).
Usable techniques	Interview techniques (Kvale & Brinkmann, 2009), requirements analysis techniques (Hay, 2011).
Output	Retrieved variables.

I-d Determine metric(s) & measurement(s)

When we have found variables, like the ‘processing time’ or ‘workload rate’ previously mentioned, we may be able to determine a suitable metric, for example *resource utilization*.

Determining a metric should be carefully done, since this metric will be used for combining the enterprise architecture (EA) and data sources (DS). Priyanto (2013) provided a template (Appendix B) that can be used as a reference and exchange file for communication when filled in (resulting in a

'Metrics File'). Kerzner (2011) and Hubbard (2010) have provided thorough literature on how these variables and metrics can be determined to be able to 'measure anything'.

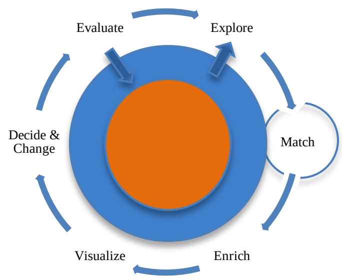
Determining a measurement means *how* we are going to measure the metric, e.g. by using a specific algorithm that uses the previously determined variables to calculate certain values. There may be multiple ways to calculate a metric, e.g. calculating a 'return on investment' (ROI), which has multiple ways to be calculated. In this step, the 'Metrics File' is modified with new information about *how* the metric is calculated.

Table 19. Description table of the 'determine metric(s) & measurement(s)' activity.

Phase I-d	***Determine metric(s) & measurement(s)***
Activity description	The determination of a metric, derived from EA/DS, and its measurement. The metric measurement means *how* we are going to measure, e.g. by using a specific algorithm that uses the previously determined variables to calculate certain values.
Possible Actor(s)	Enterprise Architect and Data Specialist.
Input	A *measureable goal*, possible changes derived from activity 4b.
Usable techniques	Metrics design (Kerzner, 2011)(Hubbard, 2010), requirements analysis techniques (Hay, 2011), algorithm design techniques (Skiena, 2008).
Output	An identified *metric* and its *measurement* (possibly logged in a 'Metrics File', cf. Appendix A)

Phase II – Match

In the second phase, *match*, we assume the organization has performed the previous activities in the EAIL. The second phase starts by performing the first activity, i.e. 'determine EA & DS subsets'. Once the activity is performed, the next activity is initiated, until the phase ends.



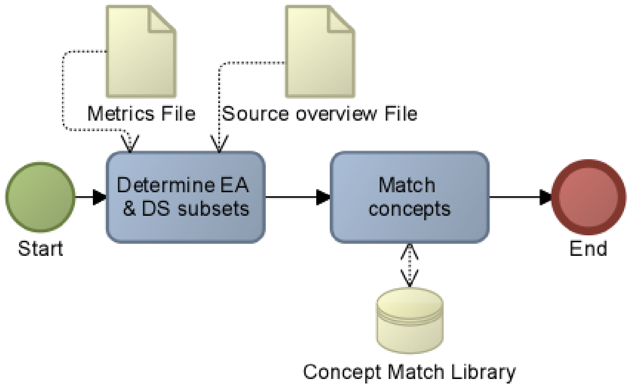


Figure 48. Processes in Phase II – Match.

II-a Determine EA & DS subsets

After having determined how we are going to measure the metric, in the previous activity (i.e. 'determine measurement(s)'), the 'Metrics File' and the 'Source overview File' are used to determine the ***parts of the enterprise architecture and data sources***, which are EA (enterprise architecture) and DS (data source(s)) subsets that are suitable as a source for further steps. It is critical that the chosen subsets from both EA and DS are relevant for the determined metric; the subsets will be used when combining them to better support decision-making in a later phase. For the EA-subset, an enterprise architect could be most suitable for determining which *EA-concepts* are relevant. In determining the subset, a scope needs to be set regarding the enterprise architecture. If the problem is related to the whole organization, the scope consists of the entire enterprise architecture. If the problem is related to only a division of the organization, the scope consists of the related concepts of the enterprise architecture (e.g. business functions, processes, application components, etc.). For the DS-subset, a data specialist who has knowledge about the availability of data in the organization could be most suitable for determining which *DS-concepts* are relevant. To determine the subset, a scope needs to be set regarding the available data. The DS-subset is determined based on the available data, which may be selected based on the information provided by the 'Source overview File'. At the end of this activity, both an EA-subset and a DS-subset are available.

Table 20. Description table of the 'determine EA & DS subsets' activity.

Phase II-a	***Determine EA & DS subsets***
Activity description	Determining the data subsets from EA and DS needed for matching.
Possible Actor(s)	Data Specialist & Enterprise Architect.
Input	A determined metric and its measurement and a source overview, as support sources to identify the EA & DS subsets.
Usable techniques	Requirements analysis techniques (Hay, 2011), data mining techniques, ETL (Kimball & Ross, 2013) etc.
Output	Determined EA & DS subsets.

II-b Match concepts

When both the EA-subset and the DS-subset have been determined, both subsets are compared in order to find matches between concepts. A match could be a certain process within the EA-subset that is related to some data found in the DS-subset. Matching concepts is complex, since it needs an understanding of the DS-subset and the EA-subset. Matching concepts is indirectly based on the previously determined metric(s). While trying to find matches, an EA-expert (e.g. an enterprise architect) and a DS-expert (e.g. a data specialist) might sit together to perform the matching. However, the activity can also be performed solely as long as the expert fully understands both the EA- and DS-subset. Having found concept-matches, the matches can be saved for later use (e.g. for other concerns, etc.) in a 'Concept Match Library'.

Matching concepts is complex, since it requires knowledge of both the enterprise architecture as well as operational data. Matching is done on a 'property and record level', meaning data values (from operational data) should be matched with object properties (from enterprise architecture). Matching may be done using the following guidelines:

Table 21. Concept Match guidelines.

Step	*Description*
1	Select an unmatched enterprise architecture property *or* select an unmatched operational data record.
2	Look whether it corresponds to/reflects any of the records or properties of the opposite source.
3	If the two correspond, create a match; manually insert the two in a mapping (e.g. using a spreadsheet application shown in Appendix C and D or using tailored software capable of guiding a user through matchmaking, i.e. loading an enterprise architecture model and operational database to map the two and store them in a new database (explained later as CML)).
4	If some do not correspond, but are critical to the EA and DS subset, look whether it is possible to 'create data' (e.g. calculating new values based on available operational data) or 'create new objects/properties' in the enterprise architecture.
5	Save the matches for further usage.

Table 22. Description table of the 'match concepts' activity.

Phase II-b	*Match concepts*
Activity description	Matching the concepts that have been determined in the enterprise architecture as well as the data sources. *Note: this is not the enrichment activity where data is combined.*
Possible Actor(s)	Enterprise Architect and Data Specialist.
Input	(1) A defined *metric* and its *measurement*. (2) Set of data (both enterprise architecture and operational data).
Usable techniques	Requirements analysis techniques (Hay, 2011).
Output	A match of concepts derived from the enterprise architecture and the determined data sources (e.g. a mapping).

The Concept Match Library (CML)

In the *'concept-match'*-activity (phase II, second activity), both *operational data* and *enterprise architecture objects* (concepts and relations) are matched (or mapped). The mapping is done using the *properties (i.e. attributes) of an enterprise architecture object* on one side, while using specific *operational data values* on the other side. For a mapping with enterprise architecture, the smallest part of the operational data is needed, i.e. a value of a specific *field*. For a mapping with operational data, the smallest part of the enterprise architecture is needed, i.e. a property of a specific object. These mappings may be stored in a model for multiple purposes, like reuse of matches *saving time* of redoing the matches, to calculate values for *analysis* purposes or even to do *forecasting*.

Determining a match is typically performed *manually* for a specific case, but matches may be reused for future cases. Outputting such a mapping may simply be done using an Microsoft Excel-spreadsheet, as shown in Appendix C, but may also be done using database technology. The mapping shown in Appendix C depicts a view on the model shown in Figure 49. The model may be the design or blueprint of an actual database. The model is explained thoroughly further on in this section.

The model to store enterprise architecture, operational data and time (simply called a 'Concept Match Library *or* CML'), is a mechanism that combines and logs data for analysis purposes. Currently, operational data is often stored in warehousing and other database related technologies. Enterprise architecture provides a clear overview of the organization for several purposes, but is almost never used for analysis purposes. With the proposed model, operational data may be combined with enterprise architecture and the combination is logged with timestamps. Each time an entry is renewed, the fields are updated, leaving a trace of data changing in time. These traces may be *analysed*, like currently done in business intelligence solutions, be may also be used for *forecasting*, e.g. when certain trends of data are recognized. The difference with already familiar analysis and forecasting solutions is that now the translation is made to concepts the business knows, tailored in its enterprise architecture.

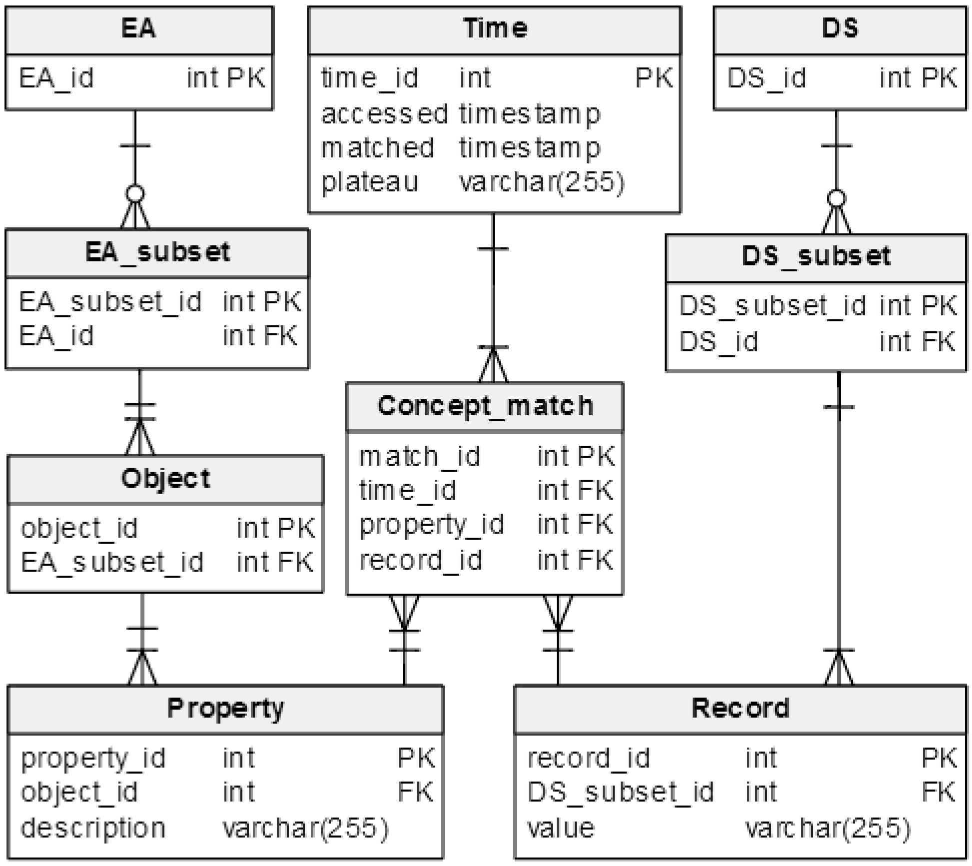


Figure 49. A model to store Enterprise Architecture, Operational Data and Time (CML)

Relation 'Concept_match'

The ***'match_id'*** attribute shown in the 'Concept_match' relation identifies each match that has been made. Each match is unique and updated when the values of the other attributes ('time_id', 'property_id' and 'record_id') change. The match_id attribute is a primary key in the Concept_match relation and each match can be found using this Concept_match relation. An updated match is changed with a new timestamp, indicating the date and time it was changed (explained later as 'matched' in the Time relation). The other attributes are references (foreign keys) to primary key values in another relation. Each Concept_match has one or more *properties* of multiple enterprise architecture objects, one or more operational data *records* the properties are matched with and one or more *time* relations.

Relation 'Property'
A ***'property_id'*** identifies a specific Property of an enterprise architecture object (e.g. a field of any business process indicating a certain 'cost' in monetary terms). It is the object that is matched with data. Next to this identifier, an ***'object_id'*** attribute identifies the object the property is part of. A *description* gives the meaning of a property; it may be used for giving names to properties.

Relation 'Object'
The enterprise architecture object, which may be a relation or a concept in enterprise architecture terms (e.g. a business actor or an application component), is shown in the model as an identifier that indicates the *type of object* (e.g. a business actor) together with a unique number. This is what defines the ***'object_id'*** attribute. For example, a 'business actor' may be filled in into the field together with a number to indicate the business actor in case e.g. there are more business actors related to the same enterprise architecture. Each object is part of a relevant part of the enterprise architecture using the ***EA_subset_id***, for which the Concept Match Library is used. This relevant part is called the EA subset. One or more objects have a single EA subset. In practice, multiple EA subsets may have one or more objects and each object may be part of multiple EA subsets. With the current model, these situations are still possible; a new instantiation of the model should then be created.

Relation 'EA_subset'
The EA subset is a part of the enterprise architecture that is used for a specific case, like solving a concern. The EA subset relates to one or more objects and is part of exactly one enterprise architecture. It keeps track of the enterprise architecture it is part of using the ***EA_id*** and is identified using the ***EA_subset_id***.

Relation 'EA'
The enterprise architecture an organization has, may have several EA subsets (e.g. for multiple relevant and different situations). For example, to show a technical view for a specific cause may not involve the business part of the enterprise architecture, but only the technical objects. Such a subset, i.e. the technical objects of the enterprise architecture, is then used in the model for matching data with. However, all subsets are part of the entire enterprise architecture.

Relation 'Record'
The operational data that is matched with an enterprise architecture object is represented in the *Record* relation. The ***'record_id'*** is an identifier to uniquely indicate each data record that is matched. The identifier never changes and forms a pair with an *object_id*, indicating a match. The values of fields it identifies, surely, may change. The ***'DS_subset_id'*** is an identifier to indicate the subset of the data source used for combining objects of enterprise architecture. Each record is part of one DS subset, i.e. the relevant part of e.g. a data warehouse or any other form of a large data set. The ***'value'*** attribute indicates the value of a field and directly relates to the *'description'* attribute in the *Object* relation. The *value* is retrieved automatically using an external data source (e.g. an external database or file) or is filled in manually. The *value* data type is undefined; it may be any type of data, depending on the data that is filled in or retrieved automatically.

Relation 'DS_subset'

The DS subset is a part of any data source (DS) used in the model. It may represent a part of a data warehouse, e.g. only the data relevant for a certain division of the organization. Each DS subset is part of a larger DS, represented by the ***DS_id***. For each DS subset, a ***DS_subset_id*** is given to identify the subset. Each subset consists of one to more records and is part of exactly one DS. In practice, many different data sources may be used that create the DS subset, however, we believe it is better to use a 'single source of truth', i.e. using a data warehouse that maintains a single version of the data.

Relation 'DS'

The DS is the data source used to combine with an enterprise architecture. It is an aggregated form of data, e.g. a data warehouse. Each DS may have zero to many DS subsets.

Relation 'Time'

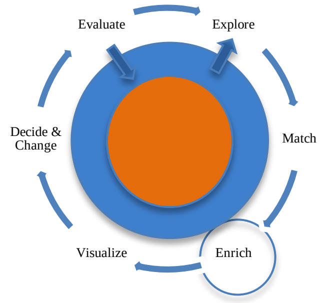
Next to the attributes defined for both operational data and enterprise architecture, another 'dimension' (relation) is added that stores *time*. Time may be used to *identify* when a row was inserted, a field was updated or even to *'trace data'* for analysis purposes. The ***'time_id'*** is an identifier and primary key of the *Time* relation. It uniquely identifies each *Time* record. Each identifier is part of a concept match and its number never changes, however the fields from other attributes in the record it identifies may change. The ***'matched'*** is an attribute that indicates when the match was created. Its data type is *timestamp*, representing the date and time of creation. A timestamp in MySQL is created based on the database server's time. The timestamp in MySQL is converted automatically to UTC time for storage and back from UTC to the current time zone for retrieval. The ***'accessed'*** attribute indicates when a match was accessed, e.g. when a calculation has been done using the data record. Its data type is timestamp, as described in the *matched* attribute explanation. Whenever the match is *accessed*, i.e. either the data it relates to (in the *Record* or *Property* relations) is changed or used, the value of the *accessed* attribute is updated to the current time. This is done using the current time of the server and converting it to UTC for storage. With the *accessed* attribute, analysis like tracing data or even *forecasting* may be made possible. For example, if an analysis uses several attributes in a concept match for calculating values, the timestamp is updated. Timestamps that are similar have most likely been used by a single calculation. Being able to indicate these fields hands an answer to questions like '*where does this data come from?*', whenever results for a calculation are presented. It saves time to figure out the actual sources being used for a calculation. The ***'plateau'*** attribute indicates to which 'plateau' or phase the record belongs in enterprise architecture terms. The *plateau* attribute is standard configured to belong to a *baseline*, indicating the current situation of an enterprise architecture. However, for planning and roadmapping and forecasting, the attribute may be set to a future state of the enterprise architecture. The concept match is then part of a planning for a future state of the organization. Attributes and their values then e.g. represent a situation the organization wants to achieve.

Implementation of the Concept Match Library (CML)

Implementing a CML does not necessarily need to have a large impact on software performance. The CML refreshes data on a predetermined moment, which is the key to software performance. If data would be refreshed every minute, the CML would grow towards a large database with lots of redundant data, since data might not be changed that much in such a short timeframe. If the timeframe would be set every day or week, the amount of data is minimized by a lot. For large enterprise architectures with e.g. over a thousand concepts and relations and an average of five properties per EA object, this would make up the sum *1000 x 5 x 365 = 1 825 000 records per year*, if the timeframe would be set to a day. Considering large databases are often in the 10^9 amount of records area and small databases in the 10^5 amount of records area, this case could be compared with a small to medium database size (in 2014). If the refresh rate of data would be set to every hour, which may be considered high, this would make up *24 x 1825 000 = 43 800 000 records per year*, which is still in the order of 10^7: an acceptable database size. Of course, this all depends on the application and the type of data that is refreshed and measured. The smaller the refresh rate, the more data is collected, the larger the database becomes, but the more precise and accurate predictions can be made. Depending on the *size of the enterprise architecture*, *the amount of matches to be refreshed*, and *the timeframe including duration time* that data is collected, the CML becomes larger or stays concise. In most cases, however, the CML will be small for a small EA and DS subset (e.g. a single problem for which only a subset of the entire EA is needed). CML technology may be implemented in an enterprise architecture suite as a functionality. Currently, however, enterprise architecture applications are not focused on connecting databases to their applications to feed or enrich an EA with data due to their overall focus on strategy. Implementing a CML that auto-refreshes data in an enterprise architecture application requires external database connectivity to request data for each time interval. However, if this is available, this could mean a serious contribution to enterprise architecture due to the CML's ability to provide data for forecasting.

Phase III – Enrich

In the third phase, *enrich*, we assume the organization has performed the previous activities in the EAIL. The third phase starts by performing the first activity, i.e. 'determine data model'. Once the activity is performed, the next activity is initiated, until the phase ends.



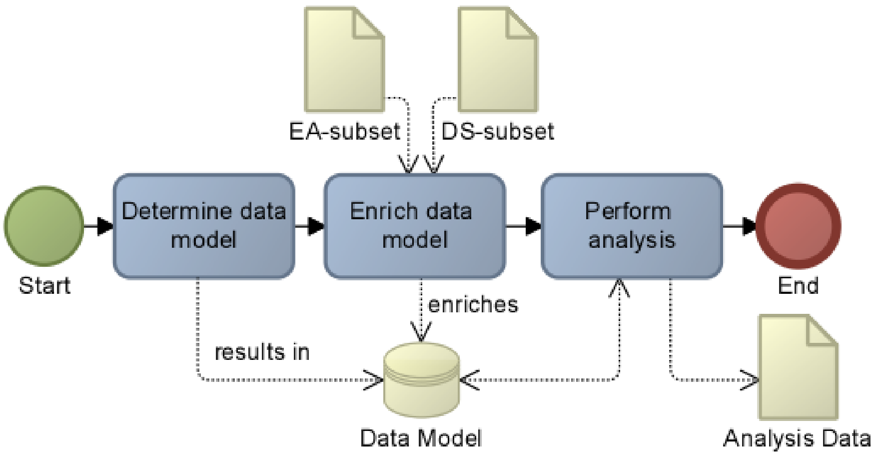


Figure 50. Processes in Phase III - Enrich.

III-a Determine data model

Having matched the concepts from both EA and DS subsets, the next step is to determine the output data model. Depending on the output preferences of stakeholders (e.g. management) and whether enterprise architecture or the data sources are able to submit to these preferences, a data model is chosen. The choices are either creating a *new* enterprise architecture (i.e. a subset of the original enterprise architecture), creating a *new* data source (i.e. an integrated data source, e.g. a data mart) or using the original enterprise architecture and data sources. Using the original sources may be risky; these sources should not be affected in any way. Possibly, a different data model is created, e.g. with different dimensions and optimized for business intelligence and other representation solutions. Creating a new enterprise architecture subset may simply be creating a copy of a selection of concepts (business processes, business functions, application components, etc.). Creating a new data source may be creating a direct copy of the most suitable data model to store data, e.g. a data mart. Determining the best data model depends on multiple factors to be taken into account by the consulting firm and the organization: both influence the eventual choice. The data model is created by a *function* somewhere in a tool, which we will refer to as the Model Builder function (Johnson et al., 2007).

Table 23. Description table of the 'determine data model' activity.

Phase III-a	***Determine data model***
Activity description	Determining (& creating) the model to be enriched (e.g. an EA or a DS).
Possible Actor(s)	Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Information to be able to determine the best choice for a model to be enriched with. For example, a match of concepts derived from the enterprise architecture and the determined data sources (mapping).
Usable techniques	Enterprise Architecture application techniques.
Output	A model (e.g. an enterprise architecture or a data source)

III-b Enrich data model

The data model that was created in the previous activity, either an enterprise architecture or a data model, needs to be combined (or enriched) with data from the other source. If an enterprise architecture was selected as the data model, it will be combined with data sources. If a data source was selected as the data model, it will be combined with an enterprise architecture. For the enterprise architecture the previously matched DS-subset will be used: the data is loaded in the EA, creating an 'enriched enterprise architecture'. For the data source the previously matched EA-subset will be used: (meta-)data is loaded in the DS. Enriching a data model may be done using query scripts that retrieve data and, provided the matching of concepts is given (e.g. using the previously mentioned 'Concept Match Library') store data in the new data model. Simply stated, the script uses the locations of data as an input and output source for retrieving respectively storing data. The result of enriching an enterprise architecture could be an EA with data in its properties (or attributes). The result of enriching a data source could be a DS with meta-data that describes the location of data in an enterprise architecture.

Table 24. Description table of the 'enrich data model' activity.

Phase III-b	***Enrich data model***
Activity description	Enriching the model with data. For example, enriching an EA-subset with data from a data subset (DS).
Possible Actor(s)	Consulting firm, (e.g. Enterprise Architect or Data Specialist).
Input	Data or meta-data (descriptive) derived from an EA-subset or a DS-subset
Usable techniques	Enterprise Architecture techniques, mathematical techniques, analysis techniques, etc.
Output	Enriched model

As discussed in our approach, there are two options for model-enrichment using both enterprise architecture and operational data. The first is to enrich an enterprise architecture model. The second is to enrich a data source model, for example a database or a data warehouse.

Enterprise Architecture Model-Enrichment

Enriching an enterprise architecture model is done on the level of 'properties' or 'attributes' of an enterprise architecture object. Data from an external data source needs to be mapped to an object (a concept or a relation in enterprise architecture terminology). Values of data may directly be related to an enterprise architecture object using 'profiles'. A profile in BiZZdesign Architect (a proprietary enterprise architecture suite) is a way of assigning attributes to a specific EA object. A profile is a script that may be loaded when using a specific EA. The script contains several property descriptions to which data may be linked, either manually or automatically. A profile is created using a scripting language, in this case a proprietary BiZZdesign Architect scripting language. An example of a 'cost profile' is illustrated as follows.

```
Profile Costs {
      assignable to AbstractCompound;
      real amount_SVU = 0;
      real amount_SVU_RS = 0;
      real amount_DVU = 0;
      real amount_GB = 0;
      real amount_FTE = 0;
      real total_costs;
      HIDDEN boolean costs_calculated = false;
      real percentage_costs;
};
```

The properties of such a profile are object variables that have values. Depending on what the needs are, i.e. which properties should be present in a profile, data may be appointed to each property at a *time*. This means that, depending on the moment in time, data is assigned to a property: for different moments in history or future, the profile may be used to show the respective data that belongs to that specific moment. This means a profile provides the meta data for different data values of the same sort, but different from each other due to the influence of time, e.g. cost changes in time. When defining a profile, the properties may be set to a value, e.g. a neutral value. For integer types, shown as 'real' in the script above, this may be a '0'. Depending on the scripting language and the property type, the assignment to a value is done. Data is stored in the enterprise architecture data model.

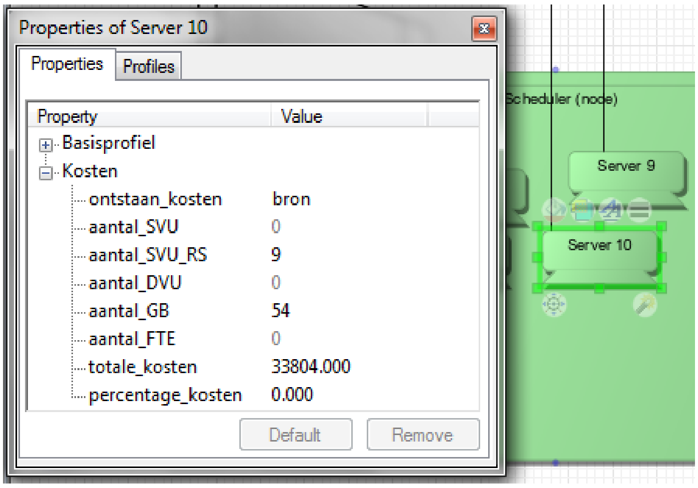


Figure 51. Profile and properties related to an object - an example.

Once the profile is set, it may be used for objects in an enterprise architecture. Depending on what is stated in the profile, the properties are shown for each object. The values in the example may be inserted manually. The values of a property may be linked to an external source using a script in order to automatically retrieve data and store it as a value to a specific property. Using properties, analysis may be performed, explained later.

Data Source Model-Enrichment

Enriching a data source model, e.g. a database or a data warehouse, is done using the same principles as used when enriching an enterprise architecture model. Again, data from a data source needs to be mapped to an object from an enterprise architecture (a concept or a relation in enterprise architecture terminology). Matching may be done manually or even automatically using a predefined script with matching logic. In any case, the logic that is needed for matching is initially determined by human interference. Automation may be applied when the logic is translated to a script. For further explanation, we use the simple manual matching of enterprise architecture objects and operational data using the exemplified mapping illustrated in Appendix C.

