

MASTER'S THESIS

**Testing the definition of the Gap of Changes visualized in ArchiMate
Using a case study replacing legacy by ERP and BOB**

Dijkstra, Richard

Award date:
2018

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy

If you believe that this document breaches copyright please contact us at:

pure-support@ou.nl

providing details and we will investigate your claim.

Downloaded from <https://research.ou.nl/> on date: 09. Sep. 2021

Open Universiteit
www.ou.nl



Testing the definition of the Gap of Changes visualized in ArchiMate

Using a case study replacing legacy by ERP and BOB

Cursus: IM9806 Afstuderen BPMIT

Student: Richard Dijkstra

Datum rapport: 13-09-2018

Versie nummer: 1.0

Status: Final

Opleiding: Open Universiteit, Faculteit Management, Science & Technology

Testing the definition of the Gap of Changes visualized in ArchiMate

Using a case study replacing legacy by ERP and BOB

Cursus: IM9806 Afstuderen BPMIT
Student: Richard Dijkstra
Identiteitsnummer: 834795290
Datum rapport: 13-09-2018
Versie nummer: 1.0
Status: Final
Opleiding: Open Universiteit, Faculteit Management, Science & Technology
Afstudeerbegeleider: Mevr. Dr. Ir. Ella Roubtsova
Meelezer: Dr. Ir. Stef Joosten

Table of contents

ABSTRACT	6
1 INTRODUCTION.....	7
2 RESEARCH PROBLEM, QUESTIONS AND METHODS.....	7
3 IMPROVEMENT OF THE DEFINITION OF A GOC	8
3.1 THE ARCHIMATE LANGUAGE	8
3.2 STYLES OF USING THE ARCHIMATE LANGUAGE AND COMPLEXITY OF DIAGRAMS.....	11
3.3 MANAGING ARCHIMATE DIAGRAM COMPLEXITY	13
3.4 THE GAP OF CHANGES (GOC) CONCEPT.....	15
3.4.1 <i>Definition in the original paper</i>	15
3.4.2 <i>Explanation of ‘replaced by’ and ‘extended by’</i>	16
3.5 ANALYSIS AND IMPROVEMENT OF THE GOC CONCEPT.....	16
3.5.1 <i>Benefits to the architect</i>	16
3.5.2 <i>Border elements instead of unchanged elements.</i>	17
3.5.3 <i>Forbid to use specialization in the GOC</i>	17
3.5.4 <i>Relation between an Object and Relation is not modelled</i>	17
3.5.5 <i>New relationships colour is optional</i>	17
3.5.6 <i>Proposal definition of an improved GOC</i>	18
4 SOFTWARE FOR BUILDING A GOC.....	19
5 TESTING THE USABILITY OF THE IMPROVED GOC.....	20
5.1 MODELLING PROCESS	20
5.2 HOW THE CASE MODEL IS BUILT.	21
5.3 ASIS BUSINESS LAYER	22
5.4 TOBE BUSINESS MODEL	24
5.5 ASIS APPLICATION MODEL	24
5.6 TOBE APPLICATION MODEL	25
5.7 GOC APPLICATION LAYER.....	26
5.8 NUMBER OF CHANGES	27
5.9 IMPROVED GOC	27
5.10 WORKSHOP SETUP	28
5.11 WORKSHOP RESULTS	29
6 DISCUSSION AND REFLECTION.....	31
7 CONCLUSION AND FURTHER RESEARCH.....	32
APPENDIX A GAP OF CHANGES (GOC)	33
OBSOLETE OBJECTS.....	33
OBSOLETE OBJECTS.....	33
CHANGED OBJECTS.....	34
UNCHANGED OBJECTS.....	34
NEW RELATIONS.....	35
OBSOLETE RELATIONS.....	36
EXTENDED BY RELATIONS.....	37
REPLACED BY RELATIONS	38
BORDER RELATIONS.....	38
APPENDIX B IMPROVED GOC.....	39
APPENDIX C. SCORING FORM.....	41

APPENDIX D. PARTICIPANT RESULT CODING	42
APPENDIX E. FINANCIAL AND SUPPLY CHAIN SYSTEM REPLACEMENT	43
ASIS, FINANCIAL AND SUPPLY CHAIN SYSTEM.....	43
TOBE, FINANCIAL AND SUPPLY CHAIN SYSTEM	44
GOC, LEGACY FINANCIAL AND SUPPLY CHAIN SYSTEM	44
LIST OF FIGURES.....	45
LIST OF TABLES.....	45
REFERENCES	45

Abstract

Enterprise architecture (EA) is a method for dealing with IT complexity. For visualization of changes in EA ArchiMate proposes a Gap concept as a static model like the AsIs and ToBe models. The Gap of Changes (GOC) has been proposed by Bakelaar et al. (2017) for more precise change views in ArchiMate. The definition includes new ‘replaced by’ and ‘extended by’ relationships that are used in views relevant for stakeholders having a role in the change. The validation of this GOC on other cases was desired as future work.

The validation shows that the GOC has added value when modelling large architectural changes over using only the AsIs and ToBe models and it let the architect check the changes in the model with a change focus instead of a state focus.

This research is a theoretical and practical validation of the GOC in case of replacing legacy systems by an ERP and BOB solution in a food distributor company. After a theoretical review, the creation of the EA model and the setup of a validation workshop is explained. The first result of this research is the actual application of the GOC in a legacy replacement case. The second result is an extension of the definitions of the GOC with a subset comprising only change related objects. The third result is the software created for analysing the GOC that tend to be large due to the type of project. Finally application restrictions of the notion of a GOC are derived from the use case.

Keywords: ArchiMate Enterprise Model, Gap of Changes, Software for Gap of Changes Analysis, Case of replacing Legacy system with ERP and Best Of Breed.

1 Introduction

Many IT related projects fail because of complexity issues caused by the need of addressing different perspectives of an enterprise. Enterprise architecture (EA) is a method for dealing with this complexity. One of the notations used in the EA domain is the ArchiMate language (ArchiMate 2017), which uses different views to abstract complexity. For visualization of changes ArchiMate proposes the Gap concept that does not describe the change precise.

A new concept is the ‘Gap of Changes’ (GOC) defined by Bakelaar et al. (2017) for views using ArchiMate expressing change. The GOC is a view derived from the AsIs (current) and ToBe (desired) architecture comprising obsolete, changed, unchanged, new elements and their relations in the context of the change. The definition includes the new ‘replaced by’ and ‘extended by’ relationships that are used in a view that is relevant for stakeholders. The GOC is illustrated on application and business views on a case of replacing an Enterprise resource planning (ERP) with best of breed (BOB) solution. The validation of this view and use on other cases was desired as future work because only one exists yet.