For operational data to be enriched with data from an enterprise architecture, we first consider a sample of data. The data is located in a single data source and is identified using a primary key. In order for data to be enriched, we need to identify a selection (or a subset) that is to be enriched with enterprise architecture data. Such a subset is illustrated in Table 25, operational data. The operational data subset should have a primary key, exemplified in Table 25 as the *record_id* attribute. If we use a model as indicated in Figure 49, we are able to use the *Concept_match* relation as illustrated and explained in Phase II of the EAIL. Using SQL to query, we are able to retrieve the data that describes the operational data and store it in new records. Possibly, the data source needs new columns to store the meta data. In Table 25, operational data is linked to an enterprise architecture object; more specifically, to a property of such an object. Being able to retrieve enterprise architecture data, we are able to determine descriptive meta data for the operational data. In Table 25, the meta data is the property *description* of an enterprise architecture object.

Table 25. An example of an operational data subset combined with enterprise architecture 'meta data'.

Enterprise Architecture		**Operational Data**		
property_id	*description*	***record_id***	*value*	*DS*
20	Total costs	200	5657	CRM database
20	Total costs	201	5353	CRM database
20	Total costs	202	56	CRM database
20	Total costs	203	353566.0	CRM database
20	Total costs	204	3535	CRM database
20	Total costs	205	12	CRM database
21	Weight	206	0.5	CRM database
21	Weight	207	1	CRM database
21	Weight	208	1	CRM database
21	Weight	209	0.7	CRM database

As shown in Table 25, the operational data subset is now 'enriched' with data from the enterprise architecture. Whereas previously only the values were known and the DS system, the data is now described with its meaning. For example, the *total costs* for a specific object. Using the Concept Match Library, we are even able to trace the specific object and use its descriptive data, e.g. the type of enterprise architecture object (business actor, business process, application component, device, etc.)

III-c Perform analysis

When either an enterprise architecture or a data source has been enriched, the enriched data model can be analysed upon. The analysis is performed by an 'analysis-function' (described by Johnson et al. (2007)) somewhere in an application (e.g. an enterprise architecture application or a business intelligence application). The analysis is done using algorithms with parameters. The algorithm is based on the *metric* and *variables* that were logged in the 'Metrics File', discussed in phase 1, explore. Several types of analysis may be performed, be it qualitative (using quantified measures) or quantitative. Measuring qualitative aspects may require phases like *preparation, coding, analysis* and *reporting* to make them measureable. Software applications like *HyperRESEARCH*, *QSR NVivo* and *ATLAS.ti* may assist in making qualitative data measureable. Information on how to measure qualitative data is explained by Hubbard (2010), amongst others. These applications may assist in phase I to determine variables and metrics. Quantitative analysis of data has been discussed in literature (e.g. M. Iacob & Jonkers (2006) on *EA-analysis*) and is already widely performed in all kinds of business intelligence applications.

Table 26. Description table of the 'perform analysis' activity.

Phase III-c	*Perform analysis*
Activity description	Performing an analysis on the enriched model.
Actor(s)	Consulting firm (e.g. Enterprise Architect or Data Specialist)
Input	An enriched model (e.g. an enriched enterprise architecture (EA-E) or an enriched data source (DS-E)) and a 'Metric File' to determine the analysis to be performed.
Usable techniques	Enterprise Architecture techniques, mathematical techniques, analysis techniques, etc.
Output	The calculated metric, outputted in an 'Analysis Data'-file (or any other file suitable for visualizing)

Enterprise Architecture Analysis

When an enterprise architecture has been enriched with data from operational data sources, there are different ways to perform an analysis on this 'enriched-enterprise architecture'. Performing analysis is a critical step in the EA-paradigm, since why having an enterprise architecture if it is not used for solving business problems? Performing enterprise architecture analysis adds insights regarding decision support. These insights may be used to concisely act upon a concern or any motivation an organization has.

Ways to perform analysis on Enriched-Enterprise Architecture

Enterprise architectures may be used to bring useful insights by either doing a *qualitative analysis* or a *quantitative analysis*. Both perspectives cover the entire spectrum of analysis on enterprise architectures. Qualitative enterprise architecture analysis is the interpretation of a view on an enterprise architecture, which may be supported by adding qualitative (e.g. descriptive) terms to the EA. For example, objects in an enterprise architecture may be added with descriptive information explaining the meaning of the objects in a specific view to clarify why they are shown. Quantitative enterprise architecture analysis is closely related to mathematics and involves formulas, calculations and quantitative data. Quantitative enterprise architecture analysis may be used e.g. to show bottlenecks in a process, time savings, cost savings, etc. (Schafrik, 2011). Both qualitative and quantitative enterprise architecture analysis should not be separated; they should be done both. They may be performed in sequence, e.g. by first performing a cost analysis (quantitative analysis) and secondly taking a look at the results to see where a bottleneck is indicated (qualitative analysis). The other way around, taking a look at the enterprise architecture to identify some key business goals (qualitative analysis) may be followed up by a cost analysis whether these business goals are fulfilled (quantitative analysis). Quantitative enterprise architecture analysis is done based on data, which e.g. may be provided by the organization the analysis is performed for. Usable data may be collected from operational data located in operational data sources, from results of consults performed by a third party doing bookkeeping, and many more. The data used should be integer, meaning the data can be trusted and is not manipulated in any way that affects eventual outcome for further analysis. Since analysis is part of the input for decision-making, it is critical that the data presented is sincere. The way enterprise architecture plays a role in decision-making is shown by the added value it brings: understandable business concepts are used to model the real world, which may help to understand the situation an organization is in. Since these models are understood by business, adding data to the enterprise architecture objects may bring a new dimension to data analysis in general. Now, organizations are able to directly map the data to the organizational objects (like business units, processes and more).

Data-tracing using Enriched-Enterprise Architecture

Performing analysis on complex enterprise architectures enriched with operational data may bring the desired results for a certain purpose the analysis was done, however, it may not always be clear where the data was retrieved from. Using the *Time* relation in the Concept Match Library, we are able to compare the *accessed* attributes for each concept match and see which have a minimal difference. Those attributes, e.g. that were accessed within a two-second timeframe, were most likely used in a single analysis. Comparing these values and highlighting them may show managers where the data was retrieved from. In Figure 52, we see the possible output of such a data traced enterprise architecture. The green highlighted EA objects show which objects were used for a certain calculation, like testing whether a certain goal has been reached.

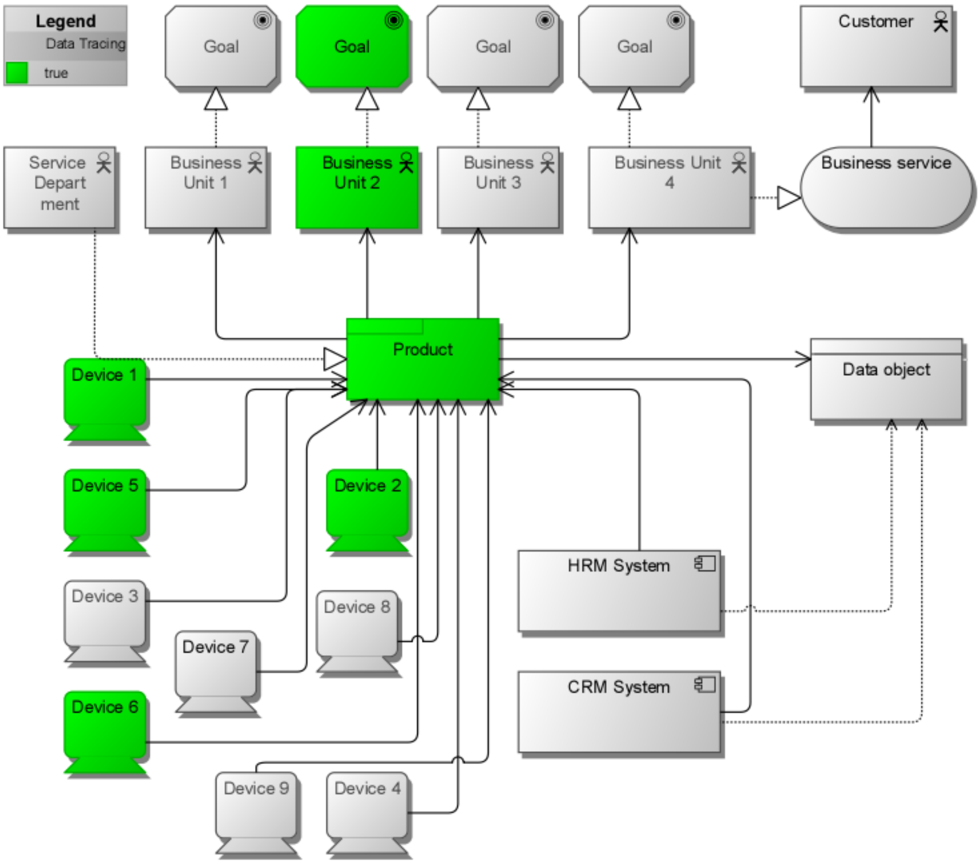


Figure 52. An example of a data traced enterprise architecture.

Data Source Analysis

In time, many different and sophisticated forms of business intelligence applications have been developed (Turban, Sharda, Delen, & King, 2010). These applications are able to show various different views on data through different forms of analysis. Like in enterprise architecture analysis, both qualitative and quantitative enterprise architecture analysis should not be separated; they should be done both. As explained, both forms may be performed in sequence, e.g. by first performing a cost analysis (quantitative analysis) and secondly taking a look at the results to see where a bottleneck is indicated (qualitative analysis). Figure 53 shows an example of a view on data using a business intelligence application. Here, the legend indicates different scales of data. If we would add meta data to this graph, like giving a meaning to each dimension, and adding information to each scale, the graph would give us more information. Likewise, if we would add enterprise architecture meta data to the graph, it would bring us more information. Information that is useful for accurate decision-making. The enterprise architecture meta data could be shown in a legend, using colours and different graphs (forms of representation).

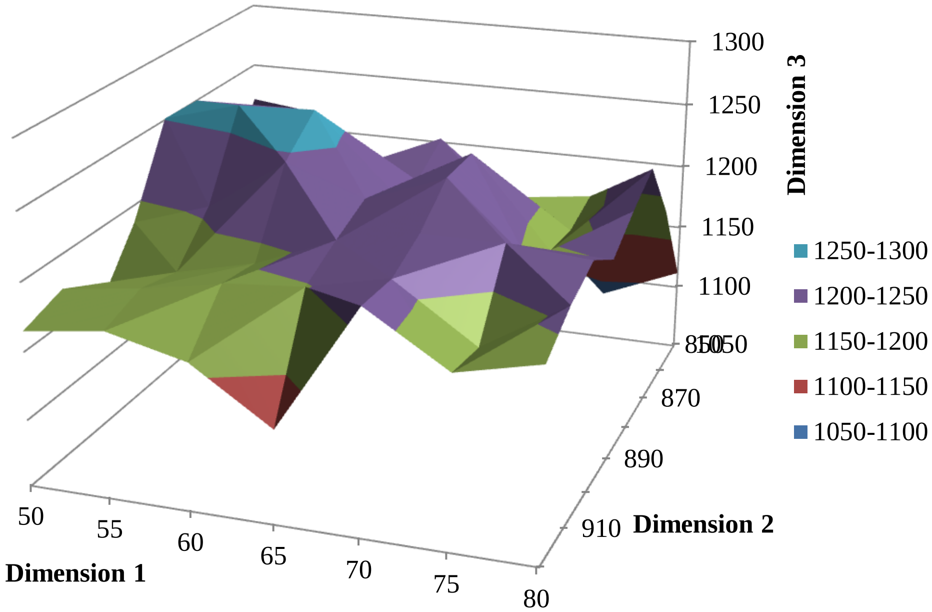
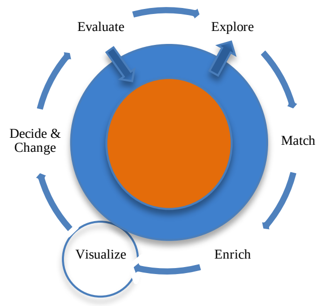


Figure 53. Example of a business intelligence view on data.

Phase IV – Visualize

In the fourth phase, *visualize*, we assume the organization has performed the previous activities in the EAIL. The fourth phase starts by performing the first activity, i.e. 'prepare data visualization'. Once the activity is performed, the next activity is initiated, until the phase ends.



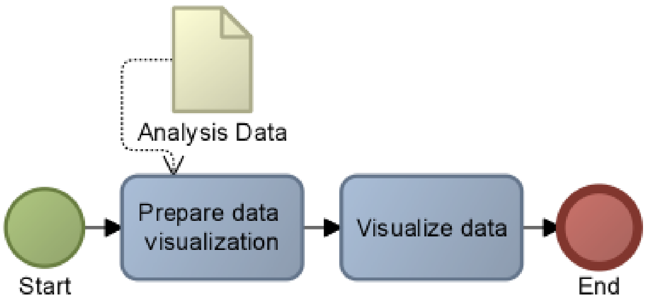


Figure 54. Processes in Phase IV - Visualize.

IV-a Prepare data visualization

When data has been analysed and made available for visualization, the dataset may be optimized for the software application that is able to represent the data in a way the decision-maker understands. Before choosing a software application, taking a moment to decide which way to represent the data would be wise. Referring to the personas we have described in the previous chapter, different types of decision-makers have different needs to be able to make thought-out decisions. These needs may determine the software application that needs to be selected. Data may then need to be reorganized and tailored to specific needs the selected software application requires the dataset to fulfil. Business intelligence suites often require a specific format like Microsoft Excel-sheets (.XLS) or Comma Separated Values (.CSV) and even different database type extensions. Enterprise architecture applications require the file to be formatted in the (often) proprietary extensions (e.g. .XMA for BiZZdesign Architect). Whichever extension is needed, the data should be prepared for the visualization to present it to the decision-maker. Preparing the data for visualization may only be necessary when a change is needed from one format to another. If a format is already optimized for the final representation, this activity may be skipped. The data should then be ready for visualization.

For example, a CIO may be very keen on numbers to act upon, since it gives him proof and a solid background for decisions to be made. The natural choice in the previous activity 'determine data model' would then have been to choose a DS due to its ability to 'slice and dice' upon data: performing different analysis forms and graphically show it using bar charts, pie charts and many more. In this case, the meta data provided by enterprise architecture, i.e. the descriptive data for the operational data previously lacking in normal business intelligence, gives the CIO the information needed he normally needs to figure out himself. The meta data could be shown in e.g. a legend; this saves time to figure out where the data is related to. Currently in business intelligence suites, this meta data is added manually where the data needs to be 'tagged' with meta data to show its meaning before the CIO interprets what he sees. Using a CML (Concept Match Library), meta data could be retrieved that corresponds to the operational data. A simple mapping between the two in an Excel spreadsheet could be enough to bring the context needed for the CIO: operational data about costs could e.g. now be put into the context of the business unit and its processes the operational data is related to, instead of having to figure out yourself. For example, in manufacturing businesses, it is critical to see the errors and trace down the problem in the assembly chain, amongst others. Simply showing the numbers on a dashboard and telling that ‘something is wrong’ does not solve the CIO’s problem. He needs to know how he can solve it, which means he needs to know where the problem is located and who is responsible for it. Using a CML, it is possible to monitor data for certain processes (possibly real-time, but this may cause a big load on applications) and directly see the bottlenecks using the enterprise architecture as meta data for the operational data. Depending on the level of granularity of processes, the KPIs set *and* the time to monitor the data, the problem may be directly traced.

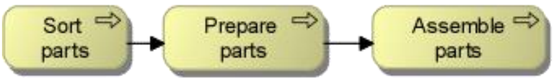


Figure 56. Example: manufacturing processes.

In Figure 56, an example of some manufacturing processes are shown that may take place when manufacturing a certain product. For optimization purposes, it may be interesting to see the corresponding data for these processes, possibly to gain insights in how these processes are performing. In Figure 55, we see an example of a BI solution (shown in Microsoft Excel). The data in the right column is the operational data. However, without the data in the left column, the operational data has no *meaning*: it is crucial for interpretation to show what the data tries to illustrate. The pie chart was added to see the effect. Visualizing the data is part of the next activity, ‘visualize data’.

EA Entity	Duration time (minutes)
Sort parts	3,2
Prepare parts	6,3
Assemble parts	2,1

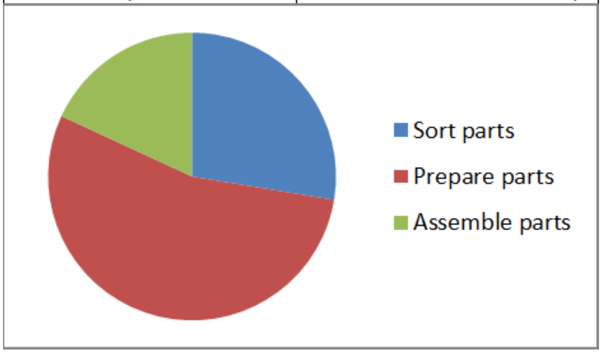


Figure 55. BI example: DS data (right column) with EA meta data (left column) and a possible pie chart to show differences.

Table 27. Description table of the 'prepare data visualization' activity.

Phase IV-a	***Prepare data visualization***
Activity description	Preparing the analysis-data, being the result from performing an analysis on the chosen data model with enriched data. For example, changing the model format from a Microsoft Excel (.XLS)-file to a Comma Separated Values (.CSV)-file.
Possible Actor(s)	Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Analysis data (e.g. from an ‘Analysis Data’-file)
Usable techniques	Business intelligence techniques (Chaudhuri, Dayal, & Vivek, 2011) (determining suitable input format for visualization application) (Kimball & Ross, 2013), SQL-techniques, etc.
Output	Prepared analysis-data

IV-b Visualize data

In this step, the dataset is ready for representation. Depending on the data model and the format the data was tailored to, a visualization application should be selected to represent the data in a manner the decision-maker understands. As we have elaborated on, different types of decision-makers exist (e.g. data-driven or structure-driven). The data that is shown should be understood (i.e. well-interpreted) by the decision-maker. For data visualization, literature about user interface design may be applied or best practices may be used as long as the organization is satisfied with the way data is presented. Priyanto (2013) explained different options that may be taken into account when developing for these business needs. The end result of this activity is data being visualized in a software application fulfilling the decision-maker's needs. It is important that the data being represented is trustworthy and not manipulated to give a better impression, since 'numbers are what they are'. The example of operational data and meta data, as illustrated in Figure 55, may be outputted in different ways, as explained by (Priyanto, 2013). Depending on the type of decision-maker, a different chart may be chosen for business intelligence solutions. In case an enterprise architecture was selected as the data model, different viewpoints with multiple output options may be chosen to represent the data best.

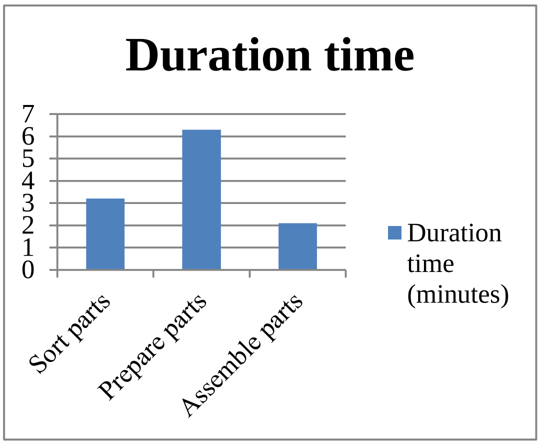


Figure 58. Visualized data: bar chart.

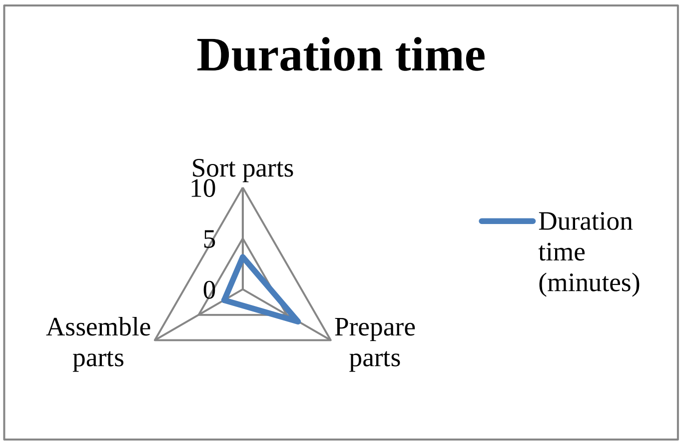


Figure 59. Visualized data: radar chart.

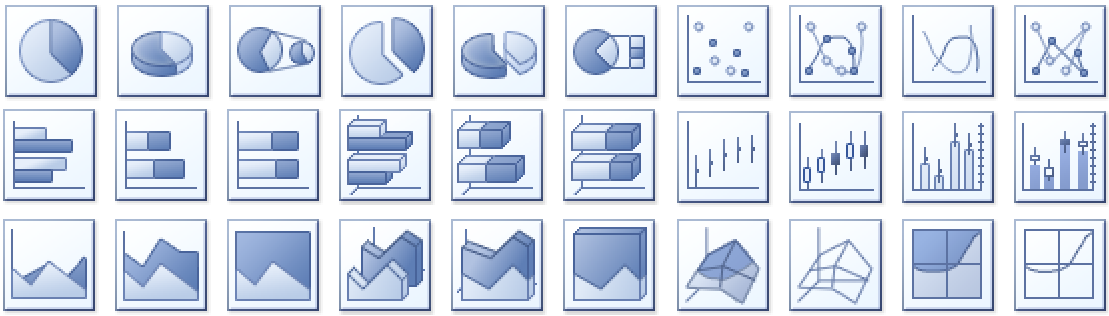


Figure 57. Different chart types in Microsoft Excel 2010.

The way data is represented has an effect on interpretation and therefore decision-making. It is thus important to select the right visualization of data for the right person and the right situation. It may be wise to get into contact with the decision-maker in order to understand its concerns and needs regarding decision-making. Lindström, Johnson, Johansson, Ekstedt, & Simonsson (2006) provided a survey of CIO concerns that may be addressed in decision-making. Using the research performed by Priyanto (2013), the concerns may be addressed by selecting the appropriate visualization of data.

Table 28. Description table of the 'visualize data' activity.

Phase IV-b	***Visualize data***
Activity description	Visualizing the prepared analysis-data, being the result from performing an analysis on the chosen data model with enriched data.
Possible Actor(s)	Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Prepared analysis-data (e.g. from an 'Analysis Data'-file)
Usable techniques	Business intelligence techniques (Chaudhuri et al., 2011), enterprise architecture techniques, scripting techniques.
Output	Visualized data

Phase V – Decide & Change

In the fifth phase, *decide & change*, we assume the organization has performed the previous activities in the EAIL. The fifth phase starts by performing the first activity, i.e. 'make decision'. Once the activity is performed, the next activity is initiated, until the phase ends.

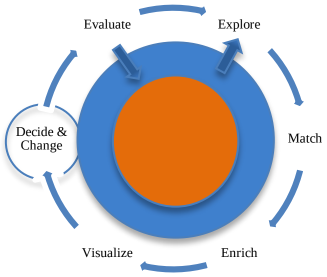


Figure 60. Processes in Phase V – Decide & Change.

V-a Make decision

Until now we have discussed the steps that have led towards the visualization of data. In phase V the organization makes a decision based on the represented data that leads to the implementation of a solution for the initial concern. Decision-making is done based on the data that was represented, interpreted by the decision-maker, which leads to information the decision-maker derives out of the data. The decision that is being made eventually leads to the development of a solution, which may be anything that solves a concern. The decision-making process may be complex and take different views on the data to be able to retrieve enough information out of the data.

As explained in Part III, Chapter 3.1, it is important to think about which 'signals' to use in order to bring the correct message. This is done in the previous phase. In the current phase, the decision-maker acts upon the data being presented, as explained in Figure 9. Here, the CIO (or other decision-maker) is the *receiver* and the person who decides upon which chart to use (e.g. a consulting firm) is the *sender*. The sender may choose different ways to visualize a dataset, as explained in previous activities.

If Figure 55 would be used for decision-making, the CIO sees that the business process 'prepare parts' takes a 'big slice of the pie' in the pie chart, meaning it may consume too much time with respect to the total amount of time to be spend on manufacturing a certain process. Without the meta data derived from EA, it is difficult to pinpoint the exact location of data, in this case possible problems regarding business processes taking too much time. There may be reasons why the time of the 'prepare parts' business process is higher, for example because it simply takes more time to prepare the parts than to assemble them. This is where common sense and interpretation mix; if the CIO has knowledge about the processes, he may know that it is normal that these business processes take more time. However, if he is not sure, he may need more data to be sure to interpret whether this is normal or a problem on which he should take action upon. Additional data may be given using different KPIs or just by asking a manager responsible for the process to explain the differences in time duration for each process.

In case an enterprise architecture is selected as a data model, the enterprise architecture may provide more information about the business processes, e.g. who the stakeholders are, what applications realize the processes and possibly have an impact on the duration time, etc. The CIO may want to know who is responsible for, or assigned to, the business processes. This is illustrated in Figure 61.

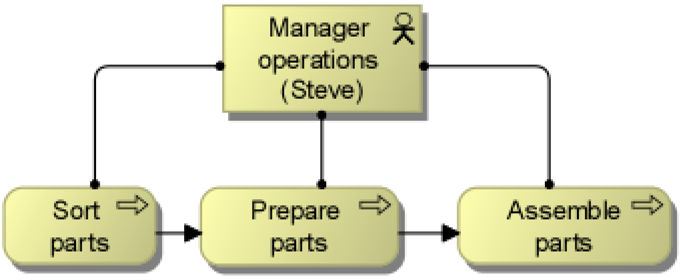


Figure 61. Business processes including the responsible manager.

Here, the manager operations could be asked to come and explain the reasons for the duration times of business processes.

The overall benefits of an enterprise architecture for decision-making are its structure, the possibilities to insert data and the option to design a future situation as a result of a decision, amongst others. The enterprise architecture helps to illustrate strategic decisions, e.g. by designing a target situation. Using different scenarios, a gap analysis may be done between different situations (called 'plateaus' in BiZZdesign Architect). Again, it is up to the person responsible for visualizing data to choose the right visualization for the decision-maker in order to support decision-making in a right way.

For business intelligence applications, meta data provided by enterprise architecture using a CML helps in pinpointing and 'tagging operational data' with understandable information derived from the enterprise architecture. It gives a context to data. For enterprise architecture applications, operational data gives evidence, i.e. measured data of pre-set KPIs (i.e. metrics). In combination with the structure of an enterprise architecture, several CIO concerns may be addressed and decisions may be made.

Table 29. Description table of the 'make decision' activity.

Phase V-a	***Make decision***
Activity description	Deciding upon a concern, based on the visualized data.
Possible Actor(s)	Organization (e.g. management, board of directors), (possibly assisted by) Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Visualized data
Usable techniques	Decision-making techniques, scenario comparison techniques (Johnson et al., 2007), etc.
Output	Decision

C1 Improve the quality of the interplay between the IT organization and the business organization, such as support, helpdesk, and end-user training
C2 Improve the quality of operation and maintenance, development, and acquisition of IT systems
C3 Decrease the costs related to the IT organization, such as wages and training for IT staff
C4 Improve the maintainability and modifiability, e.g. by improving interfaces, introducing middleware, and standardize protocols and products
C5 Improve the quality of the IT systems, such as security, performance, availability, reliability, quality of data, and correctness of functionality
C6 Provide new IT based solutions to the IT organization, such as administrative tools, b-logs, and back-up tools
C7 Provide new computer aided support to the business organization, such as new functionality, new information, and communication means
C8 Decrease the costs related to hardware and software
C9 Improve the quality of existing services or products that the business organization provides to the customers
C10 Develop new services or products that the business organization provides to the customers
C11 Decrease the cost related to the business organization, such as cost for personnel in the business organi-zation

Figure 62. Set of CIO concerns (Lindström et al., 2006).

Some of the most prominent concerns organizational leaders may have regarding IT are listed in Figure 62. Most of these concerns deal with strategy, i.e. long term decisions. They talk about future situations of the organization, while mentioning something about what is wrong with the current situation, like concern C5. Here, many KPIs are named, like ‘reliability’. But how is such a concern addressed? Using a dashboard via business intelligence solutions may give the CIO some tangible evidence, but where would the quality need to be improved (i.e. which systems)? Who are responsible for changes in quality? Who are affected by a choice (i.e. who are the stakeholders)? Will a decision impact other principles the organization has? And so on.

All of these questions may be answered by modelling an enterprise architecture (e.g. using ArchiMate including extensions with BiZZdesign Architect). The EA will bring insights and show cross-linked relations between business concepts decision-makers will understand, since the enterprise architecture is tailored to their organization. Information the decision-maker would have to have in mind without an enterprise architecture, which is difficult to manage due to the complexity of (large) organizations.

The other way around, if an organization tries to address concern C5 using only an enterprise architecture, it would have the structure and all the relations between business concepts in an overview, but it would miss the evidence of operational data that is logged in systems. This data shows undeniable history of the organization that might serve as proof of e.g. a malfunction in the organization, which is information an enterprise architecture alone will not bring.

Thus, combining both enterprise architecture, *to show context* of e.g. a concern, with operational data that *hands evidence* will improve the situation that currently exists, since both are currently separated. Looking at concern C5 in Figure 62 and using our method, we would be able to define metrics to measure ‘quality’ based on operational data that is available in the organization to provide evidence (Phase I of the EAIL), match it with the context provided by the enterprise architecture in Phase II, merging the two and performing an analysis on an enriched chosen data model in Phase III and determine a visualization of the result that suits the decision-maker and the situation in Phase IV. All of which is illustrated in Chapter 5.2. Likewise, the EAIL may be walked through, e.g. for concern C9 where the CIO decides to ‘go green’ and implement sustainability in its organization. This might mean that only sustainable semi products and materials are bought and only renewable energy is used to create the final product, e.g. a car.

Now, the decision-maker has evidence and has the context of e.g. a concern combined. Without the combination, it would have to perform both enterprise architecture and business intelligence support separately possibly causing misinterpretation of the situation with errors in decision-making, e.g. if the wrong decision is made that has an effect on the solution that is developed in activity ‘V-b – develop solution’.

V-b Develop solution

Having decided upon a certain concern, the organization is able to take action in order to minimize the concern. The result of an action for a specific concern is a *solution* to that concern: it tries to solve or minimize the problem. Multiple solutions may exist to a specific concern, all attempting to assist in heading to a better situation for the organization. Determining the best solution may need different opinions and scenarios to take into account, as mentioned by Johnson et al. (2007). A solution may be a set of actions to be taken, resulting in e.g. an action plan, but may also be a product or a service that needs to be created.

For a different concern C9, listed in Figure 62, a CIO may decide to develop a sustainability program to increase the quality of his product, e.g. a car. The enterprise architecture was used to model a new principle, namely ‘only use sustainable energy and materials’ and stakeholders were modelled that may have a stake in this principle. By performing a ‘customer satisfaction program’, the manager operations will retrieve information about the opinions of employees. Meanwhile, operational data may be used to show that business processes ‘prepare parts’ and ‘assemble parts’ are still not sustainable, indicated using a ‘colour view’ showing the percentage of renewable energy used that is not sustainable in the colour red. The CIO hopes that these processes will be sustainable after changing the way procurement is currently done. He will know using both operational data and enterprise architecture, outputted in an enterprise architecture data model as shown in Figure 63. Colour views using both ‘red’ and ‘green’ will indicate how things are performing.

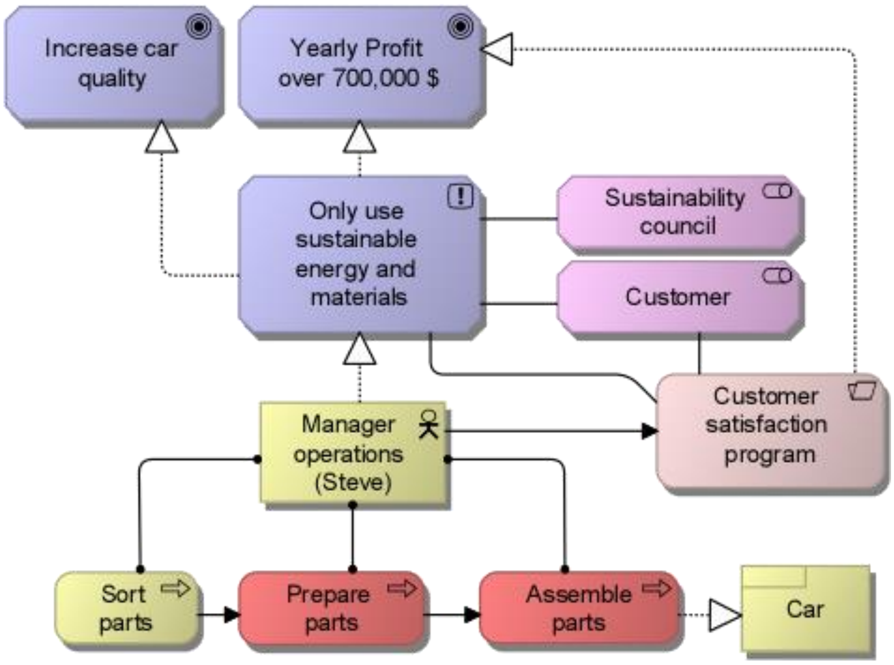


Figure 63. Example: implemented decision, shown in enterprise architecture.

Table 30. Description table of the 'develop solution' activity.

Phase V-b	***Develop solution***
Activity description	Creating a solution based on the previously made decision (e.g. a set of activities to be performed by personnel on tactical or operational level).
Possible Actor(s)	Organization (e.g. tactical or operational personnel) or consulting firm
Input	Decision
Usable techniques	e.g. tactical and operational skills for creating the most suitable solution for the current situation
Output	Created solution

V-c Implement solution

When a solution has been determined and developed, the organization should implement (or execute) the solution in the organization. Before a solution is implemented, it may be tested in a *test-environment* to be able to predict its behaviour in the *real environment*. A test-environment may be a representation of the real environment or even an extreme environment to be able to see how the solution reacts to extreme situations. When the behaviour of the solution is acceptable, the solution may be transferred to the real environment.

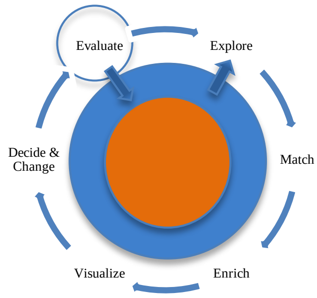
For example, looking back at the situation previously explained about concern C9 (cf. Figure 62), the organization may realize one type of car with sustainable parts to see whether the advantages (of possibly more sales) weigh up to the disadvantages (of possibly higher procurement costs). Using the customer satisfaction program shown in Figure 63, the organization hopes that goals are reached. Time will tell whether the decision was beneficial to the company, which is evaluated in the next phase.

Table 31. Description table of the 'implement solution' activity.

Phase V-c	***Implement solution***
Activity description	Implementing a solution previously developed (e.g. a set of activities to be performed by personnel on tactical or operational level).
Possible Actor(s)	Organization (e.g. management, board of directors), (possibly assisted by) Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Created solution
Usable techniques	e.g. tactical and operational skills for implementing the solution in the best way for the current situation
Output	Implemented solution

Phase VI – Evaluate

In the sixth phase, *evaluate*, we assume the organization has performed the previous activities in the EAIL. The sixth phase starts by performing the first activity, i.e. 'prepare data visualization'. Once the activity is performed, the next activity is initiated, until the phase ends.



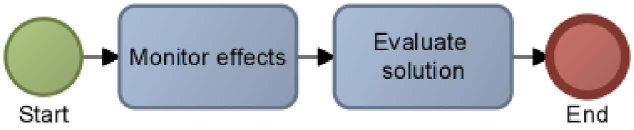


Figure 64. Processes in Phase VI - Evaluate.

VI-a Monitor effects

The solution being implemented in the organization does not necessarily mean the organization is relieved from the 'pain' the concern was bringing initially. It may be that the solution has helped in minimizing the concern, but did not manage to take eliminate the concern. To be able to determine whether a solution has helped in minimizing the initial concern means that the concern and solution should be monitored upon the effects the solution has brought to the organization. These effects may have come at a certain point and disappeared at a later point in time. This is the reason why effects need to be monitored for a period of time; the effects may not be permanent, making a final evaluation of the solution based solely on a 'snapshot' of a situation not fair. However, the effects should be taken into account when discussing the positive and negative effects of the solution. Examples of effects are *customer sales increase/decline* or *business leads increase/decline*.

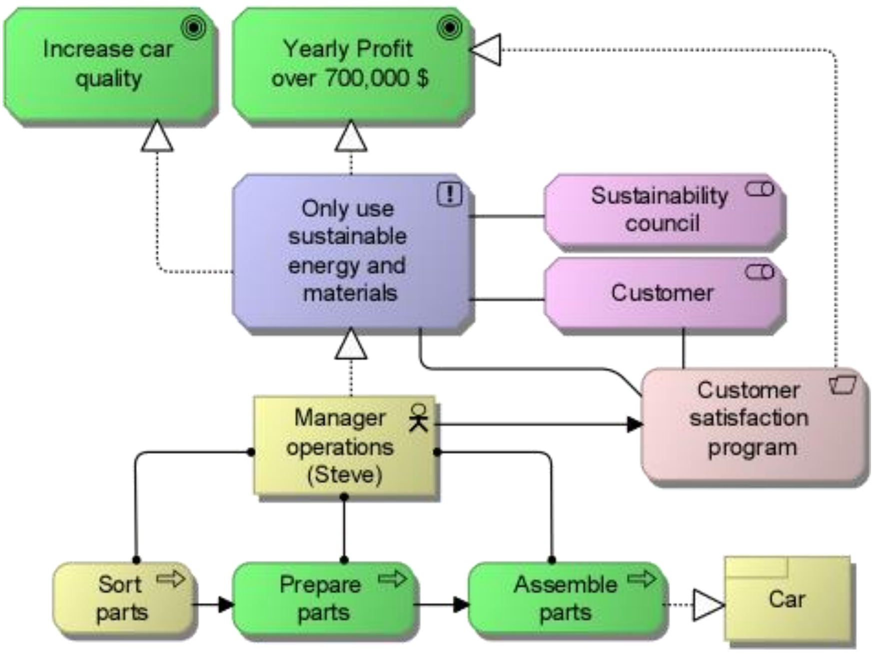


Figure 65. Example: monitored effects, shown in enterprise architecture.

Looking back at the concern C9 (cf. Figure 62), the effects of the decision made by the CIO may be measured over a period of time, e.g. a year. In Figure 65, the CIO sees that both the car quality has improved by using renewable energy and sustainable materials, while noticing that his business processes that were monitored (‘prepare parts’ and ‘assemble parts’) are also sustainable.

Table 32. Description table of the 'monitor effects' activity.

Phase VI-a	***Monitor effects***
Activity description	Monitoring the possible effects occurring as a result from implementing the solution for a period of time.
Possible Actor(s)	Organization (e.g. management, board of directors), (possibly assisted by) Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Implemented solution
Usable techniques	Business intelligence techniques (Chaudhuri et al., 2011), enterprise architecture techniques, scripting techniques.
Output	Results of monitoring effects

VI-b Evaluate solution

When the situation has come to an equilibrium, which is a stable moment when most temporary effects have appeared, the solution and situation are ready to be evaluated. Here, the organization and consulting firm discuss the way the solution has (hopefully) helped in minimizing the concern. The developed solution may completely solve the initial concern, but may also solve a part of it. When the organization decides that the concern should be minimized even more, the approach starts again in phase 1, explore. The new situation becomes the ‘current situation’ and the cycle starts all over, in a new quest for minimizing the (set of) concern(s).

Evaluating upon the concern C9, the CIO sees that customers are now satisfied and that car quality has improved, while realizing profit goal (cf. Figure 65). The decision has proven to be a success.

Table 33. Description table of the 'evaluate solution' activity.

Phase VI-b	***Evaluate solution***
Activity description	Evaluating whether or not the concern was solved, partially or entirely, e.g. using results of monitoring effects.
Possible Actor(s)	Organization (e.g. management, board of directors) and consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Results of monitoring effects
Usable techniques	Decision-making skills, evaluation skills
Output	Evaluation

Without using operational data and enterprise architecture, it would be difficult to realize the same amount of information all at once. Surely, if a lot of effort is put to fill in the gap that business intelligence has, e.g. by using many advisors that bring the information of an enterprise architecture, the CIO may come up with a same solution to his concern. Likewise, by providing the CIO with operational data, while looking at his enterprise architecture, the CIO may also come to the same solution. If the situation is automated, like in our examples, this might save time and money in the long term.

5.2 Approach I – Enterprise Architecture Enrichment

Description & Illustration – ArchiSurance Case

First of all, we have to assume an organization *requests* us, the consulting firm, to help. Assuming we don't know anything about the organization's situation, we first need to have a conversation with the stakeholder (which may be the organization's management) to discuss the *concern* about the current situation. Such a concern may be anything, there might not even be a real problem; e.g. a request for more information is also considered to be a concern (since the organization might feel they do not have enough information currently or might think .

Phase I – Explore

Activity I-a Determine concern

Let's say we are talking with the CIO of ArchiSurance who has a concern that he wants 'to decrease the costs related to the organization' (e.g. *costs for personnel in the organization*). The CIO is basically saying he does not like the ***current situation*** and wants to head for a ***better situation***. Surely, the CIO has thought of why he wants to cut on costs, but we don't know *why* he wants to cut on costs. Therefore, in order to be able to achieve a better situation for the organization, we need to know what the real problem is (e.g. by using a *root cause* analysis to determine the actual cause). After having had some interviews and informal talks with the CIO, we have determined that the root-cause is not *decreasing costs for personnel in the organization*, but *stakeholder satisfaction* (modelled as a driver in (Jonkers et al., 2012)). Using a root cause method (Rooney & Vanden Heuvel, 2004), e.g. by asking questions to encounter a more specific concern, we have encountered that *profit* is a more refined concern of the general *stakeholder satisfaction* concern (Jonkers et al., 2012). Having had some discussions with the CIO, we encountered that he was interested in graphic representations and particularly supports the *enterprise architecture* paradigm, which may point out the CIO is a *structure-driven* decision-maker.

Table 34. Description of the 'determine concern' activity (ArchiSurance).

Phase I-a	**Determine concern**
Actor(s)	CIO and Enterprise Architect (consultant).
Input	Business request of ArchiSurance.
Used techniques	Root cause analysis (Rooney & Vanden Heuvel, 2004), interview techniques (Kvale & Brinkmann, 2009).
Output	Determined concern ('stakeholder satisfaction')

Activity I-b Create source overview

In any case, after the concern or the root cause has been determined, we will try to figure out how we can create a solution by first *exploring* the current state (baseline) of the organization. The consulting firm should be able to access the enterprise architecture to see the structure of the organization and get an idea where the problem might be located, be it a small part of the organization like a division or the entire organization. Then, the consulting firm should be able to get an *overview* on the data and data sources that are located in the organization to support his thoughts (**I-b**). For now, the overview does not yet have to be very specific, however, it should be specific enough to get a global idea on how we might capture or measure the concern. The information for the overview may be retrieved from an 'Application Co-Operation View', as shown in Figure 11 in the ArchiSurance case (Jonkers et al., 2012). The type of data may be identified creating a 'Data Dissemination Diagram' for an ArchiSurance Application (Figure 14, (Jonkers et al., 2012)).

Table 35. Source overview (ArchiSurance).

Source name	Type of data
Home & Away policy administration	Policy data/ docs
Home & Away financial administration	Financial data
Document Management System	Common Files/Docs
Auto Insurance Application	Car insurance data
Legal Expense Back-Office System	Legal
Call Centre Application	Phone call data
Web portal	CRM data, reports
General CRM System	CRM data
Legal Expense CRM System	CRM data (legal expense)

Table 36. Description of the 'create source overview' activity (ArchiSurance).

Phase I-b	**Create source overview**
Actor(s)	Enterprise Architect and Data Specialist.
Input	Determined *concern*: 'stakeholder satisfaction'.
Used techniques	Interview techniques (Kvale & Brinkmann, 2009), requirements analysis techniques (Hay, 2011).
Output	An overview of available data sources and a description of data stored in these data sources.

Activity I-c Retrieve variables

When a global view has been created, we will try to find out some core elements (*variables*) that we might be able to measure regarding our concern *profit*, like prices of products/services, number of products/services sold, and so on. Finding the variables may be done in talks with, in this case, a financial expert. Having determined a set of available and interesting variables to measure, we now know what we are able to use for determining a certain metric. The encountered variables are listed in Table 37.

Table 37. Some variables to build a metric (ArchiSurance).

Variable name	**Type**
Products sales per year	Number (integer)
Product price	Number (Money)
Maintenance costs per year	Number (Money)
Number of employees	Number (integer)
Average salary costs per employee per month	Number (Money)
Total yearly revenues	Number (Money)
Salary costs per employee per year	Number (Money)

Activity I-d Determine metric(s) & measurement(s)

Having encountered the listed variables, we need to think what we can do with them, perhaps combining them into a metric. As we can see, the variables state something about ‘revenues’ and about ‘costs’. Looking back at our concern, ‘profit’, we might be able to determine a metric called ‘personnel costs’ or a more generic metric called ‘costs’. For the latter, we could add the ‘maintenance costs per year’ and determine a metric based on variables that we have. For accuracy, we will use the ‘personnel costs’ as a metric for filling in the template provided by Priyanto (2013). This template may provide input for later phases. The information currently inserted in the template, stated at the *end of this case*, is shown in blue.

When the metric has been determined, we need to determine at first *how* we will do the analysis step, that is *how to measure the metric*. Determining which type of analysis to perform may be complex, since we may need to determine a calculation ourselves or select a type of analysis already explained in literature or elsewhere. For our metric, ‘personnel costs’, we will use the variables stated in the ‘Metric File’, which are ‘*number of employees*’, ‘*average salary costs per employee per month*’ and ‘*salary costs per employee per year*’. As we can see, there are two ways to calculate the total personnel costs per year, i.e. there are two ‘ways to measure’ the same unit. The first is to multiply the variable ‘number of employees’ with the variable ‘average salary costs per employee per year’. The second is to make a sum of the variable ‘salary costs per employee per year’ for each entry. A choice is made based on rationale; choose the option that is best supported (e.g. a calculation method is already known). When considering a calculation method, we need to think about the way the calculation may be performed (e.g. top-down, bottom up (Iacob & Jonkers, 2006)) and which set of data will be needed to do the calculation with. To do so, the consulting firm may have a talk with someone inside the organization that is familiar with the data we are searching for.

Table 38. Description of the ‘determine metric(s) & measurement(s)’ activity (ArchiSurance).

Phase I-d	**Determine metric(s) and measurement(s)**
Actor(s)	Enterprise Architect and Data Specialist.
Input	Retrieved variables (cf. Table 37).
Used techniques	Metrics design (Hubbard, 2010; Kerzner, 2011), requirements analysis techniques (Hay, 2011) and algorithm design techniques (Skiena, 2008).
Output	An identified *metric ’personnel costs’* and its *measurement*.

Phase II – Match

Activity II-a Determine EA & DS subsets

The organization may bring the consulting firm into contact with one of their 'data specialists'. Then, the 'Source overview File' is used as an input for the data specialist to determine the data source (DS) subset of applications he would advise to use for calculating the total personnel costs. The consulting firm should take a look at the enterprise architecture to determine the concepts that may be related to the DS subset or where the DS may provide data for. The DS subset may be a combination of the 'Source overview File' and the determined variables, as shown below in Table 39. The DS subset illustrated here has data *about* the EA subset. The data *inside* the systems in the DS subset may be matched with objects (i.e. concepts and relations) in the EA subset.

Table 39. The DS subset (ArchiSurance).

Source name (target situation) – Possible DS-subset	**Variable name (searched for & found in sources)**	**Type**
ArchiSurance Back Office Suite	Number of employees	Number (integer)
ArchiSurance Back Office Suite	Average salary costs per employee per month	Number (Money)
ArchiSurance Back Office Suite	Salary costs per employee per year	Number (Money)

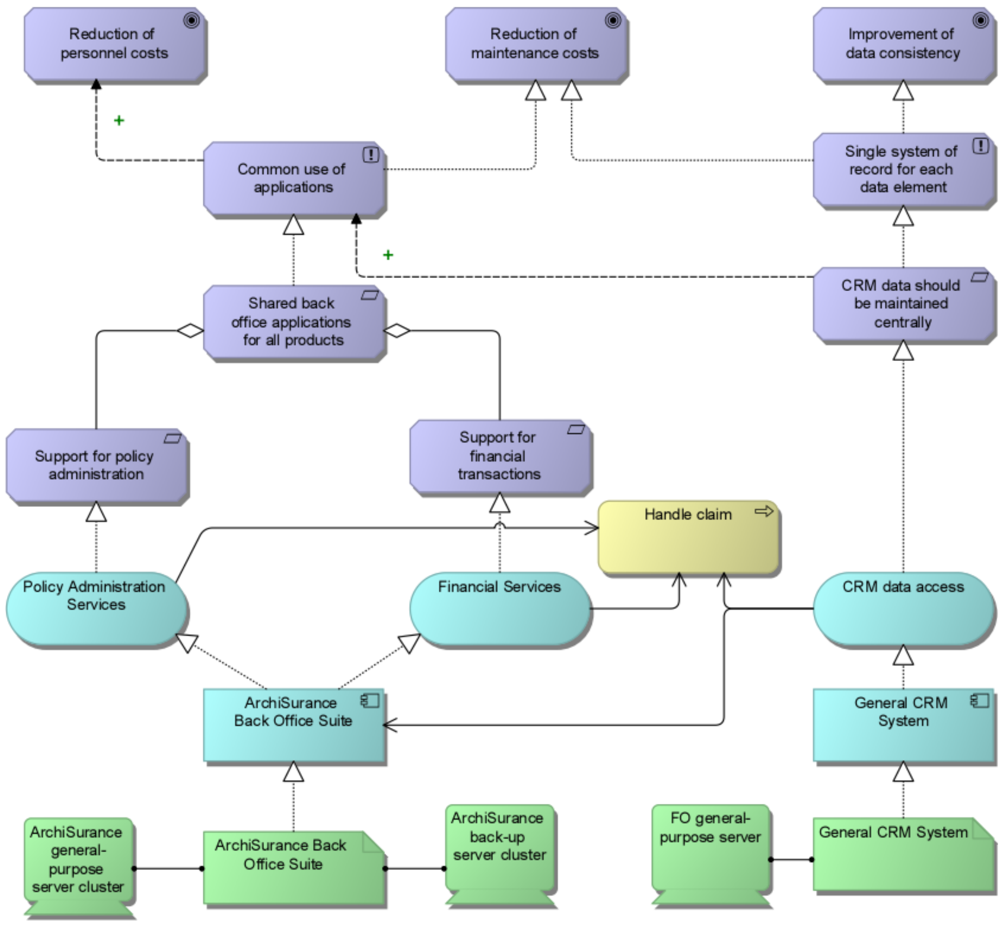


Figure 66. The EA subset (ArchiSurance).

While being in contact with the data specialist, we had a look at the organization's enterprise architecture. The enterprise architecture is a large overview of all concepts and relations related to the ArchiSurance's entity. However, not all of these concepts and relations are useful for our case. Therefore, we made a selection large enough to capture both the goals and the systems, but small enough to keep an overview and for it to be enriched with data from a DS. Some parts of the EA subset were kept to keep the structure and the overview on the case, meaning not all elements in the EA subset may necessarily be enriched with data, but may be intended to maintain structure.

Activity II-b Match concepts

Based on these *subsets*, *talks* with the data specialist and *knowledge* on how to calculate with values, we may add new information to the 'Metric File' and thereby update it (shown in orange). The EA-subset and DS-subset can be matched as follows: the data found in the DS-subset may be identified with the system they are stored in, the *IBM WebSphere Application Server*. This system is represented in the EA-subset as a node (device), the *ArchiSurance general-purpose server cluster*. This is what we call a '***concept-match***'. A concept-match may be represented as follows.

Table 40. An illustration of concept matches (ArchiSurance).

Matched Concepts			
Source name (DS-subset)	**EA-component (EA-subset)**	**Variable name (in DS) (variables)**	**Variable type**
IBM WebSphere Application Server	Financial services (application service)	Number of employees	Number (integer)
IBM WebSphere Application Server	Financial services (application service)	Average salary costs per employee per month	Number (Money)
IBM WebSphere Application Server	Financial services (application service)	Salary costs per employee per year	Number (Money)

Now the activities in phase II have been carried out, the next thing to do is to use the found data and insert it in a data model for further analysis.

Phase III – Enrich

Activity III-a Determine data model

As stated in the beginning of this case, having talked with the CIO of ArchiSurance gave us an idea what to provide as a suitable data model for decision support. Since the CIO support the enterprise architecture paradigm, we propose to do an analysis with the enterprise architecture as the primary structure: this will be our data model. Ideally, we should use the original (real) enterprise architecture currently used by the organization. Since the next step, *enriching the enterprise architecture*, may not be without risk, we discussed the possible risks and determined that the risks may be neglected if we first do a try out on a small copy of the enterprise architecture. Having performed a small illustration how this may be done, we created a direct copy of a part of the original enterprise architecture, due to the small area we will be working with.

Activity III-b Enrich data model

Enriching an enterprise architecture requires knowledge of scripting for importing data and analysis. Some enterprise architecture may provide a proprietary scripting language. The enterprise architecture scripting language we use is a proprietary BiZZdesign scripting language. We converted Excel-files to CSV-files for easy importing in our application. A CSV-file import was done using the scripting language. The Excel-data was given to us by the data specialist we were closely working with. The concept-match overview was used to link *columns* in the CSV-file (DS) with *properties* that were assigned to the respective enterprise architecture objects (EA) using the 'profile'-functionality in BiZZdesign Architect.

Activity III-c Perform analysis

Having inserted this operational data in the enterprise architecture, we will do an analysis using the enterprise architecture. As shown in the 'Metric File', a bottom-up analysis is performed (as explained by Iacob & Jonkers (2006)) and use the calculated value of the 'total yearly personnel costs' to compare it to the 'target goal' with a given value of 100.000 dollar (assuming this is a given target derived from management) that may be assigned to the 'reduction of personnel costs' goal. As shown in the figure below, a trace between the enriched concept 'ArchiSurance general-purpose server cluster' and the goal 'reduction of personnel costs' is drawn and highlighted in green. Following the criteria described in the 'Metric File', the goal is either reached or not, depending on the value of the 'total yearly personnel costs'.

> The steps we have followed until now and the results from these steps could have been done far easier using a simple calculation, however, the next step shows the added value of an enterprise architecture.

We assume the organization performed well and has reached their goal. Every month, the data in the 'ArchiSurance general-purpose server cluster' is renewed and an assessment is done based on the target goal. The analysis result for this case *is the assessment whether the goal 'reduction of personnel costs' has been realized by the organization based on data retrieved from the 'ArchiSurance general-purpose server cluster'*.