This report presents an analysis and testing of the GOC as proposed by Bakelaar et al. (2017) within an application landscape where the initial legacy architecture is replaced by a desired architecture that comprises an ERP (e.g. SAP, Oracle or Microsoft Dynamics) system combined with Best-of-Breed (BOB) solutions (e.g. Microsoft CRM, Stibo Master Data Management).

The analysis is done by critical literature review, GOC definition review and by modelling the AsIs, ToBe and their relation in the GOC. Based on this analysis the definitions of the GOC has been extended and criticised.

Quantitative testing is done by a workshop with several participants testing the created views.

Chapter 2 formulates the research problem, research questions and the research methods. In chapter 3 the definition of the GOC is analysed and in chapter 4 the new developed software is described. Chapter 5 describes how the case model is built and the quantification of the testing of the GOC. Chapter 6 and 7 are the discussion, reflection, conclusion and possible subjects for further research. Detailed result data comprises Appendix A thru D.

2 Research problem, questions and methods.

The view called GOC defined by Bakelaar et al. (2017) is not defined in the ArchiMate standard and only a single application of the GOC exists. Therefore the definition and value of the GOC need to be analysed and tested using practical cases.

Main research question is:

How is the Gap of Changes applied in a change from legacy to ERP/BOB?

For quantifying the added value of the GOC the following research questions will be answered using the indicated method.

1. What are the existing definitions of the GOC in ArchiMate related literature?

Method to answer this question is a critical literature review, including

- Describing the ArchiMate language,
- Analysis of the definition of GOC and

- Experiencing the GOC by modelling the case.

2. What is a method of building of a GOC?

This question has been answered and validated with a case study in which a real system change is modelled using ArchiMate and the GOC. The system change include the replacement of a legacy system with an ERP/BOB solution and is modelled using an AsIs, ToBe and GOC view. The modelling using the original ArchiMate Gap definition is not executed because this is expected to have little added value.

3. How can we quantify the recognition of changes?

This research question has been answered with a workshop where the participants are asked to find changes in (1) AsIs, ToBe and in (2) AsIs, ToBe and GOC. The results have been measured and analysed by coding the changes that are found. This gives a total for each participant of the changes they found listed by GOC sub-set.

Hypothesis : The GOC can be applied and has added value when modelling an legacy to ERP/BOB change. The GOC will comprise mainly application components and application functions.

3 Improvement of the Definition of a GOC

3.1 The ArchiMate language

ArchiMate is an enterprise architecture modelling language based on UML2 (Lankhorst, 2009) . Its reason for existence is the need for communication about enterprise architectures (Jonkers et al. 2006). Its latest definition is held by the Open group (ArchiMate, 2017) which governs the development of the language. ArchiMate looks to be more IT-oriented due to the applications and experiences in which ArchiMate has been practiced. It is a relatively young language compared to the birth of EA and has experienced acceptance difficulties. According Foorthuis et al (2016) the most important difficulty is little benefit of enterprise architecture in general, tools related to this domain will suffer the same. Also, enterprise architects tend to work isolated without much added value (Buschmann 2009; Wierda 2015). ArchiMate has been designed to strengthen the EA function and might contribute to the success of EA.

The latest extensions of the ArchiMate language include the modelling of strategy (Aldea et al 2015) and capabilities (Azevedo et al, 2015). These papers did result in the version 3 of the language (ArchiMate 2017). This version 3 super seeds version 2.1. In this report, a selected part of the available ArchiMate 3.0 components is used.

ArchiMate has the characteristics that are explained in a dimensions diagram in figure 1 and an example in figure 2. The ArchiMate diagrams have both a vertical and horizontal direction. The vertical direction from bottom to top expresses the structure of the system (technology, application, business). The horizontal direction from left to right expresses the aspects of the system: passive,

behaviour and active structure. The ArchiMate specification use colours for indicating the architecture layers as shown in figure 1. The core elements of ArchiMate are part of the business, application and technology layer.

Yellow, business-layer comprising business processes and business objects like ‘invoicing’ process and ‘invoice’ business objects.

Blue, application and data layer comprising the application (non-system software) part of IT systems like data objects (e.g. ‘product’ data object) and custom-made software modules.

Green, technology layer comprising the basic infrastructure like nodes (server XYZ) and generic software like email.

The physical, implementation & migration and motivation layers are non-core layers.

Let’s explain the ArchiMate diagrams with an example. Figure 2 is an example of the business layer of ArchiMate from the case subject of this paper.

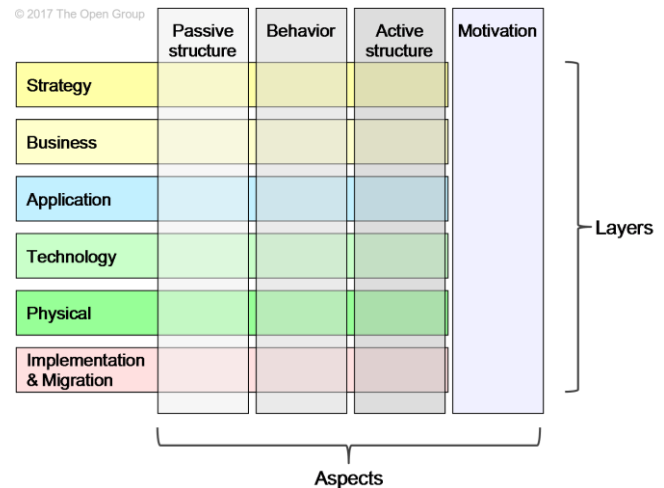


Figure 1. Full ArchiMate framework from ArchiMate (2017)

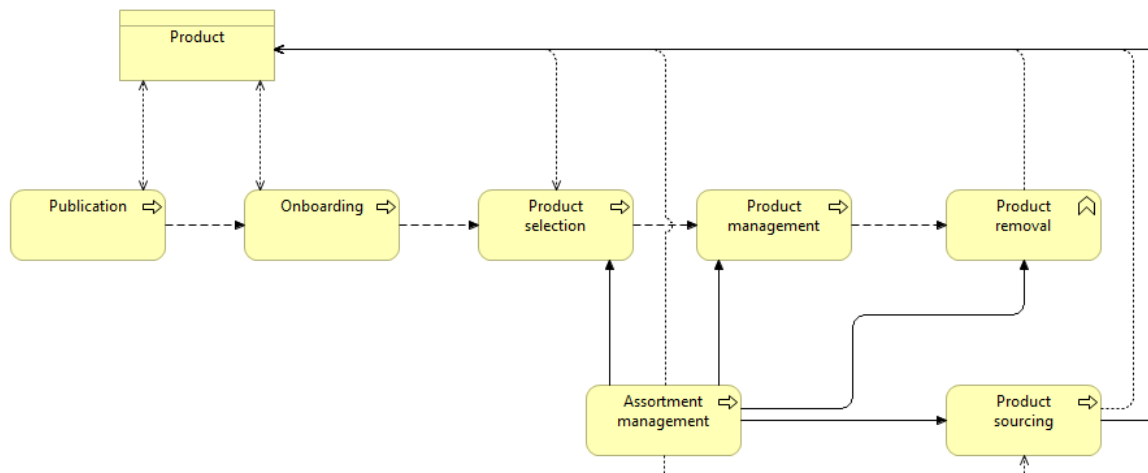


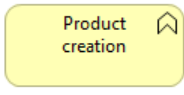
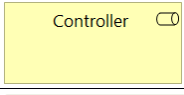
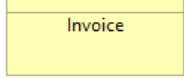
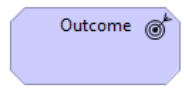
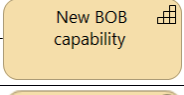
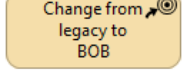
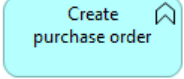
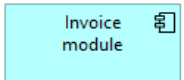
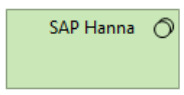
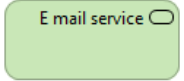


Figure 2. Business layer fragment from case study

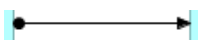
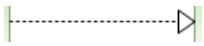
The product data management process start when the supplier publishes the product data visualized in the process ‘Publication’. This product data flows (dashed arrow) to the ‘Onboarding’ process where we store the product data in a supplier managed database. Next assortment management employee selects the product for automatic ordering in the ‘Product’ selection process. The product ‘Business object’ is accessed in both directions (read and write) by several business processes. Information (product data) flows from the business process ‘Product selection’ to ‘Product management’. Table 1 and 2 show ArchiMate elements (ArchiMate, 2017).


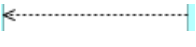

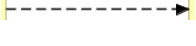


Table 1. ArchiMate elements explained

Component	Meaning (by example)
 Supplier	Business actor being the supplier of the products to the retailer.
 Product publishing	Business process modelling the activity the supplier performs for publishing the product data. Publishing is done on the publishing platform
 Product creation	Business function is a specific behaviour of the organization and can be related to a business process. Also specific competencies are modelled using business functions.
 Controller	A role is a specific behaviour of a business actor, e.g. a business controller.
 Invoice	A business object is a unit of information relevant to the organization.
 Outcome	Outcomes are high-level, business-oriented results produced by capabilities of an organization.
 New BOB capability	Capability of an element or an outcome
 Change from legacy to BOB	A Course of action for intervention for a certain goal.
 Create purchase order	Application service which is serviced by an application component
 Invoice module	Application component which represents a collection of software applications.
 SAP Hanna	Technology system software are software components close related to infrastructure of standard commodity (standard) software.
 E mail service	Technology service created by system software

Relating elements is done within ArchiMate through relationships. The following table explains the basic relationships.

Table 2. ArchiMate relationships explained

Relationship	Name and Meaning
	Assignment. The assignment relationship expresses the allocation of responsibility of execution.
	Realization. The realization relationship indicates that an entity plays a critical role in the creation, achievement or operation of a more abstract entity.

	Composition. The composition relationship is the strongest relation. It shows that the two components cannot exist without the other. The association relation is the weakest relation.
	Access. The access relationship relates data objects to behavioural components.
	Serving. The serving relation is used when a behavioural element serves an element in an element in a higher layer, e.g. an application service serves a business process.
	Flow. The flow relation represents the flow of information between elements.
	Trigger. The trigger relation indicates a causal relation between processes, functions, and events.
	Specialization. The specialization relation models the design that a component is a specialization of another component.

The Gap concept within ArchiMate (ArchiMate, 2017) is a description of the differences between two plateaus. Components in the AsIs and ToBe may be associated to this gap. The result is not a unambiguous description of the actual planned change.

3.2 Styles of using the ArchiMate language and Complexity of diagrams

Because ArchiMate is a language for modelling architectures, different architects can have different styles in certain conversation contexts. Figure 3 presents a generic ArchiMate fragment within an application layers. It is rather detailed combination a Data Object, Application function, Service, Interface and Component that may lead to complex diagrams.

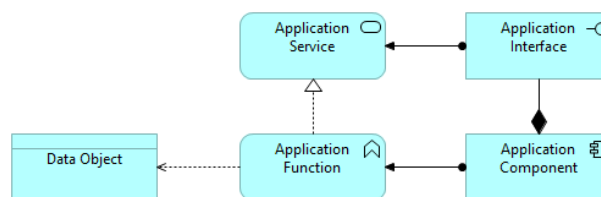


Figure 3. Typical ArchiMate fragment

Figure 4 presents an example combining elements of different layers: Technology system Windows 10 serves application services Excel that realizes Mortgage calculation service for the process Mortgage calculation. Modelling the Application service in a view is the explanation how this service is created. Therefore the service can be used a different places without repeating the model of the service. A business or an application service in itself has no content but is a place holder for underlying service creating objects like interfaces and functions. Therefore a service will only change when a service creating object changes or is replaced. This will lead to a changed service and not a replaced service.

Both figures show that EA is always an abstraction showing the elements of structure, vision, and evolution of the organization, business processes, information systems and infrastructure. It is not a model of all the IT functionality; therefore UML tools exist like sequence diagram and state diagram.

The EA repository is the set of objects and relationships that comprise the EA. Objects from the repository are used when creating the view because the view visualizes a subset of the objects and relationships for a special communication purpose. A view can be limited in expressing changes; therefore a list in human language will always be an option.

A model is an abstract, conceptual system by which a real-world system is represented (Schwaninger, 2010) by concepts and relationships between these concepts. The size and amount of capabilities determine the enterprise architecture size. According Lankhorst (2009) this architecture content must be presented in relevant views for the architectural conversation which can be difficult because of diagrammatical complexity. The diagrammatical complexity includes according Moody (2009) perceptual limits meaning that discrimination between diagrams decreases with diagram size. Also, the number of elements the reader can process is limited by the limited working memory capacity.

The effect of a design conversation is according Maier and Hoffman (1967) the product of quality and acceptance. The acceptance depends on the ability to understand the information presented by an architectural view.