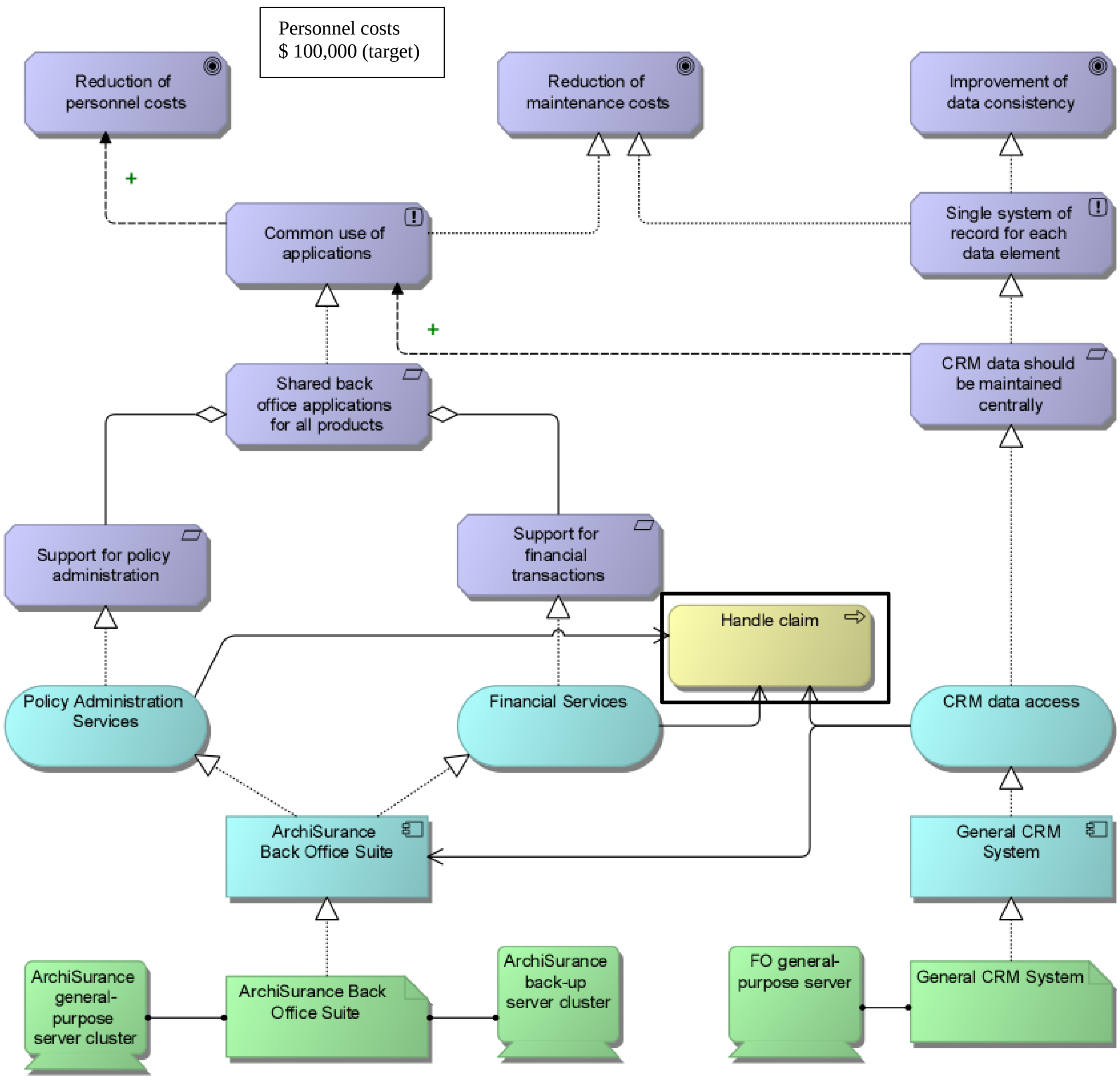
Phase IV – Visualize

Activity IV-a Prepare data visualization

When the analysis has been performed, the result should be visualized in the model that was previously determined, in our case an enterprise architecture. Depending on the visualization application that is used, we might need to prepare the analysis data for this application. In our case, the analysis result is being outputted in e.g. a CSV-file (Common Separated Values).

Activity IV-b Visualize data

Using scripts, we may be able to plot the result (**visualize the data**) of our calculation and the target, as illustrated below. In Figure 67, we see how the reduction of personnel costs is realized. With the data retrieved from the *ArchiSurance general-purpose server cluster* (which is an *IBM WebSphere Application Server*), we are able to compare the goal 'reduction of personnel costs' with the actual financial numbers.



Personnel costs
$ 111,656

Figure 67. ArchiSurance goal realization.

Comparing these numbers means looking at a *snapshot* of the situation; a moment in time. However, the situation changes as ArchiSurance operates (since personnel keeps on working and creates costs). It would be unfair to measure whether a goal was reached on the wrong moment in time. Agreements on when these goals are due date should therefore be made, which is when a final data retrieval should be done. Together with ArchiSurance, a date was set to April, 18th. The comparison of values is done using a single attribute called *personnel costs* appointed to all enterprise architecture objects where this type of costs incurs. The actual numbers were, as explained, loaded in the EA. The situation on May, 18th is visualized in Figure 67 and Figure 68. The goal *reduction of personnel costs* has not been reached yet. Here, the EA objects that significantly changed in a negative sense regarding the last time data was fetched are shown. We see that the business process *'handle claim'* is involved as well.

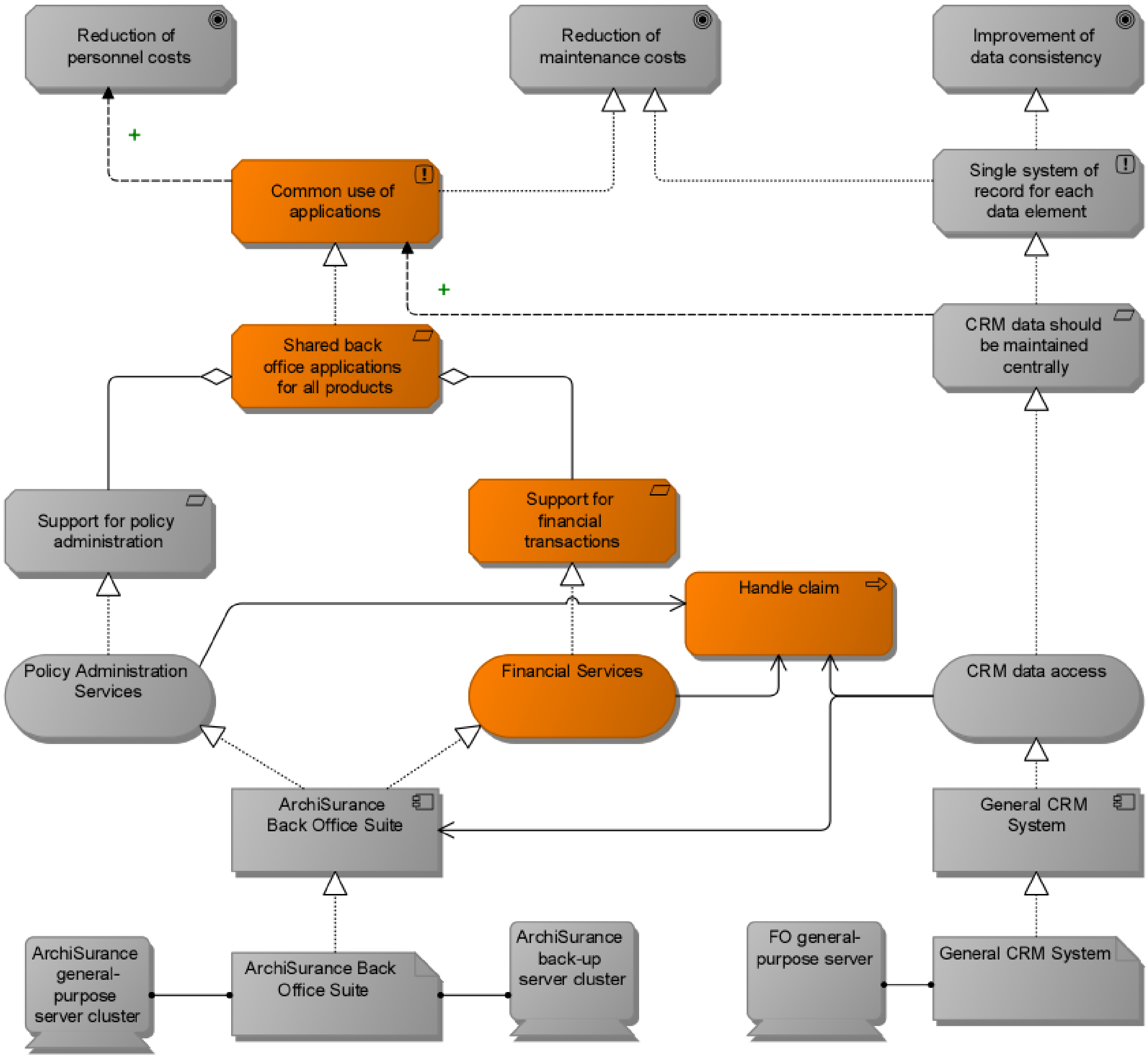


Figure 68. ArchiSurance situation on March, 4th.

The business processes that are involved for the current situation are shown in Figure 69. Here, a visualization has been made regarding the relevant business processes that significantly changed in the past period. The business process ‘handle claim’ is highlighted, which may mean something has happened in the past period.

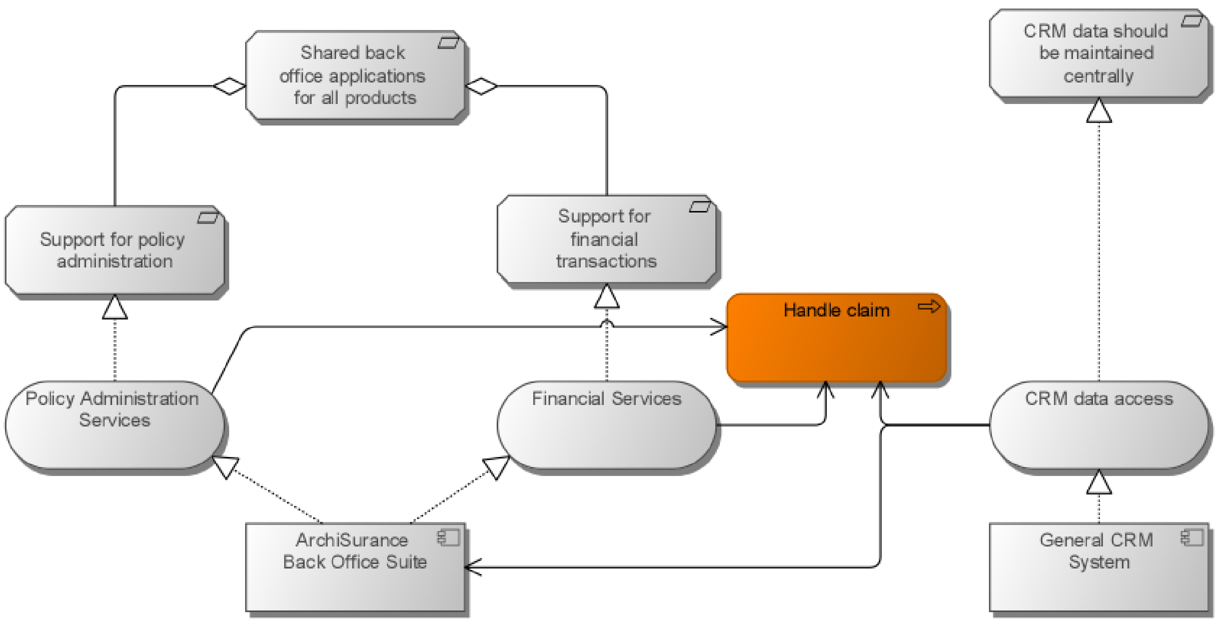


Figure 69. Business processes involved in the situation on March, 4th (ArchiSurance).

It is up to management whether they want things to be unchanged or to make a decision to change things in the organization. The indicated process is just a visualization of something that may have had an effect on the goal not being achieved. The management may want to look deeper into this process, and, depending if the Concept Match Library was set to keep track of these processes as well, the processes that are nested in 'handle claim' may also be analysed (cf. Figure 70).

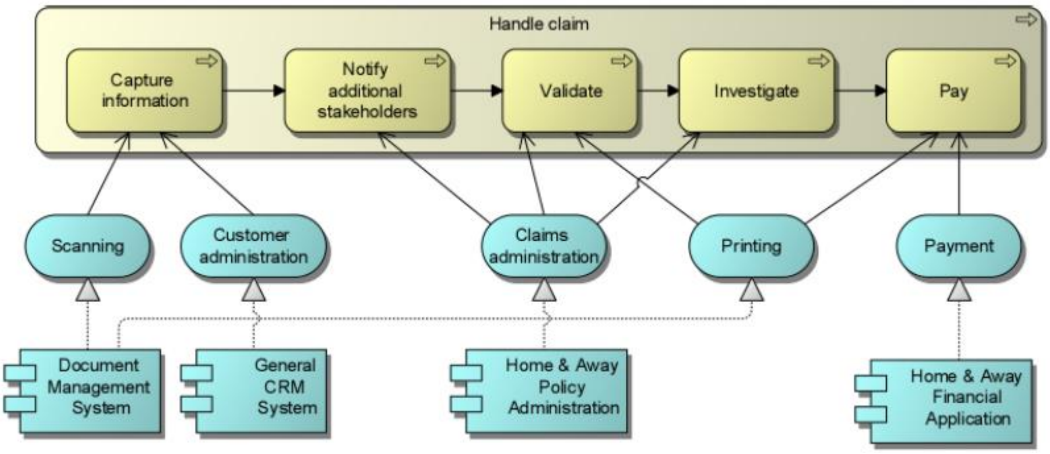


Figure 70. A closer look to the 'handle claim' business process (ArchiSurance).

Phase V – Decide & Change

Activity V-a Make decision

Now the analysis data has been visualized, showing that the target was not achieved as illustrated above, the organization could ***decide*** to change something in the organization. For example, they might take a look at their business process 'handle claim' and see if something may be improved to lower personnel costs. Also, they may just wait and see if the situation improves in the next couple of months. ArchiSurance decides to wait and see what happens in the near future because they find it risky to change the business processes; the business processes are considered to be critical to customer satisfaction. They do, however, keep on fetching data from their system to feed the Concept Match Library.

With only an enterprise architecture, it is difficult to obtain evidence indicating that something could be improved. The idea may exist, but there is no proof indicating that something needs attention, for example when a goal has not been met. Such a concern could be measured and monitored using metrics. With operational data as a result of measuring metrics, there is proof of events happening in the organization. Without operational data, decisions would be made based on intuition. Without knowing where the concern takes place in the organization, there is no focus area; decisions could be done in the wrong part of the organization, having an effect on organizational entities that should not have been addressed. Enterprise architecture shows where the 'pain' is, i.e. the location the concern is referring to. In our case, ArchiSurance has combined both worlds and has proof that things are happening because it has measured the predefined metrics. Moreover, it knows where to implement the solution because of its enterprise architecture with visualized analysis based on the data measured.

Activity V-b Develop solution

Having decided to wait and see if the situation improves can be seen as the ***solution*** the organization thinks what might solve this concern. If ArchiSurance had decided to change the business process, the artefact needed to solve the concern is considered to be the solution. This could be a manager telling the people involved in the business process to work harder, to get more work done in less hours. The solution could then be modelled in the enterprise architecture.

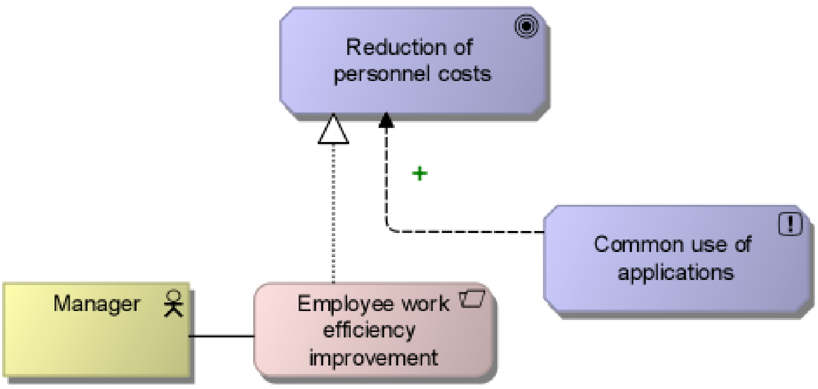


Figure 71. Example - Developed solution for ArchiSurance.

In more complex situations, ArchiSurance might even want to hire a change consulting firm to develop another solution that lowers the personnel costs.

Activity V-c Implement solution

To go back to our initial concern, we assume the organization has measured the numbers over a three months period and has ***implemented a solution*** (i.e. 'doing nothing' in this situation). If the target would still not have been achieved (e.g. by not being allowed to lower these wages due to legislation), the organization may want to consider the options they have.

Another option, as explained in the previous activity, would be that ArchiSurance implements the program 'employee work efficiency improvement', as illustrated in Figure 71. A manager would then be told to monitor the employees in the time they spend on tasks. If efficiency goes up, this may mean that some employees need to find other jobs, since they are no longer needed. This would lower personnel costs.

Phase VI – Evaluate

Activity VI-a Monitor effects

The consulting firm so far has been involved until the visualization phase ended. In the previous phase, 'decide & change', the organization decided, created a solution, which they implemented in order to solve their concern. As an evaluation step, e.g. to check whether their consult was useful, the consulting firm may want to stay in contact after their job 'visualizing the data' ended. In this last phase, the effects of the implemented solution are evaluated. The effects may be monitored by either the consulting firm (in a different setting), the organization itself or anyone else. Monitoring the effects of a solution could be done using the analysis previously discussed (e.g. calculating values every month and visualizing them in an application), but may also be done after a solution was implemented. The effects that were monitored may simply be monthly results for the calculated metric (calculated results). Based on the calculated results, forecasting may be done by recognizing the historic data and estimating the future results (e.g. based on average differences).

Table 41. Calculated and forecasted results - personnel costs (ArchiSurance).

Date	***Target***	***Calculated result (history)***	***Forecasted results (future)***	***Difference with previous***
January, 18th	$ 100,000	$ 131,064		Unknown
February, 18th	$ 100,000	$ 126,987		4,077
March, 18th	$ 100,000	$ 111,656		15,331
April, 18th	$ 100,000	$ 97,341		14,315
May, 18th	$ 100,000		$ 86,100	11,241 (average of previous 3 differences)
June, 18th	$ 100,000		$ 73,840	12,260 (average of previous 4 differences)

If ArchiSurance had decided to implement the program 'employee work efficiency improvement', this might have meant that data about employee efficiency would have to be measured and monitored over a period of time; new metrics need to be defined, etc. This may be done using a new iteration in the EAIL, thereby skipping the last activities of the current iteration in the EAIL. Such a result may look like Figure 72.

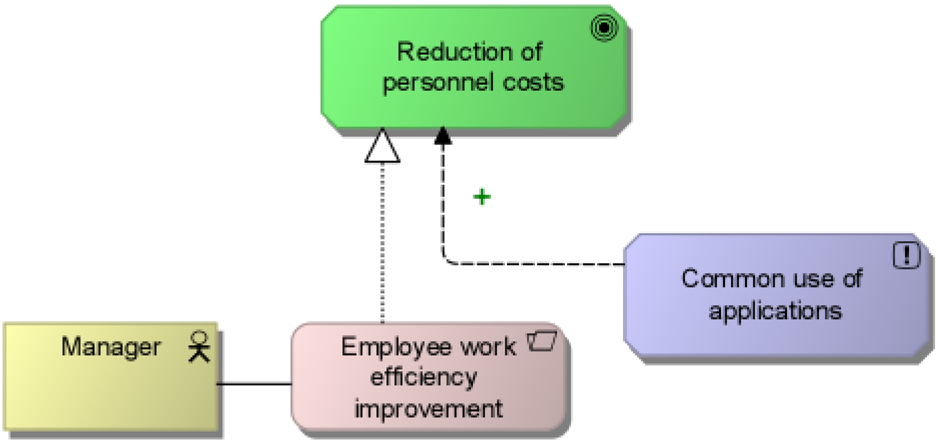


Figure 72. Possible outcome of program for ArchiSurance.

Activity VI-b Evaluate solution

The numbers shown in the table above may illustrate that the solution to the concern has helped over a period of four months, since the target was reached on April, 18th. The conclusion of the evaluation would be that this concern was minimized, creating a better situation for the organization. In this evaluation step, the consulting firm may already discuss a new concern to optimize the organization's situation even more. This means performing a new iteration of the EAIL.

Table 42. The Metric File for the ArchiSurance case.

Metric	
DESCRIPTION	
Name	*Costs (personnel)*
Information requirement	*To measure the personnel costs related to the concern 'profit'.*
Purpose	*To be able to determine if personnel costs may be minimized*
Measures	*The total costs related to employees (personnel)*
Inputs	*Number of employees, average salary costs per employee per month, salary costs per employee per year.*
ARCHITECTURAL INFORMATION	
Analysis approach	bottom-up analysis (Iacob & Jonkers, 2006).
Architecture layer	Business layer, application layer, technology layer.
DEFINITIONS	
Number of employees	The total number of employees in the organization
Average salary costs per employee per month	The average salary costs per month for an employee
Salary costs per employee per year	The specified yearly salary costs for each employee
INPUTS	
Input name	Number of employees
Unit of measure	Number (integer)
Frequency of collection	Monthly
Architectural representation	Application component & nodes (data stored in the systems)
Source of information	ArchiSurance general purpose service-cluster

Input name	Average salary costs per employee per month
Unit of measure	Money (double)
Frequency of collection	Monthly
Architectural representation	Application component & nodes (data stored in the systems)
Source of information	ArchiSurance general purpose service-cluster

Input name	Salary costs per employee per year
Unit of measure	Money (double)
Frequency of collection	Monthly
Architectural representation	Application component & nodes (data stored in the systems)
Source of information	ArchiSurance general purpose service-cluster

MEASUREMENT	
Indicator name	*Total yearly personnel costs*
Algorithm	*The number of employees variable is multiplied with the average salary costs per employee per month times twelve (months).*
Target / baseline	100.000 (dollar)
Decision criteria	Any number higher than the total yearly personnel costs will not fulfil the 'reduction of personnel costs' goal and will initiate alternative comparison view that might fulfil the higher goal. Any number lower than the total yearly personnel costs fulfils the 'reduction of personnel costs' goal.
Reference or support	(Iacob & Jonkers, 2006)

VISUALIZATION	
Visualization type	Relationship, realization (relations)
Visualization option	Enterprise Architecture Viewpoint
Sample	To show the outcome of the personnel costs and compare it to the related goal, indicating the processes (and other concepts) in the organization that are on the path (using realization relations, etc.). Showing related goals for decision support if the target goal is not fulfilled.

ANALYSIS PROCESS	
Analysis Frequency	Monthly
Interpretation	The chart shows the processes related to the personnel. Looking at the

5.3 Approach II – Data Source Enrichment

Description & Illustration – ArchiSurance Case

This approach, which focuses on enriching a data source instead of enriching an enterprise architecture, may be started at phase III. As we recall, this is where the data model is chosen. Thus far, a concept match library has been established that combines both operational data as well as enterprise architecture objects, more specifically, the properties of these EA objects. In the previous approach, we have shown that enterprise architecture is able to give an overview on the situation, which included the view on goal realization and the indication of related business concepts, shown in the enterprise architecture. The previous situation was not really exciting; the organization decided to do nothing and waited for a couple of months before possibly intervening.

On April 18th, ArchiSurance has reached their target, as the results proved to be positive and the goal was achieved: personnel costs are lower than 100,000 (cf. Table 41). ArchiSurance was intrigued by the forecasted results and decided to iterate again through the EAIL, using the same results from the previous iteration, however, this time choosing the Data Source Enrichment approach.

Phase III – Enrich

Activity III-a Determine data model

Previously, an enterprise architecture was selected as a data model. The Concept Match Library was used to enrich the enterprise architecture, in order to perform an analysis afterwards. This time, a source is determined to be the data model. Here, the meta data stored in the Concept Match Library will be used for enriching the data model, i.e. the DS subset part of the data model. Using additional data from the data source, more analysis possibilities may be possible.

Activity III-b Enrich data model

Due to the previous work in concept matching, the selected objects and their properties will be used as meta data for the records in the data sources that are part of the DS subset. Using the Concept Match Library (CML), queries are sent to the CML to retrieve the matched EA meta data. Such a query may have the following structure.

```
SELECT <insert_statement> AS <insert_statement> FROM <insert_CML_Name>
       GROUP BY <insert_grouping_statement>;
```

Depending on how the data source looks like internally, i.e. how it is structured, how *relations* (columns) are named, the query is filled in. Let us assume the query has been tailored to fit the CML relations and is able to load the data into the DS. With the data loaded into the DS, the next step is to determine and perform an analysis.

Activity III-c Perform analysis

As we have explained, ArchiSurance is interested in how future data may look like. This may help them in setting direction and deciding whether things might have to change or not. ArchiSurance wants to see the optimum situation for the next seven years regarding their total personnel costs, the average wage per month paid for personnel, while bearing in mind that the minimum wage should at least be 824 dollars. The data specialist used the data derived from the enterprise architecture to further enrich and analyse the data based on the wishes of the ArchiSurance board mentioned. The cost analysis data is shown in Table 43, with *number of employees* on the X-axis and *total personnel costs* on the Y-axis. The coloured data shows the average wages per employee per month for the next 7 years. The wages are calculated for people involved in 1 business process, 3 application services and 2 applications.

Table 43. Analysis data for the ArchiSurance case.

year		79	78	77	76	75	74	73	72
0	97341	1232,165	1247,962	1264,169	1280,803	1297,88	1315,419	1333,438	1351,958
1	86100	1103,846	1103,846	1118,182	1132,895	1148	1163,514	1179,452	1195,833
2	73840	934,6835	946,6667	958,961	971,5789	984,5333	997,8378	1011,507	1025,556
3	61580	779,4937	789,4872	799,7403	810,2632	821,0667	832,1622	843,5616	855,2778
4	57061	722,2911	731,5513	741,0519	750,8026	760,8133	771,0946	781,6575	792,5139
5	54830	694,0506	702,9487	712,0779	721,4474	731,0667	740,9459	751,0959	761,5278
6	54046	684,1266	692,8974	701,8961	711,1316	720,6133	730,3514	740,3562	750,6389
7	53793	680,9241	689,6538	698,6104	707,8026	717,24	726,9324	736,8904	747,125

Phase IV – Visualize

Activity IV-a Prepare data visualization

When the analysis has been performed, the result should be visualized in the model that was previously determined, in our case an operational data source. The data that was analysed was outputted in an XLS-file (Microsoft Excel) in order to create graphs for visualization.

Activity IV-b Visualize data

Using the built in functionality in Microsoft Excel, we have outputted the data (as shown in Appendix D and in Table 43) in a graph (cf. Figure 73). The graph shows the *average monthly wage of personnel* working in 1 business process, supporting 3 application services and 2 applications (cf. Appendix D), shown on the Y-axis. On the X-axis, the *total personnel costs* are shown and on the Z-axis the *number of employees* are shown.

Also, a graph (cf. Figure 74) was visualized that shows the estimated personnel costs data over the next 7 years. The data shows a trend of lowering personnel costs, from year one (year 0 should not be considered).

The data being visualized may hand new insights to data that could not be seen in an enterprise architecture. However, the data may be used in enterprise architecture for strategic purposes (like goal realization and planning).