Another style of ArchiMate modelling includes also the motivation modelling. The complexity can be even increased with motivation views of the ArchiMate. Capability-based strategy bridges the gap between strategy and strategy implementation (Azevedo et al., 2015). Strategy takes an important part in business communication and is closely related to tactics. According Makey and Zundel (2017) the use of the concepts strategy and tactics is blurry and inconsistent. This blurring is also available in ArchiMate because ‘Courses of action’ can be categorized as strategies or tactics. Makey and Zundel (2017) conclude that the pragmatic relationship between strategy and tactics is as formulation versus implementation or thinking versus action. Because of this, any plan for a communication pattern can be called strategic. In the context of a solution architecture, that must manage the complexity of a business change involving IT, the strategy sets the context for a solution for an issue (strategic gap). This context contains requirements for a ‘course of action’ of one or more changing capabilities because the change of the capability is the forecasted solution of the perceived issue. The motivation of this issue is the description of the issue and the impact it has on the organization. This motivation can have different forms depending on the models stakeholders use.

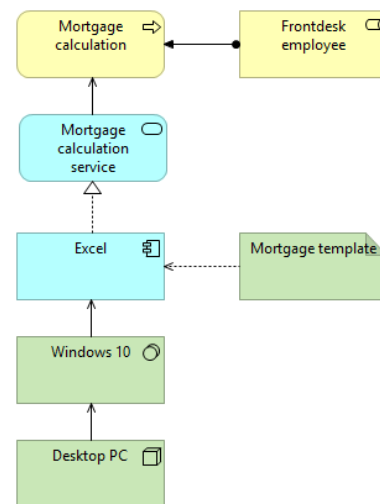


Figure 4. Excel sheet model

Because this research is about solution architecture, the use of motivation views is limited as described in the GOC meta model by Bakelaar et al. (2017)

3.3 Managing ArchiMate diagram complexity

When using ArchiMate several techniques can be used for improving an ArchiMate view in order to gain maximum effect.

Abstraction

Abstraction is the process of modelling generic concepts which represent simplified components in the real world. The use of ArchiMate is visualising these abstractions. Abstractions may have different levels resulting in more or less details of the design.

Nesting

Sometimes the relations disturb the essence of visualization. This can be prevented by gathering related component in such a way that the relations are clear without drawing the relationships as lines. Figure 5 shows 2 constructs with the same meaning but different visual experience. The left construct is nested, the right is not nested.

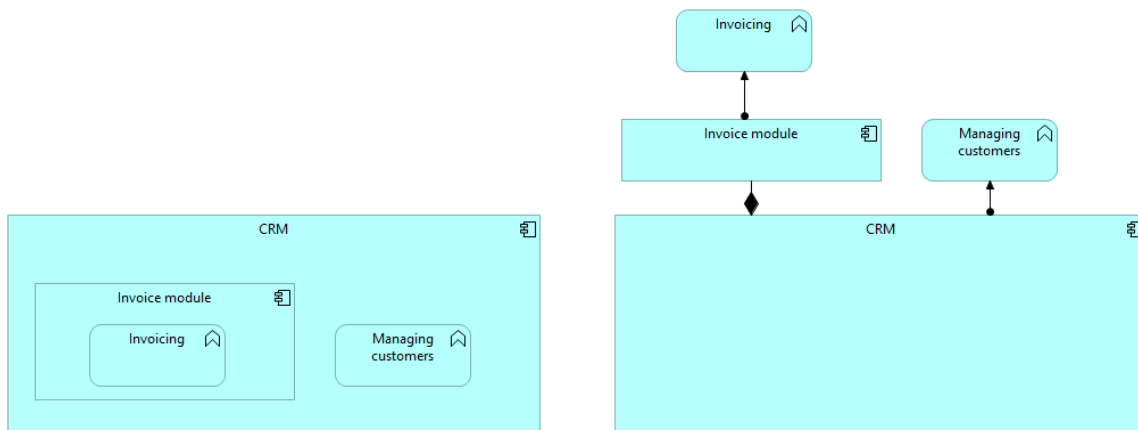


Figure 5. Nesting ArchiMate elements

Figure 5 view expresses that the ‘Invoice module’ is assigned to the ‘Invoicing’ application function which is part of the CRM application. Also the CRM application is assigned to the ‘Managing customers’ function.

Grouping

A special sort of nesting is grouping. Grouping in ArchiMate gives insight by grouping elements with a common feature. The architect may even give a self-explaining name to the group and remove the elements from the view. Figure 6 shows the reducing diagram complexity by showing three elements as one (ArchiMate 2017). The top-right show all existing structural relationships that are not visible in the other views.

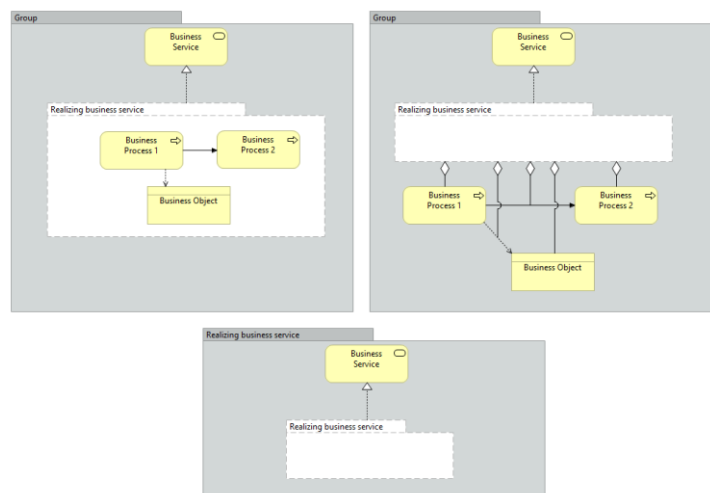


Figure 6. Grouping as diagramming technique

Derived relations

Taking shortcuts is also an ArchiMate method for limiting the number of elements while staying correct in terms of ArchiMate conventions. Relationships have strength as described in Archimate chapter 5 (2017), ranging from association being the weakest to composition which is the strongest. Derived relationships are the weakest within a given design. A widely use case for applying derived relations is shown in figure 7 visualising the service creation by an application component. Here the realization relationship remains because this relationship is weaker than the assignment. Also the derived relation of serving a business process by an application function is shown.