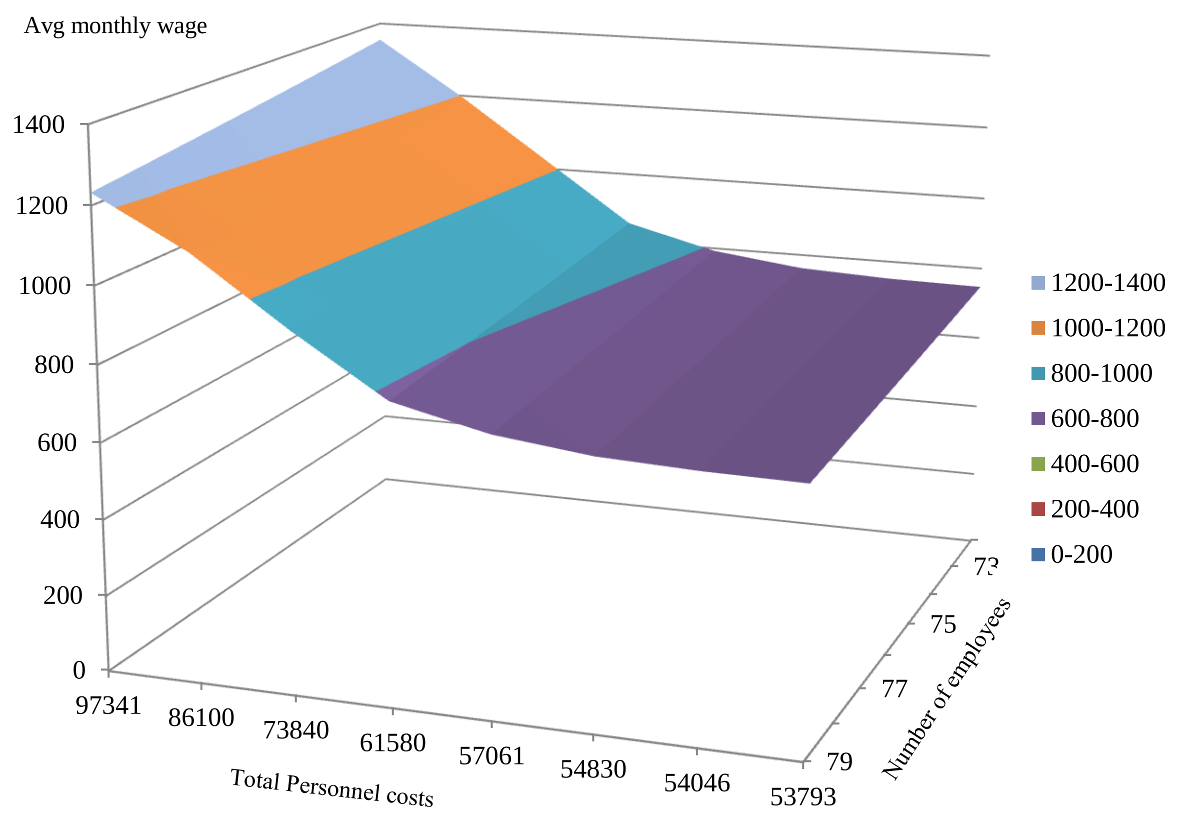


Figure 73. A 3D graph showing cost analysis data for the next 7 years.

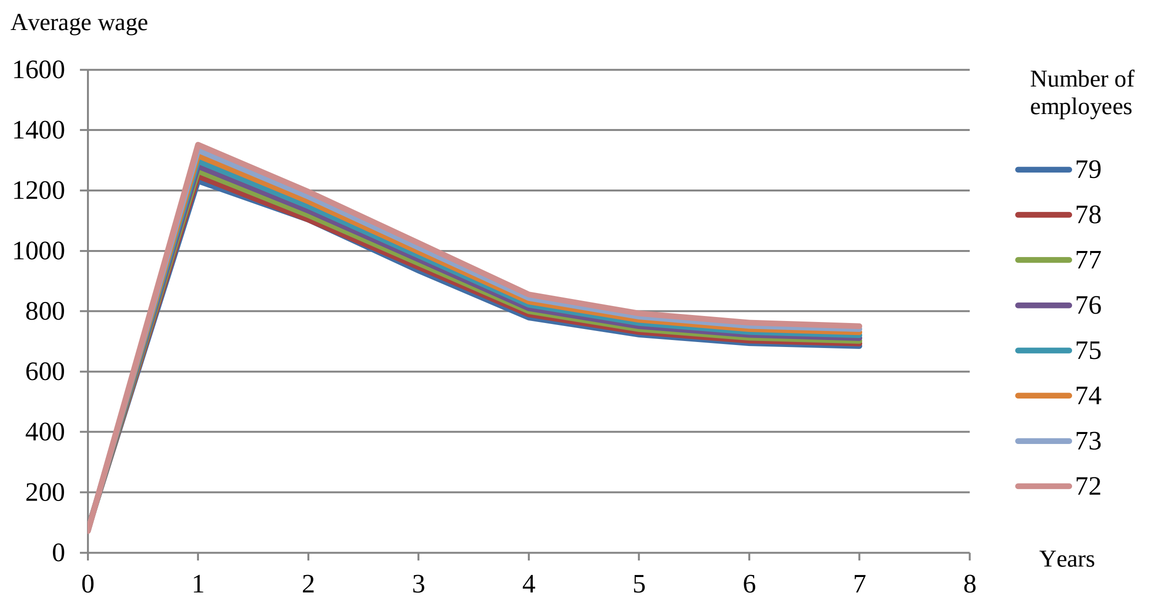


Figure 74. A graph showing the trend of personnel costs data over the next 7 years.

Phase V – Decide & Change

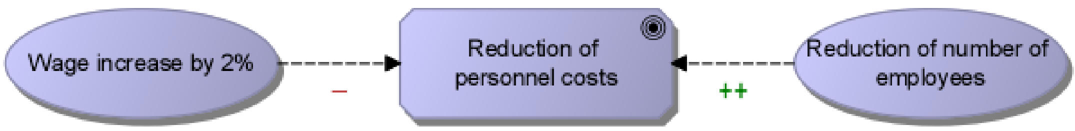
Activity V-a Make decision

Now the analysis data has been visualized, ArchiSurance has different types of information. In the first graph, Figure 73, we see the average wage and number of employees, together with the forecasted total personnel costs. From this graph we may conclude several things, namely that the personnel costs decline heavily in the first three years, but decline moderately in the years after. Also, bearing in mind the law enforcement that ‘a minimum wage of 825 dollars should be paid to anyone working in a company’, we need to keep track of when this average will be reached in the near future. Now looking at Figure 74, we see that the average of 825 dollars is reached (with the same total of personnel costs) within 4 years. ArchiSurance therefore decides to higher the wages slightly to increase employee satisfaction and avoid problems regarding legislation.

The current decision is a result of a previous iteration of the EAIL. Here, the level of focus was determined in phase II, match. With a data source as a data model, chosen in phase III, we see that ‘slicing and dicing’ (i.e. performing different views on the dataset) brings different information than an enterprise architecture would bring. The combination, however, lead to the possibility to forecast based on a dataset resulting from the combination of both enterprise architecture and operational data. Without having had a look at the enterprise architecture in the previous iteration of the EAIL, there would not have been a focus on *personnel costs* due to the ability of EA to link this variable to an organizational goal, which is shown in Figure 67. The meta data part provided by the enterprise architecture, that describes the context of the operational data, can be seen in Appendix D.

Activity V-b Develop solution

The decision means a financial change with respect to the wages of personnel. The finance department is instructed to higher the wages by 2% for the next 2 years. This may be done using enterprise architecture to model the relation the solution has with other enterprise architecture entities. The wage raise was modelled by a use case, together with a possible use case of a ‘reduction of number of employees’. Using these new EA objects, we pave the way for monitoring purposes in phase VI, explained later.



Activity V-c Implement solution

The finance department changed the numbers in the financial system and wages are paid with a 2% higher amount than last month’s pay checks. Such a change may have an influence on several parts of the organization. Not all of them may be interesting to monitor. If ArchiSurance would have implemented a decision in a small part of the organization to see and monitor the effects, ArchiSurance may receive information on how well the solution is received in the organization and loop back to activity ‘V-a make decision’ in order to decide differently.

Phase VI – Evaluate

Activity VI-a Monitor effects

Now changes were made to the situation, i.e. a decision was made, developed and implemented, ArchiSurance may want to monitor effects of the implemented solution. In this case, this might mean whether costs really do decline in the coming months or whether employee satisfaction has improved. This may mean that data is collected into the Concept Match Library (CML) and injected into the operational data source to do further analysis. Also, surveys may need to be taken to ask whether employees were happy with their increased wages (e.g. with a Likert-scale). ArchiSurance decided to do both and use the outcome for the evaluation. The data was inserted into the DS and gave new insights that were later modelled in the enterprise architecture, as shown in Figure 75. Here, we see that the ‘wage increase by 2%’ use case has a very positive effect on the ‘employee satisfaction increase’ goal, whereas the use case ‘reduction of number of employees’ has a very negative effect on employee satisfaction.

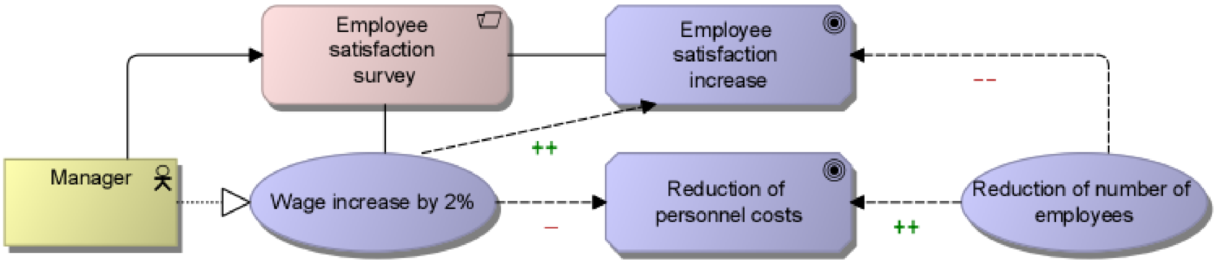


Figure 75. Result of the enriched DS used as input for the enterprise architecture.

Activity VI-b Evaluate solution

In this new EAIL cycle, the previous concern (the lowering of personnel costs) had already been met, ArchiSurance was given analysis data and visualization based on the data generated and stored by the CML. With these forecasts, ArchiSurance was able to overcome a possible problem regarding legislation, namely that average wages may not be lower than 825 dollars. Also, the fact that employees were happy with their increased wages really paid off; people are willing to work more and harder and this effect can already be seen by the fact that the personnel costs for the business process ‘handle claim’ are lower than before.

The use cases ‘wage increase by 2%’ and ‘reduction of number of employees’ have shown the possible effect on employee satisfaction increase. However, ArchiSurance decided not to fire its people, having a negative effect on the ‘reduction of personnel costs’. For ArchiSurance, it’s not worth the risk of firing people with the possibility that ArchiSurance may still need these people for other work to be done in the company.

Conclusions Part V

Using enterprise architecture and operational data in combination with the EAIL, a decision-maker is able to switch between types of information needed for several situations. We have shown different scenarios and multiple options as a result of interpretation of the different views on data, be it on an enriched enterprise architecture or an enriched data source. The EAIL support iterations and loops; it is a method that is dynamic and may be used for multiple consecutive concerns, each of which may require a different data model for visualization and interpretation by a decision-maker. Phases and activities may be skipped if the requirements for further activities are being met. We have shown the way a combination of enterprise architecture and operational data could address concerns that were listed by (Johnson et al., 2004), which is hard, if not impossible to address without the combination: doing so could be time-consuming, due to the necessity of retrieving the needed information that e.g. enterprise architecture or operational data elsewhere (like hiring consultants or asking advisors), and therefore expensive.

- Page intentionally left blank -

VI. DEMONSTRATION – TIMBER CASE STUDY

This case study is based on a real case problem in The Netherlands. For confidentiality reasons, the data used and the sketched situation are changed to a different setting.

6.1 Case Description

Timber is one of the largest firms regarding the production and processing of wood, located in Sweden. The organization has a few core business units, namely Forest, Wood, Logistics and Sales, next to the supportive business units Customer Relations, Human Resources, and IT. Timber deals with several types of costs, namely manufacturing costs (direct costs) and non-manufacturing costs (indirect costs), which can be further divided. Manufacturing costs are split into direct materials and direct labour costs. Non-manufacturing costs are split into marketing, sales, administrative and IT support costs.

Until now, the indirect costs have been always allocated to the organization as a whole. Recently, however, business units became profit and loss responsible. As an effect, they are now billed with costs by the supportive business units directly. For example, the *Sales* business unit and the *Wood* business unit are billed with server costs coming from the IT business unit. Due to the change, the business units want an IT cost clarification to identify why prices (e.g., of wood and labour) have increased. Timber, however, has no clear view on how IT costs are allocated to the business units. Therefore, they hire a consulting firm (specialized in EA) to clarify the situation.

6.2 Summary of Consult

By creating an overview of the current situation using enterprise architecture, more insight is realized that may be useful for decision support. With the overview, we are able to show the CIO and controller of the company how the IT department currently divides IT costs over the business units, indicate the problem of dividing these costs and visualize how this problem may be solved.

An enterprise architecture consultant performs the following actions. Calculating server costs for the Timber server park based on multiple variables adds up to the total costs of 'server utilization', which is the usage of server capacity for multiple business units. By calculating total costs for each operational business unit, based on server utilization, costs are allocated and are billed by the IT department accordingly.

To start off, a consultant (an Enterprise Architect) from a consulting firm will be confronted with the problem of translating the CIO's concern into decision support while combining the enterprise architecture and operational data sources available in the organization.

6.3 Phase I – Explore

For this case, an ***Enterprise Architect*** of a business consulting firm took the role of consultant to respond to the organization's motivation. Timber indicated that 'more requests are coming from the business units to clarify *why* some of their products have a high price', which is the motivation that initiated a business request for the consulting firm to assist. The organization Timber is our ***Stakeholder***. A root-cause analysis was performed to translate the motivation into a more accurate concern and, if possible, a root cause.

I-a Determine concern

Table 44. Description of the 'determine concern' activity.

Phase I-a	**Determine concern**
Actor(s)	CIO and Enterprise Architect (consultant).
Input	Business request of Timber.
Used techniques	Root cause analysis (Rooney & Vanden Heuvel, 2004), interview techniques (Kvale & Brinkmann, 2009).
Output	There is no cost-allocation.

As mentioned, the motivation that leaded to the case is that 'more requests are coming from the business units to clarify *why* some of their products have a high price'. But this is just the symptom of the real problem. The fact that more requests are coming from the business units does not necessarily mean that there is a problem. By asking *why* this is a problem, Timber mentioned that there is a concern about 'why some business unit products have a high price'. This is the concern Timber has. As mentioned in the case description, an event took place, namely that Timber decided that business units are now 'profit and loss responsible'. We are now on track of determining the problem, namely there is no clarity about the business unit product pricing, however, this is needed to conform to the profit and loss responsibility. The lack of clarity is the problem. Having identified the problem, there may be many causes leading to this problem. Determining the root-cause may take time. We need to know what leads to this unclarity and, therefore, need to know what would bring clarity or what would be the ideal situation to solve the problem. Having had talks with the CIO of Timber, we identified that product pricing is based on the sum of several cost allocations added with a profit margin. The profit margin is determined, however, data about cost allocation is missing. Shortly stated, there is no clear view regarding the relations certain operational systems have with the business. This is the lead to unclarity about why some business unit products have a high price; if we are able to show this, the organization may be helped in their goal to conform to profit and loss responsibility. The root-cause may already indicate the direction for a possible solution. The translation from *motivation* to *root-cause* is shown below.

Table 45. Root cause analysis result for the Timber case.

Motivation	More requests are coming from the business units to clarify *why* some of their products have a high price.
Concern	There is haziness about why some business unit products have a high price.

Problem	Clarity about business unit product pricing is lacking, but is needed to conform to profit and loss responsibility.
Root cause	There is no cost-allocation.

Having performed a root cause analysis, the concern was determined, as well as the problem and root-cause. Though a root cause analysis may not have been necessary to define the concern, it turned out useful when determining what is going on at Timber. The root cause hands a direction when looking for data in the organization, discussed later.

I-b Create source overview

Table 46. Description of the 'create source overview' activity.

Phase I-b	**Create source overview**
Actor(s)	Enterprise Architect and Data Specialist.
Input	Determined *concern*: 'there is no cost-allocation'.
Used techniques	Interview techniques (Kvale & Brinkmann, 2009), requirements analysis techniques (Hay, 2011).
Output	An overview of available data sources and a description of data stored in these data sources.

In this step relevant operational data source in the enterprise architecture are identified and a *source overview* is created. This source overview may be a global interpretation of the enterprise architecture and its data sources, but preferably it is a concise document that can be exchanged and referred to. An example is a mapping table between *sources* and *types of data* stored in them. The Enterprise Architect (a consultant) asks the CIO to bring him into contact with a Data Specialist to *create a source overview*. The Data Specialist creates the following overview of source systems and cost-related data (cf. Table 47). In the source overview only data sources are listed that *might* help solving the concern to avoid stating irrelevant information. The consultant gives advice to Timber on how to address the concern based on the data available *or* data that may be created using available systems, which is why the source overview is needed.

Table 47. Source overview for the Timber case.

Source	**Data type**
Financial system	Financial data (costs, revenues)
CRM system	Customer-related data (sales, etc.)
Servers 1 to 8	Utilization data (usage of servers for performing daily operations)

Within Timber several operational systems are able to output different types of data. This information is often outputted in large files (e.g. Microsoft Excel-files) with many records and columns. Looking back at our concern, namely the *lack of cost allocation*, we are mainly interested in data that is about *costs* and *usage*. Costs are interesting since they should be allocated over the several business units. Allocation could be done based on several indicators, like usage. Usage is a suitable indicator, since it can be measured and it gives a fair indication on who should pay what costs. The more a business unit uses a server, the more it should pay, the more costs are allocated to this business unit.

I-c Retrieve variables

Table 48. Description of the 'retrieve variables' activity

Phase I-c	**Retrieve variables**
Actor(s)	Stakeholder and Enterprise Architect.
Input	An overview of *available data sources* and *a description of the data* that is stored in these data sources.
Used techniques	Interview techniques (Kvale & Brinkmann, 2009), requirements analysis techniques (Hay, 2011).
Output	Retrieved *variables* (cf. Table 49).

Having created a source overview, the next step is to translate the concern, measureable goal *or* root cause into measurable variables that can be found in the data sources. These variables are the smallest elements which are later used to determine metrics. Examples of variables are units or rates like *processing time, costs,* or *workload rate*. The data types (shown in Table 49) help in allocating costs; the overview itself helps in determining variables. Determining the variables is based on the cost driver which is used to allocate costs (server costs) to cost objects (the business units) (Drury, 2007). Choosing the cost driver is based on three guidelines: the allocation must be fair, the allocation must be rational and verifiable, and the impact on the people who use or work with this information must be known (Horngren et al., 2006). Servers are able to perform different tasks using processors to calculate and storage to store data. These servers bring depreciation costs (of the processors and storage hard disks) to the company as well as maintenance and electricity costs. Based on the guidelines stated above and the information explained, the cost driver is determined to be ‘server utilization’, since this is considered to be fair, rational and verifiable. Moreover, the impact that allocation decisions may have on employees can be analysed. To define a metric, the data specialist provides cost related data per server (cf. Table 49), which is retrieved from the financial system and the servers 1 to 8 (cf. Table 47).

Table 49. Some variables to build a metric.

Variable name	**Data type**
Electricity (kWh)	Integer (in kilowatt hour (kWh))
Storage (GB)	Integer (in Gigabytes (GB))
Maintenance (hours)	Integer (in hours)
License costs (Euro)	Money (in Euro currency)
Software purchases (Euro)	Money (in Euro currency)
Server usage (percentage)	Real (percentage)

I-d Determine metric(s) & measurement(s)

Table 50. Description of the 'determine metric(s) & measurement(s)' activity

Phase I-d	**Determine metric(s) and measurement(s)**
Actor(s)	Enterprise Architect and Data Specialist.
Input	Retrieved variables (cf.Table 49).
Used techniques	Metrics design (Kerzner, 2011)(Hubbard, 2010), requirements analysis techniques (Hay, 2011) and algorithm design techniques (Skiena, 2008).
Output	An identified *metric 'total costs'* and its *measurement*.

Having found the variables (e.g., 'processing time' or 'workload rate'), we determine a suitable metric (e.g., *resource utilization*). This is an important step since it will be used for combining the EA with DSs. For this purpose, Priyanto (Priyanto, 2013) provided a metric template that can be used to structurally define the metric (resulting in a 'Metrics File'). Kerzner (2011) and Hubbard (2010) have provided thorough literature on how these variables and metrics can be determined. Determining a measurement means to determine *how* to measure the metric, e.g. by using a specific algorithm that uses the variables to calculate certain values. There may be multiple ways to calculate a metric, e.g. 'return on investment' (ROI). Defining an algorithm may require mathematical and algorithmic skills (Skiena, 2008).

To establish a cost allocation per business unit, the two-stage cost allocation process is used (Drury, 2007), which in our case comprises two steps, namely *(1) calculating product cost* based on server costs and *(2) reallocating product cost to multiple divisions*. The *total costs* metric based on the variables is calculated for each server, based on the amounts of electricity (kWh), storage (GB) and maintenance (FTE) multiplied by the prices for each variable. The prices are retrieved from the financial system. Without the 'determine variables' activity, it would be hard to determine a good metric capable of measuring data available in the organization.

6.4 Phase II – Match

II-a Determine EA & DS subsets

The Metrics File and Source overview File are used to determine the subsets of the enterprise architecture (EA) and data sources (DS) that are suitable as a source for further steps. It is critical that the chosen subsets from both EA and DS are relevant for the determined metric; the subsets are combined to better support decision-making in a later phase. An enterprise architect could determine which *EA-concepts* are relevant. If the problem is related to the whole organization, the scope consists of the entire EA. If the problem is related to a division of the organization, the scope is related concepts of the EA (e.g. business functions, processes, application components, and their relations). A data specialist with knowledge about the availability of data could determine which *DS-concepts* are relevant. To determine the subset, a scope is set regarding available data, which is based on the Source overview (cf.Table 47) At the end of this activity, both an EA-subset and a DS-subset are available for the next activity.

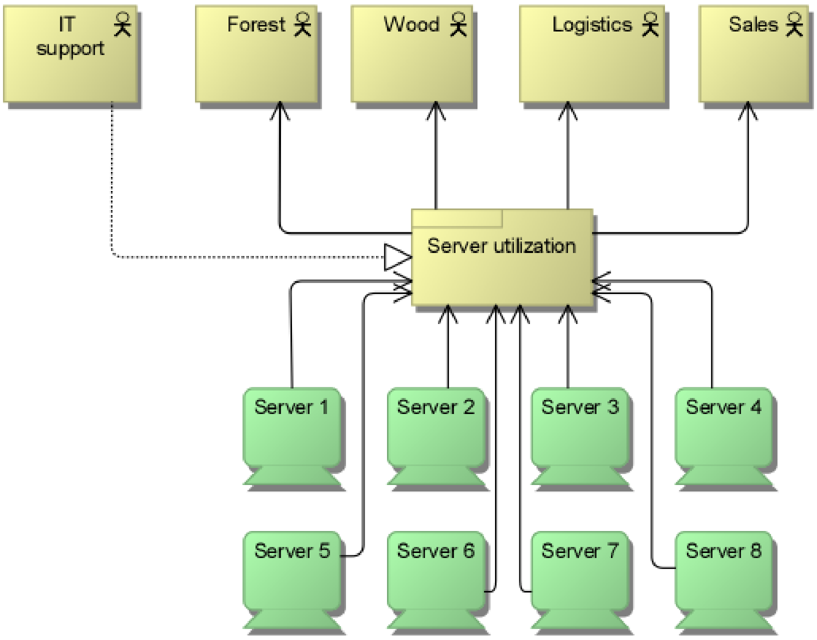


Figure 76. The EA-subset for the Timber case.

For Timber, an EA subset is modelled based on the relevant objects needed to address the concern. . The servers are modelled along with the business units. A business product 'server utilization' is added to the EA and a realization-relation links it with the IT support business actor, as illustrated in Figure 76. The DS subset consists of several Excel-files containing operational data, representing the determined variables. These files were delivered by an internal Timber data specialist.

Table 51. Description of the 'determine EA & DS subsets' activity.

Phase II-a	**Determine EA & DS subsets**
Actor(s)	Enterprise Architect and Data Specialist.
Input	A determined metric *'total costs'* and its measurement and a source overview, as support sources to identify the EA & DS subsets.
Used techniques	Requirements analysis techniques (Hay, 2011), data mining techniques, ETL, etc.(Kimball & Ross, 2013).
Output	Determined EA & DS subsets (cf.Figure 76). DS subset is not shown here – large size Excel-files.

II-b Match concepts

Table 52. Description of the 'match concepts' activity.

Phase II-b	**Match concepts**
Actor(s)	Enterprise Architect and Data Specialist.
Input	(1) A defined *metric* and its *measurement*. (2) Set of data (both enterprise architecture and operational data).
Used techniques	Requirements analysis techniques (Hay, 2011).
Output	A match of concepts derived from the enterprise architecture and the determined data sources (e.g. a mapping).

When the EA-subset and the DS-subset have been determined, both subsets are compared in order to find matches between concepts. In our terminology, a concept is either an object in an enterprise architecture (i.e., concept or relation) or a field in a data source (i.e., a value). An example of a *match* is a certain *business process attribute* within the EA-subset that is related to some *data value* found in the DS-subset.

> Matching concepts is complex, since it requires an understanding of the DS-subset and the EA-subset. However, it is possible to use the guidelines as stated in Table 21. These guidelines were used for matching the concepts; each time a match was found, the matches were saved. For example, the 'amount of GBs' property relates to a record value of '5627' for a business process with object_id '304', shown in Appendix C. Likewise, this is done for all properties in the EA and DS subsets, while bearing in mind the guidelines (cf. Table 21).

Concept matches can be saved for later use (e.g., for new iterations through the EAIL) in a 'Concept Match Library'. The library is a mapping between enterprise architecture, operational data and time, illustrated in Figure 49 and Appendix C. The model may serve as a blueprint for a database or even a spreadsheet. Using simple database functionality, *timestamps* that keep track of updates for each concept match are logged. The relations shown in the model are linked via primary keys and foreign keys in the *Concept_match* relation (cf. Figure 49). To illustrate where external data sources are placed, both the *EA* and *DS* relation were added to the model and linked to the *EA* and *DS subsets*. *EA*, *DS* and *Time* are interlinked using the *Concept_match relation*, which is directly linked to the *Property* and *Record relation*. An application of the model with a sample of data can be found in Appendix C.

For each object in the EA-subset (cf. Figure 76) attributes are added using a scripting language that is able to define attributes per EA object-type. Then, matches were made for each EA object's properties that were defined with the data given in the DS subset. The data was manually inserted in the EA subset using an enterprise architecture application called BiZZdesign Architect.

> Using a script, this 'enriching' could also be done automatically. The script has access to the CML, the enterprise architecture and the operational data source. Based on the matches made and shown in the CML, data could be refreshed in the CML by checking data changes in the enterprise architecture and operational data source on a given time interval, e.g. an hour, a day

or a week. The smaller the interval, the higher the level of detail the CML has. Due to the small set of data, we have inserted the data manually. Our case does not have to do with a time interval, thus we need no script to refresh the data; we are looking at one state of each of the matches made.