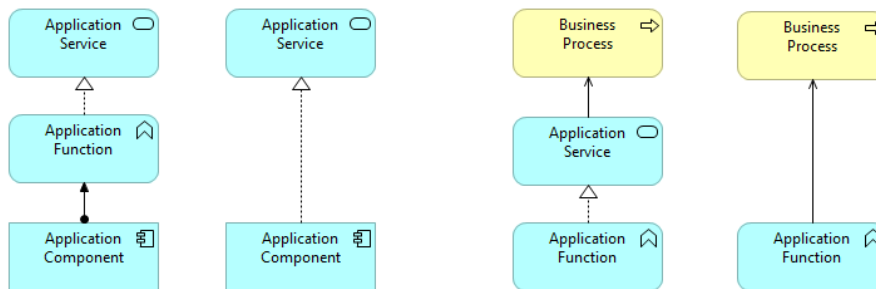


Figure 7. Derived relations

Different layers for Standard and modified software

From risk management point of view, an important project characteristic is the amount of standard software. ArchiMate can be used for applying a difference between standard and modified software in two different layers; standard software in the technology layer and modified or in-house made software in the application layer. This is a modelling style that should be agreed upon within an organisation. In this paper this style is not used.

Figure 8 illustrates the separation of the standard and modified software.

Colouring

ArchiMate has a standard colour scheme. It might be useful for effective pinpointing certain characteristics that extra colours are used. Bakelaar et al. (2017) did use this freedom when e.g. indicating changed objects using the colour orange.

Adding explaining text

The meaning of Archimate elements is often difficult to remember. Therefore adding text to elements can improve diagram understanding. However the element is part of the architecture repository. The added text might be view-dependent and will be part of the view agnostic repository. Therefore only explanations of element semantics should be added. Explanations that are only relevant for one specific view should be added as text box independent of the related element.

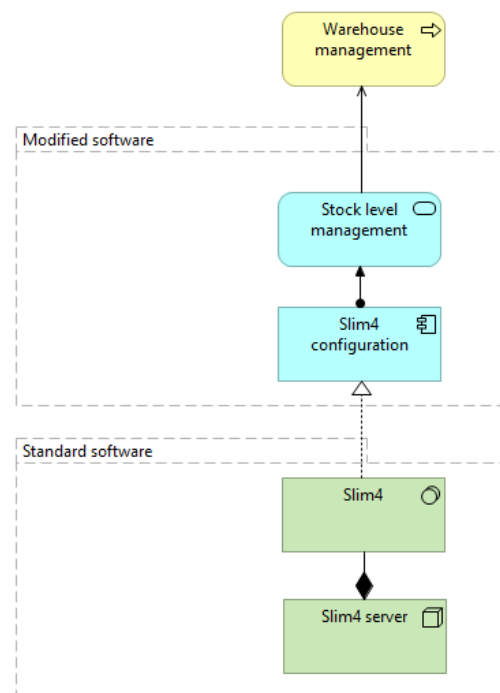


Figure 8. Distinction between standard and modified software

3.4 The Gap of Changes (GOC) concept

Bakelaar et al. (2017) described a framework for visualization of changes comprising two part: the logical definition of the GOC and the extension of ArchiMate relationships with ‘extended by’ and ‘replaced by’ relationship.

3.4.1 Definition in the original paper

The Gap of Changes is a tuple defines as follows:

$$GOC = (O_{obsolete}, O_{new}, O_{unchanged}, O_{changed}, \\ R_{obsolete}, R_{new}, R_{replaced-by}, R_{extended-by}, R_{border})$$

The objects are defined as follows:

$$O_{obsolete} = \{ o \mid o \in O_{asis} \cap o \notin O_{tobe} \}$$

Obsolete objects are visualized grey.

$$O_{new} = \{ o \mid o \notin O_{asis} \cap o \in O_{tobe} \}$$

New objects are visualized green.

$$O_{unchanged} = \{ o \mid o \in O_{asis} \cap o \in O_{tobe} \text{ and } \forall x: (o, x) \in R_{tobe} \Leftrightarrow (o, x) \in R_{asis} \}$$

Unchanged objects are visualized in the original ArchiMate colour.

$$O_{changed} = (O_{asis} \cap O_{tobe}) \setminus O_{unchanged}$$

Changed objects are visualized orange.

The relationships are defined as follows:

$$R_{obsolete} = \{ (a, b) \mid (a, b) \in R_{asis} \text{ and } (a, b) \notin R_{tobe} \}$$

Obsolete relations are visualized by grey arrows.

$$R_{new} = \{ (a, b) \mid (a, b) \notin R_{asis} \text{ and } (a, b) \in R_{tobe} \}$$

New relations are like visualized by green arrows.

Note that new relations does not imply always new objects.

$$R_{replaced-by} \subseteq O_{obsolete} \times O_{new}$$

$$R_{extended-by} \subseteq O_{changed} \times O_{new}$$

Both replaced and extended-by relations are black.

$$R_{border} \subseteq (O_{unchanged} \times O_{changed}) \cup (O_{changed} \times O_{unchanged})$$

The border relation is visualized in green.

Because the relations are part of a directed graph the sequence of the related nodes is relevant.

3.4.2 Explanation of ‘replaced by’ and ‘extended by’

Bakelaar et al. (2017) defined new relationships ‘replaced by’ and ‘extended by’ for explicit visualising change.