The snapshot of data as illustrated in Appendix C gives us an overview of some of the matches made for the Timber case. If the matches made were automatically stored in a database with attributes as stated in our model (cf. Figure 49), more possibilities would arise for data analysis. An example of a new analysis type could be 'data tracing', i.e. exactly tracing the EA or DS concepts used in a calculation, based on the 'accessed' attribute. Data tracing could serve issues like *'where does the data come from'* for the enterprise architecture field or *'what does this set of data mean in business terms'* for the business intelligence field. By defining an algorithm (Skiena, 2008) that measures the differences between timestamps in the 'accessed' attribute, data is traced for a specific analysis.

As shown in Figure 49, *property_id* relates to *record_id*. Match 11 (*match_id* with value 11) describes a property 'amount_kWh' for a certain object (*object_id*) and relates a value of '7600' using *record_id* '210'. The record belongs to a data source (*DS_id*) 1, which is a server. When a match is made, a timestamp (*matched*) is added. When the match is accessed (e.g. for a calculation), a timestamp (*accessed*) is updated. Appendix C shows a snapshot of all matches made for the Timber case.

6.5 Phase III – Enrich

III-a Determine data model

Table 53. Description of the 'determine data model' activity.

Phase III-a	**Determine data model**
Actor(s)	Enterprise Architect.
Input	Information to be able to determine the best choice for a model to be enriched with.
Used techniques	Enterprise Architecture application techniques.
Output	An enterprise architecture model.

Having matched the concepts from both EA and DS subsets, the next step is to determine the output data model. Depending on the output preferences of stakeholders (e.g. management) and whether the enterprise architecture or the data sources are able to submit to these preferences, a data model is chosen. Using the original enterprise architecture and data sources may be risky, e.g. when data is manipulated and errors are made. Possibly, a different data model is created, e.g. with different dimensions and optimized for business intelligence and other representation solutions. Creating a new enterprise architecture subset may simply be creating a copy of a selection of concepts (business processes, business functions, application components, etc.). Creating a new data source may be creating a direct copy of the most suitable data model to store data, e.g. a data mart. Determining the best data model depends on multiple factors to be taken into account by the consulting firm and the organization: both might influence the choice.

For Timber, the data model is an enterprise architecture. This is due to its ability to graphically represent business concepts understandable by the end users, the employees of Timber (i.e. the CIO, etc.). Also, an enterprise architecture is most suitable to show *how* costs are allocated and *where*. This is difficult using business intelligence applications that do not keep track of business concepts like enterprise architectures do. We will show this in some alternatives in phase IV, where we visualize the data.

III-b Enrich data model

Table 54. Description of the 'enrich data model' activity.

Phase III-b	**Enrich data model**
Actor(s)	Enterprise Architect.
Input	Data or meta-data (descriptive) derived from an EA-subset or a DS-subset.
Used techniques	Enterprise Architecture techniques, mathematical techniques, analysis techniques, etc.
Output	Enriched enterprise architecture.

The data model created in the previous activity is combined (or enriched) with data from the other source. If an enterprise architecture is the data model, it will be combined with data sources. If a data source is the data model, it will be combined with an enterprise architecture. For the enterprise architecture the previously matched DS-subset is used: the data is ‘loaded’ in the EA, creating an ‘enriched enterprise architecture’. For the data source the previously matched EA-subset is used: (meta-)data is loaded in the DS (e.g., descriptive data referring to EA properties and objects). Enriching a data model may be done using query scripts that retrieve data and, provided the matching of concepts is given (e.g., using the previously mentioned ‘Concept Match Library’), store data in the new data model. Simply stated, the script uses the locations of data as an input and output source for retrieving, respectively, storing data. The result of enriching an enterprise architecture could be an EA containing operational data as attributes of the architecture elements. The result of enriching a data source could be a data source (e.g. a data warehouse or data mart) with meta-data that describes the location of data in an enterprise architecture.

Timber enriches the enterprise architecture data model as illustrated in Figure 76. Since this example model is relatively small, data has been inserted manually for each attribute, for each server (1 to 8). The data is retrieved from the DS subset. With the enterprise architecture being enriched, calculations are made using the structure of the EA.

III-c Perform analysis

Table 55. Description of the 'perform analysis' activity.

Phase III-c	**Perform analysis**
Actor(s)	Enterprise Architect.

Input	An enriched model (e.g. an enriched enterprise architecture) or an enriched data source) and a 'Metric File' to determine the analysis to be performed.
Used techniques	Enterprise Architecture scripting techniques, mathematical techniques, analysis techniques, etc.
Output	The calculated metric, outputted in an 'Analysis Data'-file (or any other file suitable for visualizing).

The enriched data model is analysed upon. The analysis is performed by an analysis-function [9] in an application (e.g. an enterprise architecture application or a business intelligence application). The analysis is done using algorithms with parameters. The algorithm is based on the *metric* and *variables* described in the 'Metrics File', discussed in phase I. Several types of analysis may be performed, be it qualitative (using quantified measures) or quantitative. Measuring qualitative aspects may require phases like *preparation, coding, analysis* and *reporting* to make them measureable. Software applications like *HyperRESEARCH, QSR NVivo* and *ATLAS.ti* may assist in making qualitative data measureable. Information on how to measure qualitative data is explained by (Hubbard, 2010), amongst others. These applications may assist in phase I to determine variables and metrics as well. Quantitative analysis of data has been discussed in literature (Iacob & Jonkers, 2006) and is already widely performed in all kinds of business intelligence applications.

The first analysis to be performed for Timber to come to a total cost for all servers is calculated for each server, based on the amounts of electricity (kWh), storage (GB) and maintenance (FTE), multiplied by the prices for each variable, where prices are retrieved from a financial system. The second analysis to be performed for Timber is to allocate the total costs to each business unit, based on the server usage. Total costs are divided based on the server usage percentage and multiplied by the total server costs. The analysis algorithm is scripted inside BiZZdesign Architect using the EA subset as a data model. The script of 'data tracing' is shown below as an example of how scripting works within BiZZdesign Architect.

```
header "Data Tracing";
forall TypeID("ArchiMate:AbstractCompound") a1 in modelpackage {
        if (a1.hasProfileRef("ArchiMate:Time")) {

                forall TypeID("ArchiMate:AbstractCompound") b1 in modelpackage {
                        //comparing time attributes: if the difference is minimal
(in our case 1), then it is considered to be used in one and the same calculation.
                        if(((b1.attrValueRef("accessed") -
a1.attrValueRef("accessed")) <=1) &&(b1.attrValueRef("accessed")!=0)){
                                b1.setAttrValue("match", true);
                                output a1, a1.attrValueRef("match");
                        }
                }
        }
}
```

6.6 Phase IV – Visualize

IV-a Prepare data visualization

Table 56. Description of the 'prepare data visualization' activity.

Phase IV-a	**Prepare data visualization**
Actor(s)	Enterprise Architect.
Input	Analysis data (e.g. from an 'Analysis Data'-file).
Used techniques	Business intelligence techniques (Chaudhuri et al., 2011)(determining suitable input format for visualization application)(Kimball & Ross, 2013), SQL-techniques, etc.
Output	Prepared analysis-data.

When data has been analysed and made available for visualization, the dataset may be optimized for the software application that is able to represent the data in a way the decision-maker understands. Different types of decision-makers have different needs to be able to make thought-out decisions. These needs may determine the software application that needs to be selected. Preparing the data for visualization may only be necessary when a change is needed from one format to another. If a format is already optimized for the final representation, this activity may be skipped. The data should then be ready for visualization. For Timber, it is decided that the enterprise architecture application is suitable to show the analysis data. Therefore, the analysis data need not to be prepared to e.g. an Excel format, but was calculated ad hoc inside BiZZdesign Architect for each enterprise architecture object.

IV-b Visualize data

Table 57. Description of the 'visualize data' activity

Phase IV-b	**Visualize data**
Actor(s)	Enterprise Architect.
Input	Prepared analysis-data (e.g. from an 'Analysis Data'-file), in our case inside in an EA.
Used techniques	Business intelligence techniques (Chaudhuri et al., 2011), enterprise architecture techniques, scripting techniques.
Output	Visualized data.

In this step, the dataset is ready for representation. Depending on the data model and the format the data was tailored to, a visualization application should be selected to represent the data in a manner the decision-maker understands. As we briefly mentioned, different types of decision-makers exist (e.g., data-driven or structure-driven). The data that is shown should be understood (i.e. well-interpreted) by the decision-maker. For data visualization, literature about user interface design may be applied or best practices may be used as long as the organization is satisfied with the way data is presented. Priyanto (2013) explained different options that may be taken into account when developing for these business needs. The end result of this activity is data being visualized in a software application fulfilling the decision-maker's needs. For our problem, i.e. 'there is no cost allocation', we have visualized the allocation performed in two graphs, corresponding to the two-stage allocation process (Drury, 2007).

Figure 77 shows *how* the total costs of the product 'server utilization' is done, namely by calculating the total costs for each server, then adding them together, creating the total costs for the product.

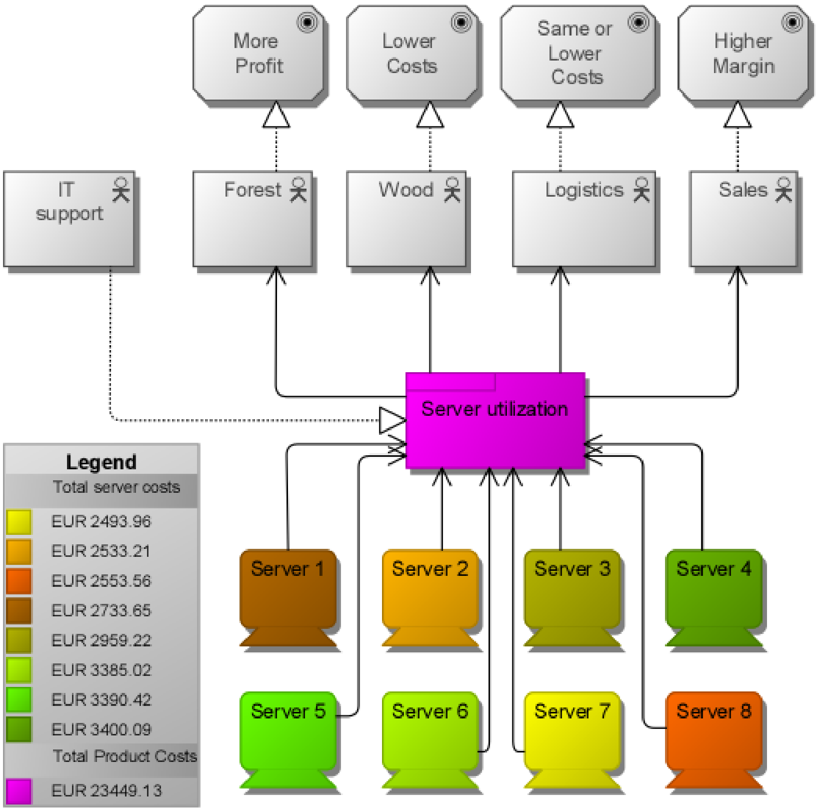


Figure 77. Calculating the product cost for 'server utilization'.

Figure 78 shows *how* the total costs of the product 'server utilization' are reallocated to the operational business units. This is done based on numbers from operational data sources and further analysis, as previously explained. If solely business intelligence was used to allocate the total costs of the product 'server utilization', we would have to have knowledge of *how* the servers relate to the product 'server utilization' and how allocation could be done fairly. Numbers would then be shown using different kind of charts, like pie charts and bar charts, but these do not reflect business concepts (in formal notation like ArchiMate) and are unable to relate data, these business concepts and relations all at once. Data specialists would need to get into contact with people from the business side in order to know how to 'tag' the operational data, i.e. how it should be represented to the business people. They would need information about which business units have which goals, etc. This is all represented in the enterprise architecture, which may also show future states of the organization (e.g. using plateaus in BiZZdesign Architect). Using business intelligence, multiple charts would be needed to represent the same information as we have done in a single enterprise architecture for this specific case. Figure 79 shows a fraction of business intelligence charts needed to represent a part of what is shown in Figure 77 and Figure 78. The right chart in Figure 79 would need an extra legend to represent the costs. Both charts do not hand possibilities to show *extra information* of the enterprise, like a motivation behind the presence of e.g. business processes. They are fixed regarding context. Enterprise architecture, however, is not. A combination of operational data and enterprise architecture provides both a dynamic context (i.e. expendable enterprise architecture) and evidence (i.e. proof of historic facts using operational data). Using our method, we have shown that there is a relation, by matching concepts and defined a measurement by finding variables to build a suitable metric. The information and data about *how* the servers relate to a defined product like 'server utilization' is information retrieved from conversations with a data specialist. Information about fair allocation is retrieved from literature. Cost related data comes from an operational system used by Timber. All of this data and information is brought together using the EAIL. Enterprise architecture and operational data are intertwined and were brought together in an enterprise architecture, which gave Timber the insight needed to explain why the business units are now billed with costs related to server utilization.

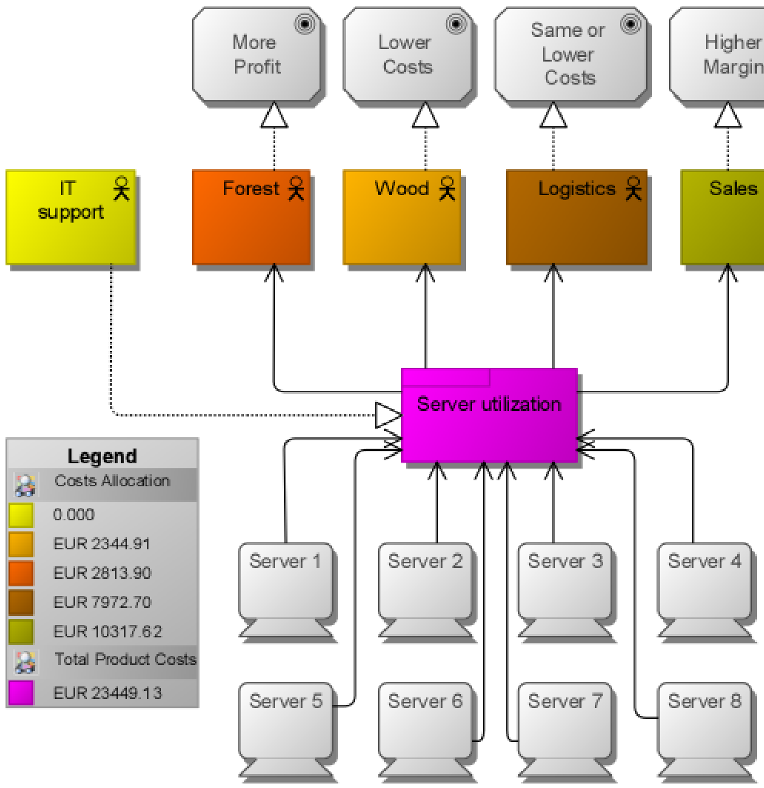


Figure 78. Reallocating the product costs to multiple division.

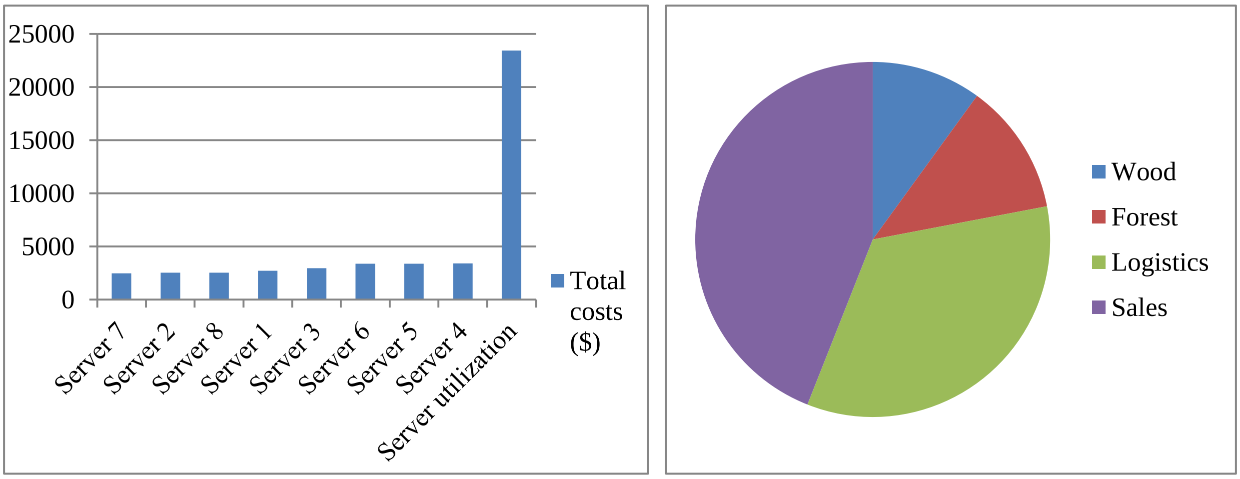


Figure 79. Some BI examples: total costs calculation (left) and costs reallocation to business units (right).

6.7 Phase V – Decide & Change

V-a Make decision

Table 58. Description of the 'make decision' activity.

Phase V-a	**Make decision**
Actor(s)	Timber's Board of Directors.
Input	Visualized data.
Used techniques	Decision-making techniques, scenario comparison techniques (Johnson et al., 2007), etc.
Output	Costs will be allocated according to business goals.

Until now we have discussed the steps that have led towards the visualization of data. In phase V, the organization makes a decision based on the represented data that leads to the implementation of a solution for the initial concern. Decision-making is done based on the data that was represented, interpreted by the decision-maker, which leads to information the decision-maker derives out of the data. From Figure 77 and Figure 78 Timber is able to see the relations between servers and a new 'product' that was identified in the concept match activity, more specifically, following step 4 in Table 21. This product relates to four of the five business units in a way that they are 'using' the product. The enterprise architecture is able to show that the 'IT' business unit also relates to the 'server utilization' product in a way that it does not use it, but realizes it. These differences are difficult to visualize using charts. The way data, relations, business objects and legend mix in enterprise architecture would need a lot of effort to realize using business intelligence, possibly needing multiple charts that are not going to cover the other facets (e.g. motivation, implementation of future states, etc.) that are already available in enterprise architecture. Moreover, such dashboards are considered to be confusing for decision-makers since they often present too much information (Bodt, 2014). Surely, sole business intelligence has its pros for data visualization, but our case required a form of 'traceability'. For cases that need the option to 'slice and dice' the data, it would be good to share meta data, as explained by Sun & Heller (2012), e.g. from enterprise architecture.

Timber is able to see the relations that were not visible before. The management had no clue about which business unit using what server, not even talking about the extent to which they are using the servers supported by the IT business unit. These relations became visible after the enterprise architect had talks and interviews with people from Timber of lower level management. Since the IT department and management deal with different areas of the organization, these talks are not present on a daily basis: the business does not know who to look for and the IT department does not know what to hand to the management. The enterprise architect provided the 'glue' between the two worlds. Furthermore, data that is now represented in the EA was available in the operational data systems somewhere, but it was not clear *where* and *what* data should be used. By just showing some data in a business intelligence application, without using a structured approach like the EAIL, the focus in traceability provided by enterprise architecture and the evidence given by operational data would not be used. The EAIL provided a way to systematically explore the organization, selecting the appropriate parts of data (to create a metric) and enterprise architecture representing the parts of the organization to be addressed, combined the two worlds and visualized them while maintaining the overview of structure and handing the evidence needed for decision-making. Questions like, 'where did we get this data from that we see here?' and 'how did you choose the allocation base?' are easy to answer now we used the structure of the EAIL. Wrong conclusions may be the result of wrong interpretations, e.g. if the relations between the 'server utilization' and the different business units were not shown. The IT department namely realizes the 'server utilization' product and the other business units (Forest, Wood, Logistics and Timber) use it. This level of detail is often generalized in simple charts.

Timber has several options to decide upon with both the structure and evidence that enterprise architecture, respectively operational data, provide. It may decide to allocate costs slightly different to be able to reach most targets. It may decide that the current allocation base, which was determined to be fair, should be extended with another allocation base in order to get a more detailed cost allocation, however, this may cost more time and, therefore, money. It may decide that the business unit IT should also take part in the cost allocation, since they also use systems for monitoring and possibly other reasons not directly traceable to the other business units (Wood, Forest, Logistics and Sales).

Due to the idea that the allocation is now fair and represents the situation as it should be, i.e. that the business units except IT should pay the costs, Timber decides that costs will be allocated according to the 'server utilization' product that was created based on operational data.

V-b Develop solution

Table 59. Description of the 'develop solution' activity.

Phase V-b	**Develop solution**
Actor(s)	Timber's middle management together with an Enterprise Architect.
Input	Costs will be allocated according to the 'server utilization' product.
Used techniques	e.g. tactical and operational skills for creating the most suitable solution for the current situation.
Output	The enterprise architecture is left as it was created by the Enterprise Architect and costs will be allocated accordingly.

Having decided upon a certain concern, the organization is able to take action in order to minimize the concern. The result of an action for a specific concern is a *solution* to that concern: it tries to solve or minimize the problem. Multiple solutions may exist to a specific concern, all attempting to assist in heading to a better situation for the organization. Determining the best solution may need different opinions and scenarios to take into account, as mentioned by (Johnson et al., 2007). A solution may be a set of actions to be taken, resulting in e.g. an action plan, but may also be a product or a service that needs to be created.

By asking the CIO and CFO to think of some measureable goals regarding the set goals, which are rather ambiguous currently, the consultant is able to dedicate properties to each goal with which goals can be measured. For Timber, the consultant sets the property 'total_costs' to a certain amount for each goal. Then, operational data in the enterprise architecture is calculated and compared to these goals' properties and once the optimal solution has been developed, i.e. most goals are reached, the solution is acceptable.

V-c Implement solution

Table 60. Description of the 'implement solution' activity.

Phase V-c	**Implement solution**
Actor(s)	Timber's Board of Directors & Enterprise Architect.
Input	The enterprise architecture is left as it was created by the Enterprise Architect and costs will be allocated accordingly.
Used techniques	Enterprise architecture techniques.
Output	Implemented solution.

When a solution has been determined and developed, the organization may implement (or execute) the solution in the organization. Before a solution is implemented, it could be tested in a *test-environment* to be able to predict its behaviour in the *real environment*. A test-environment may be a representation of the real environment or even an extreme environment to be able to see how the solution reacts to extreme situations. When the behaviour of the solution is acceptable, the solution may be transferred to the real environment.

For the Timber case, both an allocation of costs has been established, as shown previously in Figure 77 and Figure 78. Next to this, a goal realization has been additionally provided, showing how costs are allocated and relate to each business unit's goal.

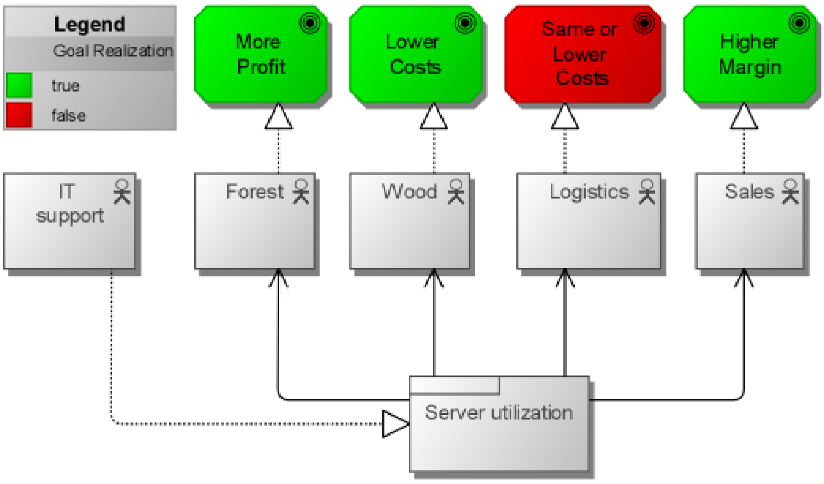


Figure 80. Goal realization for the Timber case.

6.8 Phase VI – Evaluate

VI-a Monitor effects

Table 61. Description of the 'monitor effects' activity.

Phase VI-a	**Monitor effects**
Actor(s)	Timber's Board of Directors & Enterprise Architect.
Input	Implemented solution.
Used techniques	Business intelligence techniques (Chaudhuri et al., 2011), enterprise architecture techniques, scripting techniques.
Output	Results of monitoring effects.

The solution being implemented in the organization does not necessarily mean the organization is relieved from the 'pain' the concern was causing, initially. It may be that the solution has helped in minimizing the concern, but did not manage to take eliminate the concern. To be able to determine whether a solution has helped in minimizing the initial concern means that the concern and solution should be monitored upon the effects the solution has brought to the organization. These effects may have come at a certain point and disappeared at a later point in time. This is the reason why effects need to be monitored for a period of time; the effects may not be permanent, making a final evaluation of the solution based solely on a 'snapshot' of a situation not fair. However, the effects should be taken into account when discussing the positive and negative effects of the solution. Examples of effects are *customer sales increase/decline* or *business leads increase/decline*. Timber keeps on monitoring the effects by monthly checking the utilization of servers and costs using its EA.

VI-b Evaluate solution

Table 62. Description of the 'evaluate solution' activity.

Phase VI-b	**Evaluate solution**
Actor(s)	Timber's Board of Directors & Enterprise Architect.
Input	Results of monitoring effects.
Used techniques	Decision-making skills, evaluation skills.
Output	Evaluation, possibly leading to a new EAIL iteration.

When the situation has come to an equilibrium, which is a stable moment when most temporary effects have appeared, the solution and situation are ready to be evaluated. Here, the organization and consulting firm discuss the way the solution has (hopefully) helped in minimizing the concern. The developed solution may completely solve the initial concern, but may also solve a part of it. When the organization decides that the concern should be minimized even more, the approach starts again in phase I, explore. The new situation becomes the 'current situation' and the EAIL starts all over, in a new quest for minimizing the (set of) concern(s). For now, Timber has explained it is happy with the provided insights and that it will keep in close contact regarding future possibilities.

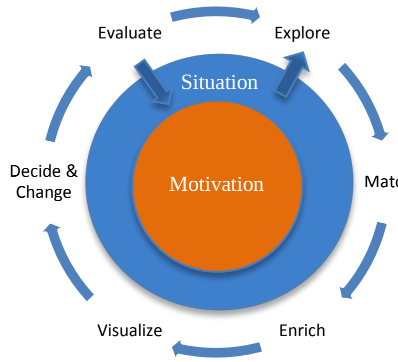
Conclusions Part VI

For the Timber case, an enterprise architecture was tailored and enriched with operational data. By following the activities provided by the EAIL, the organization was subject to an exploration phase, where operational data was identified, variables selected and a metric ‘total costs’ was created. The fact that the metric was not selected based on gut feeling, but selected based on the available data that was identified in the organization, indicates that a structure is needed instead of just finding some data and plotting them in graphs using business intelligence: using the EAIL gives a structured approach to providing information for decision-making. After matching concepts between operational data and enterprise architecture, matches were saved and used for enriching Timber’s enterprise architecture with operational data. Since the Timber case requires choosing a suitable data model that is able to visualize an allocation, an enterprise architecture was chosen that reflects comprehensible business concepts; i.e. entities the decision-makers are familiar with. By providing context using enterprise architecture, Timber is able to oversee relations between organizational entities and understand the allocation provided, instead of providing a dashboard with many charts that lack traceability (i.e. location and relations of entities), which is a ‘no go’ for decision-making due to the large amount of information presented (Bodt, 2014).