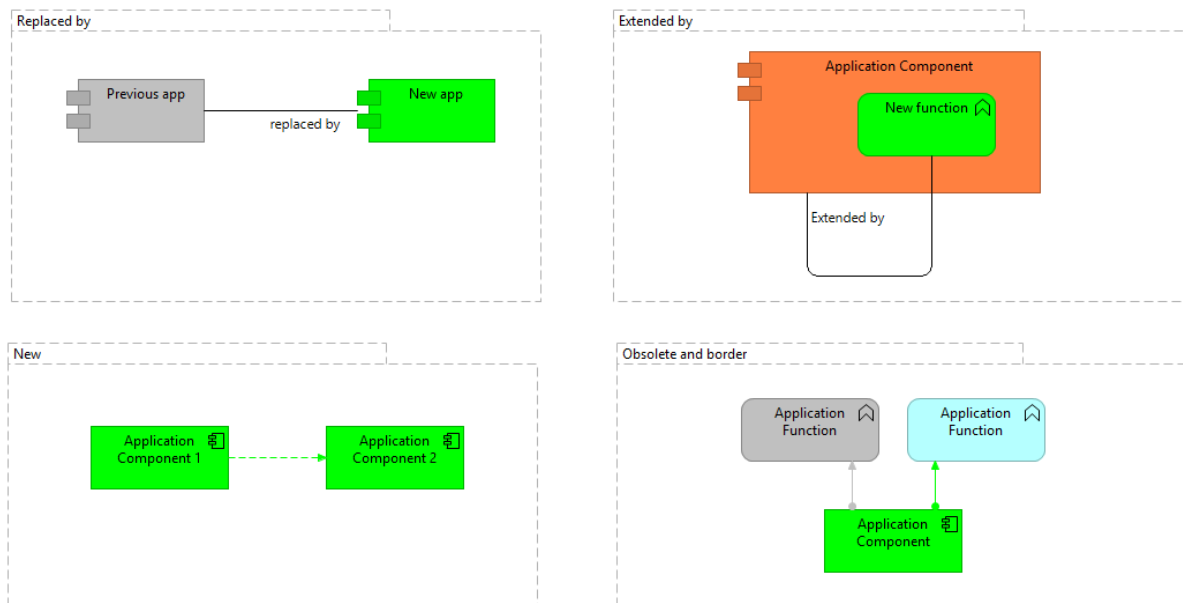


Figure 9. GOC relations

In figure 9 the replaced-by example expresses that the ‘Previous App’ is ‘replaced-by’ the ‘New App’. The extended-by example gives the Application component a New function. In the new example both application components are new including the flow relation between them; in this case no relation exists with the AsIs design. The last example is an Application component that has 2 application functions in the AsIs design. In the ToBe design the grey object is obsolete while the other application function is unchanged and is qualified as ‘unchanged object’ in the Bakelaar et al. (2017) definition.

This unchanged object is a border object (same colour) in the proposed extension, assigned via a border relation (green).

Considering an object as obsolete that will be replaced triggers an architectural conversation comprising the removal of that component meaning that it can be designed from scratch; however that will not be the case in most cases because ‘some’ system is also using this object, e.g. data-object, because ‘some’ system might not be part of the repository. This is especially interesting for the data object because data can be used easily the shadow IT systems which become more and more important according Haag and Eckhardt (2017).

3.5 Analysis and Improvement of the GOC concept

3.5.1 Benefits to the architect

The proposed method by Bakelaar et al (2017) is besides the standard modelling ArchiMate methods of great benefit because the architect needs to be more precise during the architectural conversations because an exact replacement and extension design can be made.

3.5.2 Border elements instead of unchanged elements.

The GOC definition results in a large set of objects and relations in the EA repository comprising the GOC because the set of unchanged objects may be large compared to the change. Using a subset of objects and relations of the GOC that are directly related to the change will overcome this issue of redundancy. In this report, it is proposed to call this the ‘improved GOC’ comprising the concept of border object and including this concept in definition of changed objects in order to create a definition without all unchanged objects.

Another reason for adding the concept of “border object” is because when 2 or more changes exist in a repository and 2 changes have a shared border object, this might not be noticed visually. Analysing the repository will always reveal these shared objects.

3.5.3 Forbid to use specialization in the GOC

A second addition to the GOC is the constraint not using specializations because when adding a member to a generic type does not change the generic type so this object is not changed. This cannot be said about the other relationships.

3.5.4 Relation between an Object and Relation is not modelled

An optional change to either the GOC is due to the introduction in ArchiMate (2017) the possibility to model a data object associated to a (e.g. flow) relationship. This gives the challenge to model a relation from an object to a relation e.g. given object a, b, c and relationship (a,b), the relationship to a relationship will be (c,(a,b)). This is an option for future research and will not be subject to this research.

3.5.5 New relationships colour is optional

The GOC view in figure 20 (and during the workshop) did not have the prescribed colours for new relationships being green, because when a new element is nested in a new element the relation will not be seen because they have the same colour. The option to un-nest the new elements makes the view as a whole messy as show in figure 16. This is also the case for border elements that are nested in changed objects. The border relation itself is not visible, but un-nesting will make the view messy again and the orange line will not be visible within the changed object.

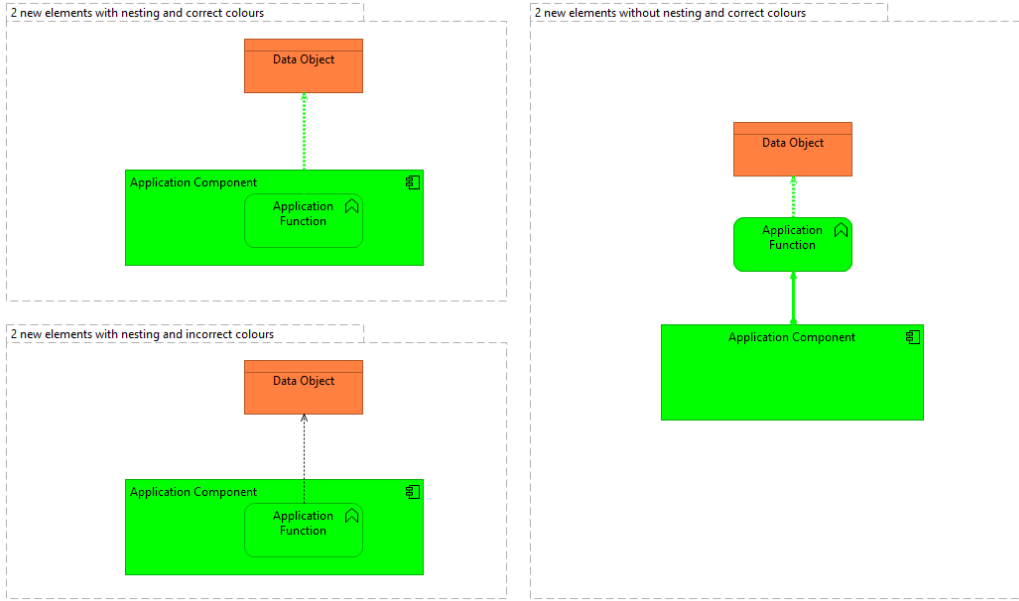


Figure 10. Colour and nesting issue

This will not always be an issue because a new object (green) nested in a changed object (orange) will be visually fine.

3.5.6 Proposal definition of an improved GOC

The GOC of change is tuple including more elements than the net change. Views needs to be created for abstracting the net change from the GOC for communication to stakeholders. The definition of the net change can be used for analysis and communication. The following definition is proposed.

Definition of an improved GOC:

Given from GOC: $O_{tobe}, O_{asis}, O_{new}, O_{obsolete}, R_{tobe}, R_{asis}$,

$$T = \{ o \mid o \in O_{tobe} \cap o \in O_{asis} \}$$

This leads to the definition of changed objects:

$$O_{changed} = \{ o, x \mid o \in O_{asis} \cap o \in O_{tobe} \text{ and}$$

$$\exists x: (o, x) \in R_{tobe} \Leftrightarrow (o, x) \in R_{asis} \text{ and}$$

$$\exists x: (o, x) \in R_{tobe} \Leftrightarrow (o, x) \in R_{asis} \}$$

Meaning an object is changed when it is part of AsIs and of ToBe and has at least one relation in both AsIs and ToBe and has at least one relation that is part of ToBe and not of AsIs or part of AsIs and not of ToBe.

Unchanged objects are part of AsIs and ToBe and are not part of the set changed objects.

$$O_{unchanged} = (T \setminus O_{changed})$$

The existing concept of border relation is defined by relations between changed and unchanged objects taking the direction of the relation into account:

$$R_{border} = \{(a, b) | (a \in O_{unchanged} \cap a \in O_{changed}) \cup (a \in O_{changed} \cap a \in O_{unchanged})\}$$

The new concept of border objects is defined by unchanged objects having at least one border relation.

$$O_{border} = \{o | o \in O_{unchanged} \text{ and } \exists x: (o, x) \in R_{border}\}$$

With the above definition the following definition of the improved GOC is proposed:

$$GOC' = (O_{obsolete}, O_{new}, O_{border}, O_{changed}, R_{obsolete}, R_{new}, R_{replaced-by}, R_{extended-by}, R_{border})$$

4 Software for building a GOC

The size of the GOC appeared to be large and hardly to count manually. Extra software for analysing the GOC appeared to be rather practical and easy to create. The software is written as part of this research in Python (Python, 2018) using a Jupyter Notebook (Jupyter, 2018) and available on GitHub (Dijkstra R., 2018). The software analyses the repository through exports from the modelling tool Archi (Archi, 2018). Two test sets are available, not the case subject of this study and how to use this software is explained on GitHub.

Because the complete system replacement many changes need to be managed in order to list all changes comprising the change. The repository has been checked using a Python Notebook (Dijkstra R., 2018) that helped to verify the model and specific objects and relationships that are exported to csv files and imported into the Notebook. The Notebook uses a simple data structures like dictionary and Pandas DataFrame (2018) that let the architect create selections, lists and other operations for gap analysis. Also the definitions used in Bakelaar et al. (2017) are validated using the software applying set theory. E.g. the new objects are defined as follows

$$O_{new} = \{o | o \notin O_{asis} \cap o \in O_{tobe}\}$$

In the Python Notebook this is checked with the following assertion using sets, where the symbol ‘==’ is used in Python for testing equality:

```
set_new_elements == set_tobe_elements - set_asis_elements
```

Another example is describing a feature of ‘replaced by’ relations as follows:

$$R_{replaced-by} \subseteq O_{obsolete} \times O_{new}$$

In the Python notebook the following set assertion verifies this:

```
In [134]: # This test confirms that the replaced by relations replace obsolete objects with new objects.
# -----
{(relations_dict[r].source, relations_dict[r].target) for r in set_replaced_by_relations} <= \
{*product(set_obsolete_elements, set_new_elements)}
Out[134]: True
```

Running the software against the repository helped to improve the model because isolated objects with no relations and multiple objects referring to the same real life object appeared. During analysis some inconsistencies between the AsIs, ToBe and GOC appeared. In fact these were no inconsistencies but unknown and undecided architectural choices. This can be described as differences in value and moments of communication of certain abstractions in architectural conversation strategies (Lankhorst, 2009). These conversation moments depend on the following without trying to be complete:

- Knowledge of the architect and stakeholder
- Completeness of the repository containing the raw design
- Created views for specific communication moments
- Interpretation of the stakeholder.

5 Testing the usability of the improved GOC

The test comprises the usage of the improved definition of a GOC as proposed. The testing method includes:

1. Building of the AsIs and ToBe models.
2. Building of the GOC supported by the created software module.
3. Counting all changes.
4. Conducting a workshop with stakeholders of AsIs and ToBe and counting the found changes.
5. Conducting a workshop with stakeholders of AsIs, ToBe and a GOC and counting the found changes.
6. Analysis of the added value of the GOC for stakeholders.

5.1 Modelling process

During the modelling as preparation of the workshop the following experiences revealed.

When replacing ‘complete’ systems the GOC might look like figure 11. But this will lead to a lot of extra ‘replaced-by’ relationships that will make the views crowded. Adding the extra relationships to the repository will be useful but using them in a view must be judged according to the rules of view clearness. This has been resolved as follows.

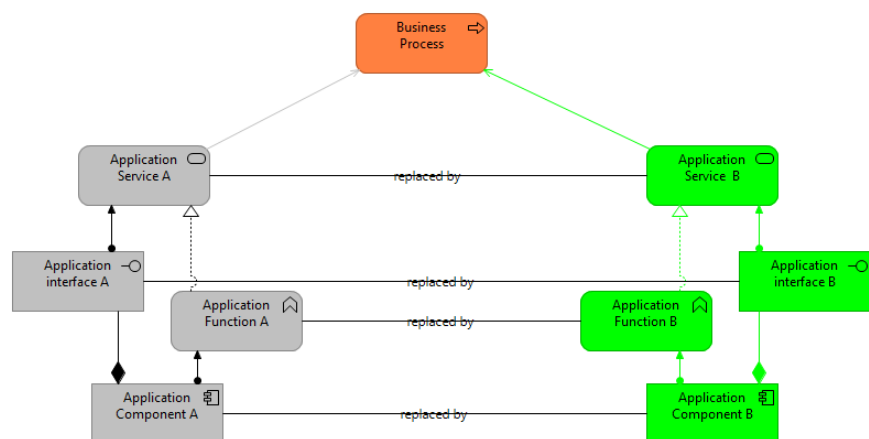


Figure 11. Full system replacement

Figure 12 shows the GOC for the application layer with minimal required ‘replaced-by’ relationships as used in the case study. According ArchiMate (2017) “a service is the externally visible behaviour of the providing system, from the perspective of systems that use that service”. Thus from the business process point of view the Application service X is changed because the business process served by this application service X, should be unchanged. This agreement is an example of ‘Way of modelling’ mentioned by Lankhorst (2009). These agreements are needed because the ArchiMate language in itself has too much freedom that will drive inconsistency and mis-communication.

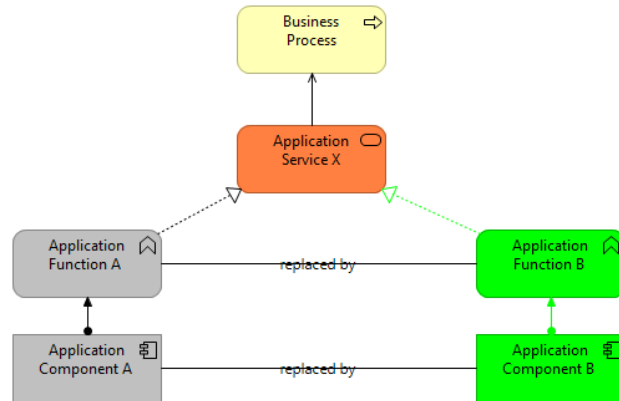


Figure 12. Used modelling convention

Using the new concepts of the ‘replaced by’ and ‘extended by’ relation created a change focus while I made the models. Questions like “Can this application function replace all of the obsolete function?” made me model more precise and aware of the dynamics of the change.