- Page intentionally left blank -

VII. EVALUATION

In this thesis we have tried to define a methodology to combine enterprise architecture and operational data called the *Enterprise Architecture Intelligence Lifecycle (EAIL)*. The methodology was developed based on experiences in the field of enterprise architecture consultancy while bearing in mind the available literature on this topic. The methodology is built up in six phases following up on each other in sequence. Organizations may lose the overview on their organization while the amount of IT grows and the organization itself becomes more complex. Our approach tries to help these organizations by providing both the overview using *enterprise architecture (EA)* as well as the quantifiable information often needed for decision-making using *operational data (OD)*. The methodology tries to assist in combining both fields using a structured approach.



We have shown in Part V that using sole business intelligence solutions do not always bring the needed information to decision-makers in a concise way; it often needs many charts and graphs to represent the same as an enriched enterprise architecture, resulting from the EAIL, could bring. We have demonstrated this as well in Part VI, where an allocation was done using an enriched enterprise architecture as a data model; enterprise architecture provides the context while operational data provides the evidence needed for decision-making. Business intelligence has difficulty showing relations between business concepts, not even mentioning the fact that it has no common notation for representing business concepts; something enterprise architecture does provide. On the other hand, however, enterprise architecture does not say anything factual about the organization if it does not include operational data; without cost data inserted in the Timber enterprise architecture, we would not have been able to represent useful information to the decision-makers. Using the EAIL, we have combined both worlds that, we believe, has resulted in better decision-support. Not combining enterprise architecture and operational data would have made it difficult to represent the context the case takes place, if not impossible. Without using the EAIL, steps would not be done as explained in our method, meaning parts like *using a metric based on available data* might not have been performed, meaning wrongly selected data and meta data would have been matched, amongst others. Surely, business intelligence applications support the description of data (e.g. in legends), but why doing double work if this information is already included in the enterprise architecture? Moreover, the enterprise architecture is dynamic, meaning it support extensions like motivation for choices that affect future states of the organization, which in itself can also be modelled using enterprise architecture. On the other hand, business intelligence may be very effective if the focus is solely on data and no context is needed.

Using the EAIL requires an organization to have or develop an enterprise architecture. Most organizations, however, will find difficulty in seeing benefits adapting enterprise architecture in their organization. The benefits of enterprise architecture often go back to the fact that it provides an overview; unless the organization is large enough, many organizations will not necessarily need an enterprise architecture to gain overview. However, the large organizations that *do* embrace enterprise architecture will try to gain the most of it and may be looking for new ways to use the investment, both in time and money, developing an enterprise architecture takes. Our approach tries to help these organizations.

Even though an organization may have an enterprise architecture, it does not necessarily mean its data sources are structured. This may be a problem when trying to find operational data for the EAIL. The effort spent on following the EAIL should be paid off by the benefits it tends to achieve, i.e. trying to solve a concern or to help an organization in providing new information using its enterprise architecture descriptive meta data (the business terminology expressed in concepts and relations) and its operational data (representing facts about the business). Thus far, the EAIL has not been extensively tested in the field as far as the effort needed for this to be spent on it concerned, however, its structure may already give a guideline for activities that need to be bore in mind by organizations when combining EA and OD. We believe when combining these worlds, it is necessary to perform at least the phases of the EAIL in general, while possibly skipping some of the activities that are not necessarily relevant for all cases. We acknowledge that some activities may be skipped, e.g. when iterating through the EAIL, while others may be extended by performing extra activities not mentioned in the EAIL.

As far as enterprise architecture concerned, we suggest that organizations not having an enterprise architecture gradually begin building its EA by addressing the complex parts in its organization in order to bring structure and overview. Using these insights, the organization may start migrating and implementing a new situation (e.g. using the Implementation and Migration extension (The Open Group, 2013a)). Inserting operational data in the enterprise architecture may hand facts and thereby reasons to move to a more beneficial situation that is e.g. less complex. We have shown the role of enterprise architecture for planning and control, namely by showing that goals may be assessed using operational data. Adding features like *alerting*, i.e. automatically notifying the organization when necessary, organizations may be given useful information at the right moment in time while directly showing *where* in the organization the problem is.

Looking at operational data, we admit the fact that data comes in all sorts and shapes and might be difficult to aggregate. We have shown ways to overcome this problem, e.g. by using data warehousing technologies that aggregate and cleanse the data. Next to this, *master data management* may be considered when looking for an integrative approach in data aggregation and its management (Loshin, 2008). Looking from a cost perspective, it would be ideal if financial systems would be combined with other operational data systems in a single data source, like a data warehouse. In such a system, most applications may operate a small subset of the data warehouse (i.e. a data mart) and data redundancy is limited, though we admit there are cons on using such a single source of data. In combination with the EAIL and the CML (Concept Match Library), data may be retrieved from and inserted in the data warehouse to better support decision-making using the concepts enterprise architecture brings.

Looking at the objectives we stated at the beginning of this thesis, we will concisely evaluate whether the objectives have been achieved and the degree to which our research has achieved them.

MO: To provide a decision support approach that combines enterprise architecture and data warehousing techniques.

We believe the approach described in this thesis is the main objective we were looking for. Surely, other approaches may also be able to achieve the main objective, however, the main guidelines are as we described in the EAIL.

SO1: Description of suitable input for enterprise architecture enrichment.

We have explained how enterprise architecture may be enriched. The input may be any type of data that can be found in operational data sources. In our approach, we have shown how an enterprise architecture may be enriched with data manually. Due to limitations of the tooling we used, we were not able to show how an enterprise architecture application may be combined with an external database. However, there are options to load data from CSV-files into an enterprise architecture. The input types here may be any type of data (e.g. integers, Strings, etc.).

SO 2: To define an enterprise architecture enrichment approach

The Enterprise Architecture Intelligence Lifecycle (EAIL) may be used either to enrich an operational data source (like a data warehouse) or to enrich an enterprise architecture. The latter approach shows how this objective is achieved.

SO3: Suitable cost analysis types for enterprise architecture enrichment

The cost analysis types have been thoroughly explained in Part III, Chapters 3.8, 3.9, 3.10 and 3.11. We have shown that cost allocation is suitable for enterprise architecture enrichment. Cost analysis types like cost-benefit analysis and break-even analysis may be performed likewise, by adding data to properties and calculating algorithms that belong to these type of cost analysis (cf. (Drury, 2007)). Their suitability lays, according to our opinion, in the overview provided regarding concepts familiar to the organization to which the enterprise architecture belongs.

SO4: Recommendation on cost analysis output for business intelligence

We have explained that business intelligence is often used by decision-makers (e.g. managers, board of directors). The data that is outputted as a result from cost analysis should be relevant, i.e. they should *(1) occur in the future* and *(2) differ among the alternative courses of action* (Horngren et al., 2012), as explained in Part III, Chapter 3.11.

- Page intentionally left blank -

VIII. CONCLUSIONS

In this thesis we provided a methodology to combine enterprise architecture and operational data. A structured approach was used as research methodology, namely the Design Science Research Methodology, as explained in Part II. We have given an overview of related literature of the main topics, namely *information, data, databases, data warehousing, enterprise architecture,* and *cost related subjects*. These topics can be found in Part III. In Part IV, we pose the idea on how an approach may be developed for decision-makers and what should be kept in mind while doing so. Part IV explains the approach we pose as the main contribution to literature, where we combine operational data and enterprise architecture to better support decision-making. Part VI demonstrates the application of the Enterprise Architecture Intelligence Lifecycle on a real-life case problem. Part VII evaluated on our approach and referred back to the main objectives stated at the beginning of this thesis and the way this thesis is able to achieve those objectives. In this part, Part VIII, we try to answer the research questions posed at the beginning of this thesis, we explain the limitations of our research, set some directions for future research and make recommendations on our work and related topics we came across while performing this research.

8.1 Research Questions

In the previous chapter we have already explained how the objectives are achieved. These objectives were the main source for the research questions we posed at the beginning of our thesis. We will provide a solution to the main thesis problem by answering the research questions, posed in Chapter I.

RQ1: What enterprise data can be used as input for EA-enrichment?

RQ1.1: What is the relation between enterprise data and costs?
RQ1.2: How to ensure data integrity for cost-related enterprise data?

Basically, all enterprise data may be used for enriching an enterprise architecture with. The main issue is how to get the data in the enterprise architecture. We have shown how this is done manually, however, for larger batches of data this may cause a problem. We believe that at this phase in enterprise architecture enrichment, it is sufficient to enrich enterprise architectures manually, since the enterprise architecture parts to be enriched in most cases will not be too large in order to do this. Enterprise data is data related to the organization. Costs occur in organizations as a result of daily operations and are, when logged, becoming cost data. If such cost data is subject to interpretation, it becomes cost information. Data integrity may be ensured using data warehousing techniques that 'cleanse' the data, like ETL (Extract, Transform, Load). Also, master data management (MDM) solutions may be implemented in an organization (Loshin, 2008), however thoughts on this topic differs in the research field, which is why MDM was not discussed in this thesis.

RQ2: How to enrich EA with cost-related enterprise data?

RQ2.1: What are the premises for EA to be enriched with enterprise data?
RQ2.2: What techniques exist to enrich EA?
RQ2.3: What is the influence of EA-enrichment on the enterprise?

Enterprise architecture may be enriched using a method like the EAIL that structurally passes all phases needed for EA-enrichment. Cost-related enterprise data may come from all kinds of different operational data sources, but ideally from a data warehouse-like application in order to save time and money in trying to perform ETL-like technologies to enrich the EA. Techniques like the Simple Query Language (SQL) and the Extended Markup Language (XML) could be used to enrich an enterprise architecture, however, in most cases it is sufficient to manually insert the data for each object in the EA if the EA subset is not that large. Enriching an EA may cost time and thus money. Moreover, it requires an embracement of the enterprise architecture paradigm, we believe. However, the benefits of enriching an enterprise architecture may be significant. For example, if historic data is able to show trends for future data, organizations may be able to predict and strategically use their enterprise architecture to overcome challenges or to make decisions that will benefit the organization regarding competition, amongst other possibilities. When using the EAIL, this does not mean other project methodologies should not be used: the EAIL is a specific methodology that does not exclude methodologies like PMBoK (Project Management Institute, 2013)(not explained in this thesis due to the thesis scope).

RQ3: Which analysis techniques are suitable for analysing EA enriched with cost-related data?

RQ3.1: Which methods for cost analysis exist?
RQ3.2: How can these methods for cost analysis be used in architectures enriched with cost-related data?

We have explained the analysis types possible regarding cost analysis, explained in Part III, Chapters 3.8 to 3.11. Many different types of cost analysis exist, but most are focused on *cost allocation, cost-effectiveness* and *cost-benefit* (Drury, 2007; Horngren et al., 2012; Sewell & Marczak, n.d.). Using scripts, the algorithms that belong to these different types of cost analysis may be implemented to run on top of the enterprise architecture, often outputted in a visualized manner. Analysis techniques that require a level of accuracy regarding the concern's context would fit for analysing an enterprise architecture.

RQ4: What cost analysis data derived from enriched EA can be used as input for business intelligence applications to support enterprise managers?

RQ4.1: How can enterprise managers be supported by EA?
RQ4.2: What cost-related information is needed regarding decision support for enterprise managers?

For decision-making, only cost analysis data that is *relevant* should be provided to enterprise management. These costs should *(1) occur in the future* and *(2) differ among the alternative courses of action* (Horngren et al., 2012), as explained in Part III, Chapter 3.11. Enterprise managers may be supported by EA, since EA provides *traceability* regarding the organization's entities managers are familiar with. Instead of providing all kinds of graphs and dashboards, like most business intelligence applications do, enterprise architecture is able to show the organization's concepts and give an overview of the situation the enterprise manager is interested to see.

Overall, we claim that our approach is able to better support decision-making due to its ability to translate problems into concepts known to decision-makers using familiar concepts tailored in their enterprise architecture, while using facts coming from operational data. Knowing that in an empirical study on 'effective information delivery and effective information use from a senior executive's perspective', it was confirmed that "*the organization's information systems' ability to deliver integrated information to people in the organization's hierarchy and processes positively influences* *the effective organizational use of information to support business activities and strategies*" (Kettinger, Zhang, & Chang, 2013), we believe our approach adds up to the organization's ability to *deliver integrated information* to these people 'in the organization's hierarchy'. In doing so, it 'positively influences the effective organizational use of information' in order to support the decision-making process.

8.2 Contributions

A. Contribution to practice

For practitioners, the Enterprise Architecture Intelligence Lifecycle (EAIL) hands structure for combining enterprise architecture and operational data. The method guides both organizations and consulting firms through all six phases of the EAIL to be able to decide upon a concern based on evidence provided by operational data, while using the context (meta data) of enterprise architecture.

For consulting firms and organizations, the Concept Match Library (CML) may be the solution for investing in better decision-support. Once implemented, the CML will provide the bridge between enterprise architecture and operational data, combining both worlds either to improve enterprise architecture or business intelligence, depending on the data model selected. Using all kinds of analysis, the CML collects operational data and meta data that may serve as a source for different kinds of visualization to be presented to decision-makers. Depending on the configuration of the CML, the CML may grow large enough to provide rich data for forecasting purposes and data tracing, amongst other analysis possibilities.

B. Contribution to science

For both the research fields of *enterprise architecture* and *business intelligence*, the Enterprise Architecture Intelligence Lifecycle (EAIL) aims to fill the gap that currently exists between both worlds. Handing a method that structurally leads an organization through all phases needed to combine operational data and enterprise architecture, the organization is given a new way of decision-making by giving a context to operational data and evidence to enterprise architecture.

For both the research fields of *enterprise architecture* and *business intelligence*, the Concept Match Library (CML) hands grip to the EAIL. New studies may be done with both EA and BI combined in the CML, looking whether the EAIL and CML indeed improve decision-making, e.g. by adding traceability and accuracy regarding data provision to decision-makers.

8.3 Limitations

The Enterprise Architecture Intelligence Lifecycle (EAIL) we pose in our thesis is based on a small set of related literature. Though we believe the approach is useful, more literature background would give a sound theoretical foundation for our approach. Furthermore, we were not able to thoroughly test our approach on a larger set of EA objects (a larger enterprise architecture) due to limited time. Moreover, our approach is not tested by other firms than the organization this thesis research was performed for. We would have loved to see our approach being used with a fully operational Concept Match Library, however, due to time limitations we have not implemented such a solution in an enterprise architecture application since these projects often take months to years.

8.4 Further Research

Future research may be performed in the following ways. First of all, it would be good to test the EAIL in other organizations and different sectors to see where possible improvements may be done regarding the EAIL. Also, it would be interesting to see new ways of enriching enterprise architecture, preferably in an automated way, i.e. using scripts and databases that e.g. use an implementation of the Concept Match Library. Moreover, we think the future for enterprise architecture is in its ability to support strategic decision-making. Forecasting algorithms and technology implemented in enterprise architecture applications that use the historic data provided using the EAIL approach are of great interest. We believe the next step is to integrate all kinds of different standard analysis types in enterprise architecture applications to show new reasons for embracing the enterprise architecture paradigm.

8.5 Recommendations

A. Recommendations for BiZZdesign

For BiZZdesign, we believe the future lays in enterprise architecture analysis. It would be good to provide different types of analysis packed in the BiZZdesign Architect Suite. Moreover, it would be good to provide new technologies in the field of graphically showing data. Thus far, the options are limited, i.e. there is a small set of data output types. If this set would be extended, we think it would benefit the position BiZZdesign has regarding its competition. It would be good to investigate the possibilities of implementing a Concept Match Library or a similar technology to log data. Such technology may form the basis for new technologies like forecasting, currently done in many different business intelligence applications. It may be wise to take a look at the status quo in business intelligence suites to benchmark what options may be integrated into BiZZdesign Architect. It would be good to create possibilities to integrate more database connectivity in order to automate processes and scripts that work together with the CML. This would benefit both BiZZdesign as well as its clients due to less time-consuming consults and more options regarding data visualization, like forecasting, amongst others.

B. Recommendations in general

One of the most time-consuming activities in performing the EAIL is being able to use an integrated set of data. As most organizations use many different applications, it may be difficult to be able to output data in a way that is not time-consuming. It would be good to use a ‘single source of truth’-like technology, like data warehousing, that often comes with a set of tools to perform queries on its large data set. Such tools may be able to output data in such a way that it is less time-consuming than having to look at all different data sources for relevant data to be combined with enterprise architecture in order to support decision-making.

References

American Accounting Association. Committee to Prepare a Statement of Basic Accounting Theory. (1966). *A statement of basic accounting theory*. American Accounting Association.

Asadullaev, S. (2009). Learning course: Data warehouse architectures and development strategy. IBM.

Barker, R. (1990). *CASE Method: Entity Relationship Modelling* (p. 240). Reading, MA: Addison-Wesley Professional.

Beynon-Davies, P. (2002). *Information systems: an introduction to informatics in organisations*. *European Journal of Information Systems* (Vol. 12, pp. 243–244).

Beynon-Davies, P. (2009a). Neolithic informatics: The nature of information. *International Journal of Information Management*, *29*(1), 3–14. doi:10.1016/j.ijinfomgt.2008.11.001

Beynon-Davies, P. (2009b). The "language" of informatics: The nature of information systems. *International Journal of Information Management*, *29*(2), 92–103. doi:10.1016/j.ijinfomgt.2008.11.002

BiZZdesign. (2013). Tool Manual Architect.

Bodt, M. (2014). *The Many Ways of Creating Dashboards Using SAS®* (p. 16). Wellington, New Zealand.

Chaudhuri, S., & Dayal, U. (1997). An Overview of Data Warehousing and OLAP Technology. In *ACM SIGMOD '97* (pp. 518–526).

Chaudhuri, S., Dayal, U., & Vivek, N. (2011). An overview of business intelligence technology. *Communications of the ACM*, *54*(8), 88–98.

Checkland, P., & Scholes, J. (1999). *Soft systems methodology: a 30-year retrospective* (Vol. 1999, p. 418). Wiley. Retrieved from http://books.google.com/books?id=r5coAQAAMAAJ&pgis=1

Codd, E. F., Codd, S. B., & Salley, C. T. (1993). *Providing OLAP (On-line Analytical Processing) to User-analysts: An IT Mandate* (p. 20). E. F.Codd & Associates. Retrieved from http://www.minet.uni-jena.de/dbis/lehre/ss2005/sem_dwh/lit/Cod93.pdf

Cross, N. (2001). Designerly Ways of Knowing: Design Discipline versus Design Science. *Design Issues*, *17*, 49–55.

Drury, C. (2007). *Management and Cost Accounting* (7th ed., p. 816). London: Cengage Learning; 7Rev Ed edition. Retrieved from http://www.amazon.co.uk/Management-Cost-Accounting-Colin-Drury/dp/1844805662

Elmasri, R., & Navathe, S. (2010). *Fundamentals of Database Systems* (6th ed., p. 1200). Addison-Wesley.

Forrester Research - Topic Overview: Business Intelligence. (2013). Retrieved December 10, 2013, from http://www.forrester.com/Topic+Overview+Business+Intelligence/-/E-RES39218?objectid=RES39218

Fuller, R. B., & McHale, J. (1965). *World Design Science Decade, 1965-1975: Five Two Year Phases of a World Retooling Design*. Retrieved from http://books.google.nl/books/about/World_design_science_decade_1965_1975.html?id=uPnTAAAAMAAJ&pgis=1

Gartner IT Glossary - Business Intelligence (BI). (2013). Retrieved December 10, 2013, from http://www.gartner.com/it-glossary/business-intelligence-bi/

Gero, J. (2004). The PhD Program in Design Science at the University of Sydney. *Chaoyang Journal of Design*, *5*, 15–19. Retrieved from http://163.17.26.248/~dcollege/conn/2004-design_5_3_fulltext.pdf

Greefhorst, D., & Proper, E. (2011). *Architecture Principles: The Cornerstones of Enterprise Architecture (The Enterprise Engineering Series)* (p. 220). Springer.

Gregor, S., & Jones, D. (2007). The Anatomy of a Design Theory. *Journal of the Association for Information Systems*, *8*(5), 312–335.

Gregory, S. (1966). *The Design Method* (p. 354). London: Butterworths. Retrieved from http://www.worldcat.org/title/design-method/oclc/1957643

Hay, D. C. (2011). *Requirements Analysis: From Business Views To Architecture* (p. 485). Prentice Hall Professional.

Hevner, A., & Chatterjee, S. (2010). *Design Research in Information Systems: Theory and Practice* (p. 320). Springer. Retrieved from http://www.amazon.com/Design-Research-Information-Systems-Integrated/dp/1441956522

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*, 75–105. doi:10.2307/249422

Horngren, C. T., Datar, S. M., & Rajan, M. V. (2012). *Cost Accounting: A Managerial Emphasis* (14th ed., p. 896). Upper Saddle River, New Jersey: Prentice Hall. Retrieved from http://www.amazon.com/Cost-Accounting-Managerial-Emphasis-14th/dp/0132109174

Horngren, C. T., Sundem, G. L., Stratton, W. O., Teall, H. D., & Gekas, G. (2006). *Management Accounting* (5th Canadi., p. 880). Pearson Education Canada.

Hubbard, D. W. (2010). *How to Measure Anything: Finding the Value of "Intangibles" in Business* (p. 323). New Jersey: John Wiley & Sons, Inc.

Iacob, M.-E., & Jonkers, H. (2006). Quantitative Analysis of Enterprise Architectures. In *Interoperability for Enterprise Software and Applications* (pp. 239–252). Springer, London.

IBM. (2010). *Capitalizing on Complexity: Insights from the Global Chief Executive Officer Study* (p. 71).

IEEE. (2007). ISO/IEC 42010:2007(E) - Systems and software engineering - Recommended practice for architectural description of software-intensive systems. The Institute of Electrical and Electronics Engineers, Inc.

Inmon, W. H. (2005). *Building the Data Warehouse*. (R. Elliott, K. Shafer, P. Hanley, K. Duggan, & M. B. Wakefield, Eds.) (Fourth edi., p. 543). Indianapolis, Indiana: John Wiley and Sons, Inc.

Inmon, W. H., Imhoff, C., & Sousa, R. (2001). *Corporate Information Factory* (p. 400). Wiley.

Inmon, W. H., Strauss, D., & Neushloss, G. (2008). *DW 2.0: The Architecture for the Next Generation of Data Warehousing* (1st ed., p. 400). Morgan Kaufmann.

Jensen, C. S., Pedersen, T. B., & Thomsen, C. (2010). *Multidimensional Databases and DataWarehousing* (p. 111). Morgan & Claypool Publishers.

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*. Stockholm, Sweden.

Johnson, P., Johansson, E., Sommestad, T., & Ullberg, J. (2007). A Tool for Enterprise Architecture Analysis. In *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007)* (pp. 142–142). IEEE. doi:10.1109/EDOC.2007.25

Jonkers, H., Band, I., & Quartel, D. (2012). ArchiSurance Case Study. The Open Group.

Jonkers, H., Lankhorst, M., Buuren, R. Van, Hoppenbrouwers, S., & Bonsangue, M. (2004). Concepts for Modelling Enterprise Architectures. *International Journal of Cooperative Information Systems*, *13*(3), 257–287. doi:10.1142/S0218843004000985

Kappelman, L., Mcginnis, T., Pettite, A., & Sidorova, A. (2008). Enterprise Architecture: Charting the Territory for Academic Research. In *Proceedings of the Fourteenth Americas Conference on Information Systems* (pp. 1–10). Toronto, Canada.

Kerzner, H. R. (2011). *Project Management Metrics, KPIs, and Dashboards: A Guide to Measuring and Monitoring Project Performance* (First., p. 380). John Wiley & Sons, Inc.

Kettinger, W. J., Zhang, C., & Chang, K. (2013). A View from the Top: Integrated Information Delivery and Effective Information Use from the Senior Executive's Perspective. *Information Systems Research*, *24*(3), 842–860.

Kimball, R., & Ross, M. (2002). *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling* (2nd ed., p. 421). John Wiley and Sons, Inc.

Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed., p. 600). John Wiley & Sons.

Kvale, S., & Brinkmann, S. (2009). *InterViews: Learning the craft of qualitative research interviewing* (2nd ed., p. 376). Sage Publications, Inc.

Lindström, Å., Johnson, P., Johansson, E., Ekstedt, M., & Simonsson, M. (2006). A survey on CIO concerns-do enterprise architecture frameworks support them? *Information Systems Frontiers*, *8*(2), 81–90. doi:10.1007/s10796-006-7972-0

Loshin, D. (2008). *Master Data Management* (1st edition., p. 304). Morgan Kaufmann. Retrieved from http://www.amazon.com/Master-Data-Management-The-Press/dp/0123742250

Morris, C. (1964). *Signification and Significance A Study of the Relations of Signs and Values*. The M.I.T. Press. Retrieved from http://www.amazon.com/Signification-Significance-Study-Relations-Values/dp/B0071OR5QA

Mutschler, B. (2008). *Modeling and Simulating Causal Dependencies on Process-Aware Information Systems from a Cost Perspective*. University of Twente.

Object Management Group (OMG). (2011). Business Process Model and Notation (BPMN). Retrieved May 03, 2014, from file:///C:/Users/r.veneberg/Downloads/formal-11-01-03.pdf

Paras, G. S. (2013). The Evolving Nature of the EA Profession. *Troux Technologies*, 10.

Peffers, K., Tuunanen, T., Rothenberger, M. a., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. doi:10.2753/MIS0742-1222240302

Pettey, C., & Van der Meulen, R. (2012). Gartner Says Big Data Disruptions Can Be Tamed With Enterprise Architecture. *Gartner*. Retrieved May 29, 2014, from http://www.gartner.com/newsroom/id/1986015

Priyanto, A. A. (2013). *Bringing enterprise architecture to the boardroom*. University of Twente. Retrieved from http://purl.utwente.nl/essays/64365

Project Management Institute. (2013). *A Guide to the Project Management Body of Knowledge* (p. 616). Newtown Square, PA, USA.

Rooney, J. J., & Vanden Heuvel, L. N. (2004). Root cause analysis for beginners. *Quality Progress*, *37*(7), 45–56.

Ross, J. W., Weill, P., & Robertson, D. (2006). *Enterprise Architecture As Strategy: Creating a Foundation for Business Execution* (p. 256). Boston, MA, USA: Harvard Business School Press. Retrieved from http://www.amazon.com/Enterprise-Architecture-Strategy-Foundation-Execution-ebook/dp/B004OC07EE

Schafrik, F. (2011). A practical guide to developing enterprise architecture. *IBM*. Retrieved March 06, 2014, from http://www.ibm.com/developerworks/rational/library/enterprise-architecture-maximum-value/

Schmandt-Besserat, D. (1978). The Earliest Precursor of Writing. *Scientific American*, *238*(6), 50–59. doi:10.1038/scientificamerican0678-50

Sewell, M., & Marczak, M. (n.d.). Using Cost Analysis in Evaluation. *University of Arizona*. Retrieved May 03, 2014, from http://ag.arizona.edu/sfcs/cyfernet/cyfar/Costben2.htm

Shannon, C. E. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, *27*(July 1928), 379–423.

Simon, D., Fischbach, K., & Schoder, D. (2013). Enterprise architecture management and its role in corporate strategic management. *Information Systems and E-Business Management*, 1–38. doi:10.1007/s10257-013-0213-4

Simon, H. A. (1996). *The Sciences of the Artificial*. *mitpressmitedu* (Vol. 4, p. 231). doi:10.2307/1511391

Skiena, S. S. (2008). *The Algorithm Design Manual* (2nd ed., p. 739). London: Springer-Verlag.

Stamper, R. K. (2001). Organisational Semiotics: Informatics without the Computer? In K. Liu, R. J. Clarke, P. B. Andersen, & R. K. Stamper (Eds.), *Information, Organisation and Technology* (pp. 115–171). Boston, MA: Springer US. doi:10.1007/978-1-4615-1655-2

Suh, N. P. (1990). *The Principles of Design* (p. 415). Oxford University Press. Retrieved from http://books.google.nl/books/about/The_Principles_of_Design.html?id=Z5fF5qQPQ9sC&pgis=1

Sun, H., & Heller, P. (2012). *Oracle Information Architecture: An Architect's Guide to Big Data* (p. 23). Redwood Shores, CA. Retrieved from http://www.oracle.com/technetwork/topics/entarch/articles/oea-big-data-guide-1522052.pdf

The Open Group. (2011). *TOGAF Version 9.1* (p. 692). The Open Group.

The Open Group. (2013a). *ArchiMate 2.1 Specification* (p. 181). The Open Group.

The Open Group. (2013b). ArchiMate®. Retrieved May 03, 2014, from http://www.opengroup.org/subjectareas/enterprise/archimate

Turban, E., Sharda, R., Delen, D., & King, D. (2010). *Business Intelligence* (2nd ed., p. 159). Prentice Hall. Retrieved from http://www.amazon.com/Business-Intelligence-Edition-Efraim-Turban/dp/013610066X

Van Aken, J. E., & Romme, G. (2009). Reinventing the future: adding design science to the repertoire of organization and management studies. *Organization Management Journal*, *6*, 2–12. doi:10.1057/omj.2009.1

Walls, Joseph G.Widmeyer, George R.El Sawy, O. A. (1992). Building an Information System Design Theory for Vigilant EIS. *Journal of Information Systems Research*, *3*(1), 36–59.

Watson, H. J., & Ariyachandra, T. (2005). *Data Warehouse Architectures: Factors in the Selection Decision and the Success of the Architectures* (pp. 1–55). Athens, Georgia and Cincinnati, Ohio.

Wieringa, R. (2009). Design science as nested problem solving. In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology - DESRIST '09*. Malvern, Pennsylvania, USA: ACM Press. doi:10.1145/1555619.1555630

Wieringa, R. (2010). Relevance and Problem Choice in Design Science. In *Proceedings of the 5th International Conference on Design Science Research in Information Systems and Technology* (pp. 61–76). St. Gallen, Switzerland: Springer-Verlag.

Wieringa, R., & Heerkens, H. (2008). Design Science, Engineering Science and Requirements Engineering. In *16th IEEE International Requirements Engineering Conference* (pp. 310–313). Barcelona, Spain: IEEE Computer Society Press. doi:10.1109/RE.2008.63

Willem, R. A. (1990). Design and science. *Design Studies*, *11*(1), 43–47. Retrieved from http://www.sciencedirect.com/science/article/pii/0142694X90900133

Zachman, J. A. (1987). A Framework for Information Systems Architecture. *IBM Systems Journal*, *26*(3), 276–292.

APPENDICES

A. Concern & Metric Templates

Concern Template

Concern	
DESCRIPTION	
Concern name	What is the concern?
Stakeholder	Who is the stakeholder?
Goals	What are the measurable goals to achieve based on the concern?
Measurement frequency	How often is the assessment of the goal achievement performed?
GOAL <number>	
Name	What is the goal to assess?
Analysis type	What type of EA-based analysis is performed?
Analysis metrics	What are the metrics to measure?
Information source	Where to find the source of information to measure the metrics? (e.g. from the enterprise architecture model or others)

Derived from Priyanto (2013).

Metric Template

Metric	
DESCRIPTION	
Name	What is the selected metric?
Information requirement	What is the motivation for selecting the metric?
Purpose	What is the purpose to measure the metric?
Measures	What will be measured? (output variables)
Inputs	What inputs are needed to calculate the measures? (input variables)
ARCHITECTURAL INFORMATION	
Analysis approach	Does the analysis use top-down or bottom-up approach?
Architecture layer	Which layer in the architecture is addressed by this metric? (business layer, application layer, or technology layer)
DEFINITIONS	
<Variable name 1>	Provide the definition of the input and output variables from the description section above
<Variable name 2>	...
<Variable name ...>	...
INPUTS	
Input name	Mention the first input variable
Unit of measure	What is the measurement unit? *(e.g. month, day, second, number, percentage, amount of money, count, et cetera)*
Frequency of collection	How often is the data collection performed?
Architectural representation	What is the representation of this variable in the architecture? *(e.g. business processes, application components, property of business functions, et cetera)*
Source of information	From whom is this input collected? *(e.g. system owner, system manager, or directly available in the architecture)*
Input name	Mention the second input variable, and so on..

Unit of measure	...
Frequency of collection	...
Architectural representation	...
Source of information	...
MEASUREMENT	
Indicator name	Mention the measured (output) variables
Algorithm	How the calculation will be performed to get this measures? *(e.g. the cost of an application component is contributed by the total cost of the resources from the lower layer of the architecture that are directly connected to this node)*
Target / baseline	What is the target or baseline?
Decision criteria	What is the meaning of the selected target or baseline for the output variable?
Reference or support	Mention the reference or source which is used to perform the calculation, if any *(e.g. journal, book, well known technique, et cetera)*
VISUALIZATION	
Visualization type	What is the relevant type of the visualization? *(comparison, relationship, distribution, or composition)*
Visualization option	What are the possible options for a visualization of analysis result by means of charts? Refer to chart suggestions guide *(e.g. line chart, bar chart, scatter-plot chart, etc.)*
Sample	Provide example(s) of the visualization (a real chart) and a brief information about the selected chart
ANALYSIS PROCESS	
Analysis Frequency	How often is the measurement analysis performed?
Interpretation	How should the chart visualization be interpreted and what kind of decision or insight can be acquired from the result?

Derived from Priyanto (2013).

B. Activity Description Tables

Activities Phase I – Explore

Phase I-a	***Determine concern***
Activity description	The determination of a concern.
Possible actor(s)	Stakeholder and Enterprise Architect.
Input	Business initiation request (motivation).
Usable techniques	Interview techniques (Kvale & Brinkmann, 2009), root cause analysis (Rooney & Vanden Heuvel, 2004), etc.
Output	Determined *concern*.

Phase I-b	***Create source overview***
Activity description	An exploration activity performed by the Enterprise Architect and Data Specialist to find suitable data in the organization in order to support the solving of the previously determined concern.
Possible Actor(s)	Enterprise Architect and Data Specialist.
Input	Determined *concern* (possibly translated in 'business request').
Usable techniques	Interview techniques (Kvale & Brinkmann, 2009), requirements analysis techniques (Hay, 2011).
Output	An overview of available data sources and a description of the data that is stored in these data sources (possibly outputted in a source overview).

Phase I-c	***Retrieve variables***
Activity description	The retrieving of 'variables' from the organization out of which a metric exists.
Possible Actor(s)	Stakeholder and Enterprise Architect.
Input	An overview of *available data sources* and *a description of the data* that is stored in these data sources (possibly outputted in a source overview).
Usable techniques	Interview techniques (Kvale & Brinkmann, 2009), requirements analysis techniques (Hay, 2011).
Output	Retrieved variables.

Phase I-d	***Determine metric(s) & measurement(s)***
Activity description	The determination of a metric, derived from EA/DS, and its measurement. The metric measurement means *how* we are going to measure, e.g. by using a specific algorithm that uses the previously determined variables to calculate certain values.
Possible Actor(s)	Enterprise Architect and Data Specialist.
Input	A *measureable goal*, possible changes derived from activity 4b.
Usable techniques	Metrics design (Kerzner, 2011)(Hubbard, 2010), requirements analysis techniques (Hay, 2011), algorithm design techniques (Skiena, 2008).
Output	An identified *metric* and its *measurement* (possibly logged in a 'Metrics File', cf. Appendix A)

Activities Phase II – Match

Phase II-a	***Determine EA & DS subsets***
Activity description	Determining the data subsets from EA and DS needed for matching.
Possible Actor(s)	Data Specialist & Enterprise Architect.
Input	A determined metric and its measurement and a source overview, as support sources to identify the EA & DS subsets.
Usable techniques	Requirements analysis techniques (Hay, 2011), data mining techniques, ETL (Kimball & Ross, 2013) etc.
Output	Determined EA & DS subsets.

Phase II-b	***Match concepts***
Activity description	Matching the concepts that have been determined in the enterprise architecture as well as the data sources. *Note: this is not the enrichment activity where data is combined.*
Possible Actor(s)	Enterprise Architect and Data Specialist.
Input	(1) A defined *metric* and its *measurement*. (2) Set of data (both enterprise architecture and operational data).
Usable techniques	Requirements analysis techniques (Hay, 2011).
Output	A match of concepts derived from the enterprise architecture and the determined data sources (e.g. a mapping).

Activities Phase III – Enrich

Phase III-a	***Determine data model***
Activity description	Determining (& creating) the model to be enriched (e.g. an EA or a DS).
Possible Actor(s)	Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Information to be able to determine the best choice for a model to be enriched with. For example, a match of concepts derived from the enterprise architecture and the determined data sources (mapping).
Usable techniques	Enterprise Architecture application techniques.
Output	A model (e.g. an enterprise architecture or a data source)

Phase III-b	***Enrich data model***
Activity description	Enriching the model with data. For example, enriching an EA-subset with data from a data subset (DS).
Possible Actor(s)	Consulting firm, (e.g. Enterprise Architect or Data Specialist).
Input	Data or meta-data (descriptive) derived from an EA-subset or a DS-subset
Usable techniques	Enterprise Architecture techniques, mathematical techniques, analysis techniques, etc.
Output	Enriched model

Phase III-c	***Perform analysis***
Activity description	Performing an analysis on the enriched model.
Actor(s)	Consulting firm (e.g. Enterprise Architect or Data Specialist)
Input	An enriched model (e.g. an enriched enterprise architecture (EA-E) or an enriched data source (DS-E)) and a 'Metric File' to determine the analysis to be performed.
Usable techniques	Enterprise Architecture techniques, mathematical techniques, analysis techniques, etc.
Output	The calculated metric, outputted in an 'Analysis Data'-file (or any other file suitable for visualizing)

Activities Phase IV – Visualize

Phase IV-a	***Prepare data visualization***
Activity description	Preparing the analysis-data, being the result from performing an analysis on the chosen data model with enriched data. For example, changing the model format from a Microsoft Excel (.XLS)-file to a Comma Separated Values (.CSV)-file.
Possible Actor(s)	Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Analysis data (e.g. from an 'Analysis Data'-file)
Usable techniques	Business intelligence techniques (Chaudhuri et al., 2011) (determining suitable input format for visualization application) (Kimball & Ross, 2013), SQL-techniques, etc.
Output	Prepared analysis-data

Phase IV-b	***Visualize data***
Activity description	Visualizing the prepared analysis-data, being the result from performing an analysis on the chosen data model with enriched data.
Possible Actor(s)	Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Prepared analysis-data (e.g. from an 'Analysis Data'-file)
Usable techniques	Business intelligence techniques (Chaudhuri et al., 2011), enterprise architecture techniques, scripting techniques.
Output	Visualized data

Activities Phase V – Decide & Change

Phase V-a	***Make decision***
Activity description	Deciding upon a concern, based on the visualized data.
Possible Actor(s)	Organization (e.g. management, board of directors), (possibly assisted by) Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Visualized data
Usable techniques	Decision-making techniques, scenario comparison techniques (Johnson et al., 2007), etc.
Output	Decision

Phase V-b	***Develop solution***
Activity description	Creating a solution based on the previously made decision (e.g. a set of activities to be performed by personnel on tactical or operational level).
Possible Actor(s)	Organization (e.g. tactical or operational personnel) or consulting firm
Input	Decision
Usable techniques	e.g. tactical and operational skills for creating the most suitable solution for the current situation
Output	Created solution

Phase V-c	***Implement solution***
Activity description	Implementing a solution previously developed (e.g. a set of activities to be performed by personnel on tactical or operational level).
Possible Actor(s)	Organization (e.g. management, board of directors), (possibly assisted by) Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Created solution
Usable techniques	e.g. tactical and operational skills for implementing the solution in the best way for the current situation
Output	Implemented solution

Activities Phase VI – Evaluate

Phase VI-a	***Monitor effects***
Activity description	Monitoring the possible effects occurring as a result from implementing the solution for a period of time.
Possible Actor(s)	Organization (e.g. management, board of directors), (possibly assisted by) Consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Implemented solution
Usable techniques	Business intelligence techniques (Chaudhuri et al., 2011), enterprise architecture techniques, scripting techniques.
Output	Results of monitoring effects

Phase VI-b	***Evaluate solution***
Activity description	Evaluating whether or not the concern was solved, partially or entirely, e.g. using results of monitoring effects.
Possible Actor(s)	Organization (e.g. management, board of directors) and consulting firm (e.g. Enterprise Architect, Data Specialist).
Input	Results of monitoring effects
Usable techniques	Decision-making skills, evaluation skills
Output	Evaluation

C. Concept Match Library – An Example View

Concept Match Library											
	Enterprise Architecture				Operational Data			Time			
match_id	***property_id***	*EA_id*	*object_id*	*description*	***record_id***	*value*	*DS_id*	***time_id***	*matched*	*accessed*	*plateau*
1	40	10	304 (Business process)	Amount of Gigabytes (GB)	200	5657	12 (CRM database)	1	02-03-2014 15:35:41	null	baseline
2	41	10	309 (Business actor)	Amount of Gigabytes (GB)	201	5353	13 (CRM database)	2	02-03-2014 16:12:28	null	baseline
3	42	10	310 (Business actor)	Utilization percentage	202	56	14 (CRM database)	3	03-03-2014 12:35:41	null	baseline
4	43	10	311 (Business actor)	Total costs (dollar)	203	353566.0	15 (CRM database)	4	05-04-2014 11:34:46	null	baseline
5	44	10	405 (Application component)	Amount of Gigabytes (GB)	204	3535	16 (CRM database)	5	15-05-2014 13:25:26	null	baseline
6	45	10	406 (Application component)	Utilization percentage	205	12	17 (CRM database)	6	15-05-2014 13:27:49	null	baseline
7	46	10	506 (Used-by relation)	Weight	206	0.5	18 (CRM database)	7	09-09-2014 12:09:06	null	baseline
8	47	10	507 (Composition relation)	Weight	207	1	19 (CRM database)	8	12-09-2014 11:12:56	null	baseline
9	48	10	508 (Used-by relation)	Weight	208	1	20 (CRM database)	9	null	null	target
10	49	10	509 (Composition relation)	Weight	209	0.7	21 (CRM database)	10	null	null	target
11	50	11	402 (Device)	Total costs (dollar)	210	33777	22 (Node)	11	12-09-2014 11:13:56	12-09-2014 11:17:56	baseline
12	51	11	403 (Device)	Total costs (dollar)	211	56432	23 (Node)	12	12-09-2014 11:13:58	12-09-2014 11:17:56	baseline
13	52	11	404 (Device)	Total costs (dollar)	212	46784	24 (Node)	13	12-09-2014 11:14:12	12-09-2014 11:17:56	baseline
14	53	11	405 (Device)	Total costs (dollar)	213	12456	25 (Node)	14	12-09-2014 11:14:15	12-09-2014 11:17:56	baseline
15	54	11	601 (Business Product)	Total costs (dollar)	214	90209	null	15	12-09-2014 11:14:19	12-09-2014 11:17:56	baseline
16	55	11	602 (Business Product)	Total costs (dollar)	215	46784	null	16	12-09-2014 11:14:27	12-09-2014 11:17:56	baseline
17	56	11	603 (Business Product)	Total costs (dollar)	216	12456	null	17	12-09-2014 11:14:35	12-09-2014 11:17:56	baseline
18	57	11	702 (Business Unit)	Total costs (dollar)	217	149449	null	18	12-09-2014 11:14:42	12-09-2014 11:17:57	baseline

D. Operational Data Source (Enriched) – ArchiSurance Case (Example)

Operational Data Source											
Enterprise Architecture				Operational Data						Time	
property_id	*EA_id*	*object_id*	*description*	***record_id***	*value*	*personnel_costs*	*nr_employees*	*av_salary_employee_month*	*salary_employee_year*	***time_id***	*plateau*
40	10	304 (Business process)	Personnel costs	200	49286,6	49286,6	40	1232,165	14785,98	1	baseline
41	10	309 (Application service)	Personnel costs	201	14785,98	14785,98	12	1232,165	14785,98	2	baseline
42	10	310 (Application service)	Personnel costs	202	12321,65	12321,65	10	1232,165	14785,98	3	baseline
43	10	311 (Application service)	Personnel costs	203	16018,145	16018,145	13	1232,165	14785,98	4	baseline
44	10	405 (Application component)	Personnel costs	204	2464,33	2464,33	2	1232,165	14785,98	5	baseline
45	10	406 (Application component)	Personnel costs	205	2464,33	2464,33	2	1232,165	14785,98	6	baseline
46	10	304 (Business process)	Personnel costs	206	47512926,25	43594,92	40	1089,873	13078,476	7	1 month
47	10	309 (Application service)	Personnel costs	207	14253877,87	13078,476	12	1089,873	13078,476	8	1 month
48	10	310 (Application service)	Personnel costs	208	11878231,56	10898,73	10	1089,873	13078,476	9	1 month
49	10	311 (Application service)	Personnel costs	209	15441701,03	14168,349	13	1089,873	13078,476	10	1 month
50	10	405 (Application component)	Personnel costs	210	2375646,312	2179,746	2	1089,873	13078,476	11	1 month
51	10	406 (Application component)	Personnel costs	211	2375646,312	2179,746	2	1089,873	13078,476	12	1 month
52	10	304 (Business process)	Personnel costs	212	34945329,81	37387,34	40	934,6835	11216,202	13	2 months
53	10	309 (Application service)	Personnel costs	213	10483598,94	11216,202	12	934,6835	11216,202	14	2 months
54	10	310 (Application service)	Personnel costs	214	8736332,452	9346,835	10	934,6835	11216,202	15	2 months
55	10	311 (Application service)	Personnel costs	215	11357232,19	12150,8855	13	934,6835	11216,202	16	2 months
56	10	405 (Application component)	Personnel costs	216	1747266,49	1869,367	2	934,6835	11216,202	17	2 months
57	10	406 (Application component)	Personnel costs	217	1747266,49	1869,367	2	934,6835	11216,202	18	2 months
58	10	304 (Business process)	Personnel costs	218	24304417,13	31179,748	40	779,4937	9353,9244	19	3 months
59	10	309 (Application service)	Personnel costs	219	7291325,14	9353,9244	12	779,4937	9353,9244	20	3 months
60	10	310 (Application service)	Personnel costs	220	6076104,283	7794,937	10	779,4937	9353,9244	21	3 months
61	10	311 (Application service)	Personnel costs	221	7898935,568	10133,4181	13	779,4937	9353,9244	22	3 months

62	10	405 (Application component)	Personnel costs	222	1215220,857	1558,9874	2	779,4937	9353,9244	23	3 months
63	10	406 (Application component)	Personnel costs	223	1215220,857	1558,9874	2	779,4937	9353,9244	24	3 months
64	10	304 (Business process)	Personnel costs	224	20868177,33	28891,644	40	722,2911	8667,4932	25	4 months
65	10	309 (Application service)	Personnel costs	225	6260453,198	8667,4932	12	722,2911	8667,4932	26	4 months
66	10	310 (Application service)	Personnel costs	226	5217044,331	7222,911	10	722,2911	8667,4932	27	4 months
67	10	311 (Application service)	Personnel costs	227	6782157,631	9389,7843	13	722,2911	8667,4932	28	4 months
68	10	405 (Application component)	Personnel costs	228	1043408,866	1444,5822	2	722,2911	8667,4932	29	4 months
69	10	406 (Application component)	Personnel costs	229	1043408,866	1444,5822	2	722,2911	8667,4932	30	4 months
70	10	304 (Business process)	Personnel costs	230	19268249,41	27762,024	40	694,0506	8328,6072	31	5 months
71	10	309 (Application service)	Personnel costs	231	5780474,824	8328,6072	12	694,0506	8328,6072	32	5 months
72	10	310 (Application service)	Personnel costs	232	4817062,354	6940,506	10	694,0506	8328,6072	33	5 months
73	10	311 (Application service)	Personnel costs	233	6262181,06	9022,6578	13	694,0506	8328,6072	34	5 months
74	10	405 (Application component)	Personnel costs	234	963412,4707	1388,1012	2	694,0506	8328,6072	35	5 months
75	10	406 (Application component)	Personnel costs	235	963412,4707	1388,1012	2	694,0506	8328,6072	36	5 months
76	10	304 (Business process)	Personnel costs	236	18721168,19	27365,064	40	684,1266	8209,5192	37	6 months
77	10	309 (Application service)	Personnel costs	237	5616350,458	8209,5192	12	684,1266	8209,5192	38	6 months
78	10	310 (Application service)	Personnel costs	238	4680292,048	6841,266	10	684,1266	8209,5192	39	6 months
79	10	311 (Application service)	Personnel costs	239	6084379,663	8893,6458	13	684,1266	8209,5192	40	6 months
80	10	405 (Application component)	Personnel costs	240	936058,4097	1368,2532	2	684,1266	8209,5192	41	6 months
81	10	406 (Application component)	Personnel costs	241	936058,4097	1368,2532	2	684,1266	8209,5192	42	6 months
82	10	304 (Business process)	Personnel costs	242	18546305,2	27236,964	40	680,9241	8171,0892	43	7 months
83	10	309 (Application service)	Personnel costs	243	5563891,56	8171,0892	12	680,9241	8171,0892	44	7 months
84	10	310 (Application service)	Personnel costs	244	4636576,3	6809,241	10	680,9241	8171,0892	45	7 months
85	10	311 (Application service)	Personnel costs	245	6027549,189	8852,0133	13	680,9241	8171,0892	46	7 months
86	10	405 (Application component)	Personnel costs	246	927315,2599	1361,8482	2	680,9241	8171,0892	47	7 months
87	10	406 (Application component)	Personnel costs	247	927315,2599	1361,8482	2	680,9241	8171,0892	48	7 months

E. Concept Match Library – SQL 'Create'-Script

```
-- Created by Vertabelo (http://vertabelo.com)
-- Designed by Roel Veneberg
-- Script type: create
-- Scope: [tables, references, sequences]
-- Generated at Tue May 27 10:20:44 UTC 2014

-- tables
-- Table Concept_Match
CREATE TABLE Concept_Match (
    match_id int  NOT NULL,
    time_id int  NOT NULL,
    property_id int  NOT NULL,
    Time_time_id int  NOT NULL,
    Record_record_id int  NOT NULL,
    CONSTRAINT Concept_Match_pk PRIMARY KEY (match_id)
);

-- Table DS
CREATE TABLE DS (
    DS_id int  NOT NULL,
    CONSTRAINT DS_pk PRIMARY KEY (DS_id)
);

-- Table DS_subset
CREATE TABLE DS_subset (
    DS_subset_id int  NOT NULL,
    DS_id int  NOT NULL,
    CONSTRAINT DS_subset_pk PRIMARY KEY (DS_subset_id)
);

-- Table EA
CREATE TABLE EA (
    EA_id int  NOT NULL,
    CONSTRAINT EA_pk PRIMARY KEY (EA_id)
);

-- Table EA_subset
CREATE TABLE EA_subset (
    EA_subset_id int  NOT NULL,
    EA_id int  NOT NULL,
    CONSTRAINT EA_subset_pk PRIMARY KEY (EA_subset_id)
);

-- Table Object
CREATE TABLE Object (
    object_id int  NOT NULL,
    EA_subset_id int  NOT NULL,
    CONSTRAINT Object_pk PRIMARY KEY (object_id)
);

-- Table Property
CREATE TABLE Property (
    property_id int  NOT NULL,
    object_id int  NOT NULL,
    description varchar(255)  NOT NULL,
    CONSTRAINT Property_pk PRIMARY KEY (property_id)
);

-- Table Record
```

```sql
CREATE TABLE Record (
    record_id int  NOT NULL,
    DS_subset_id int  NOT NULL,
    value varchar(255)  NOT NULL,
    CONSTRAINT Record_pk PRIMARY KEY (record_id)
);

-- Table Time
CREATE TABLE Time (
    time_id int  NOT NULL,
    CONSTRAINT Time_pk PRIMARY KEY (time_id)
);

-- foreign keys
-- Reference:  Concept_Match_Property (table: Concept_Match)

ALTER TABLE Concept_Match ADD CONSTRAINT Concept_Match_Property
FOREIGN KEY Concept_Match_Property (property_id)
    REFERENCES Property (property_id);
-- Reference:  Concept_Match_Record (table: Concept_Match)

ALTER TABLE Concept_Match ADD CONSTRAINT Concept_Match_Record FOREIGN
KEY Concept_Match_Record (Record_record_id)
    REFERENCES Record (record_id);
-- Reference:  Concept_Match_Time (table: Concept_Match)

ALTER TABLE Concept_Match ADD CONSTRAINT Concept_Match_Time FOREIGN
KEY Concept_Match_Time (Time_time_id)
    REFERENCES Time (time_id);
-- Reference:  DS_subset_DS (table: DS_subset)

ALTER TABLE DS_subset ADD CONSTRAINT DS_subset_DS FOREIGN KEY
DS_subset_DS (DS_id)
    REFERENCES DS (DS_id);
-- Reference:  EA_subset_EA (table: EA_subset)

ALTER TABLE EA_subset ADD CONSTRAINT EA_subset_EA FOREIGN KEY
EA_subset_EA (EA_id)
    REFERENCES EA (EA_id);
-- Reference:  Object_EA_subset (table: Object)

ALTER TABLE Object ADD CONSTRAINT Object_EA_subset FOREIGN KEY
Object_EA_subset (EA_subset_id)
    REFERENCES EA_subset (EA_subset_id);
-- Reference:  Property_Object (table: Property)

ALTER TABLE Property ADD CONSTRAINT Property_Object FOREIGN KEY
Property_Object (object_id)
    REFERENCES Object (object_id);
-- Reference:  Record_DS_subset (table: Record)

ALTER TABLE Record ADD CONSTRAINT Record_DS_subset FOREIGN KEY
Record_DS_subset (DS_subset_id)
    REFERENCES DS_subset (DS_subset_id);

-- End of file.
```

F. Concept Match Library – SQL ‘Drop’-Script

```
-- Created by Vertabelo (http://vertabelo.com)
-- Designed by Roel Veneberg
-- Script type: drop
-- Scope: [tables, references, sequences]
-- Generated at Tue May 27 10:41:47 UTC 2014

-- foreign keys
ALTER TABLE Concept_Match DROP FOREIGN KEY Concept_Match_Property;
ALTER TABLE Concept_Match DROP FOREIGN KEY Concept_Match_Record;
ALTER TABLE Concept_Match DROP FOREIGN KEY Concept_Match_Time;
ALTER TABLE DS_subset DROP FOREIGN KEY DS_subset_DS;
ALTER TABLE EA_subset DROP FOREIGN KEY EA_subset_EA;
ALTER TABLE Object DROP FOREIGN KEY Object_EA_subset;
ALTER TABLE Property DROP FOREIGN KEY Property_Object;
ALTER TABLE Record DROP FOREIGN KEY Record_DS_subset;

-- tables
-- Table Concept_Match
DROP TABLE Concept_Match;

-- Table DS
DROP TABLE DS;

-- Table DS_subset
DROP TABLE DS_subset;

-- Table EA
DROP TABLE EA;

-- Table EA_subset
DROP TABLE EA_subset;

-- Table Object
DROP TABLE Object;

-- Table Property
DROP TABLE Property;

-- Table Record
DROP TABLE Record;

-- Table Time
DROP TABLE Time;

-- End of file.
```

- Last page – intentionally left blank -