Adding the replaced by and extend by relationships create the responsibility towards colleague architects to model the project reality, e.g. replaced by components should be represented with the correct replaced by relations. An architecture standard could also define to removal of the obsolete elements and added extended by and replaced by relations because they are time dependent.

Technology is not really important compared to the application complexity because systems (PIM, SIAM, BOM, AS400) are simply replaced by others (STIBO + SAP). This can be written down as a list, the visual modelling has little benefit.

The data entities ‘Product’ can be modelled in two formats. The first is to model on a higher abstraction level as one entity. The second is product per namespace because in fact product exists in multiple namespaces (applications). The case has been modelled using at a higher abstraction level.

Sufficient modelling is possible using the same objects (retaining the same id in the repository) when these are unchanged or changed. Obsolete elements remain in the repository while new element, possible part of ‘replace with’ relation, are created. In the repository only new relations are created; obsolete are not removed. This is not a problem because deleting repository concepts would destroy historical information. An EA tool with version control would solve this issue.

The GOC view can be difficult to model as horizontal view. It is more natural to make it vertical because of routing tool does not automatically route correctly. Architects might get confused however because they are used to modelling architecture vertically. Nonetheless the GOC views are also vertically oriented in this case.

Last but not least did I have the experience that views become too crowded quite quick. Maximum of 20 elements is my experience to be useful. More than 20 is difficult to make and difficult to understand by the stakeholder. This confirms the perceptual limits Moody (2009) described.

5.2 How the case model is built.

The goal of the change is the replacement of (1) the legacy financial and supply chain systems by SAP S4 Retail and (2) the legacy master data management systems by a BOB master data

management solution Stibo. The change of the MDM system is used in the workshop and subject for the change counting. The change towards SAP is not analysed but the designs are shown in appendix E.

The goal of the change is shown in figure 13. The current architecture consists of legacy systems. The main reasons for choosing BOB and ERP application type is to prevent software development and use best practices collected in the software. But the change is constrained by continuing current practices like the publishing of master data (MD) by the trade product suppliers. Also the online platform ‘Sligro online’ and the data warehouse platform remains in the ToBe design. The driver for the change is ‘share holder value’ created by agility, growth and Cost control.

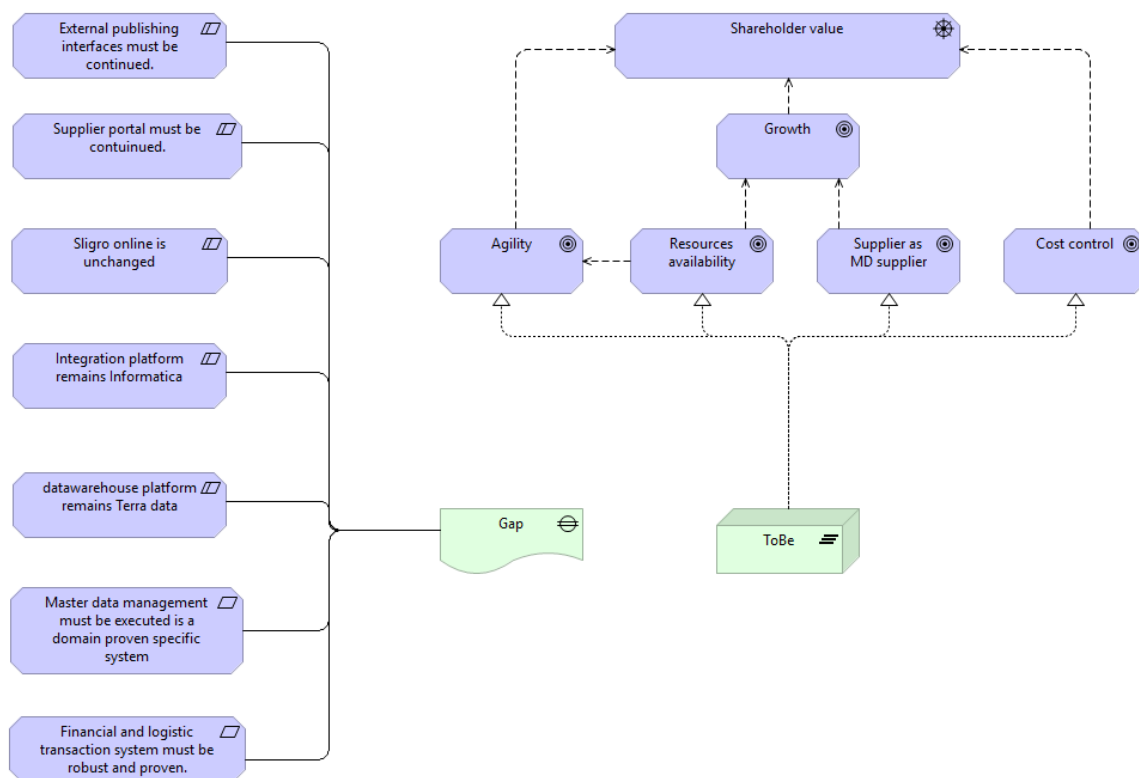


Figure 13. Motivation for implementing ERP and MDM

As the reader can see, the change is rather large. The change is input for program management that initiates projects with a certain scope. The GOC cannot be presented as a single view. The GOC of the application layer is used because applications are replaced with as less as possible impact for the primary process of the company.

The business process is shown for explanation of the application layers and is no subject for the GOC analysis.

Therefore the change is divided into two logical parts, the BOB changed and the ERP change. Shared components will be part of both views. Secondary processes like marketing are not in scope.

5.3 AsIs business layer

The scope of the case are two main processes: master data management and the primary process of the company. Master data management starts at the publication of the master data by the supplier himself, followed by data enrichment and maintenance as show in figure 14. The data publication is visualized

in ArchiMate using a 'Business process' symbol. This process reads and writes 'Product data' visualized by a 'Business object' called 'Product'. Information flows from the Publication process to the Onboarding process visualized by the dashed arrow.

This product data can be used in the primary process starting at the business process 'Product sourcing'. This sourcing is followed by 'warehouse management' and 'Product delivery'. Sourcing includes the ordering and receiving of products. 'Warehouse management' include the effective stock management while 'Delivery' represents the actual physical product delivery at the customer location.

Because the data object 'Product' has relation with all concepts; the relationships with 'Product' are not shown because of readability of the view.

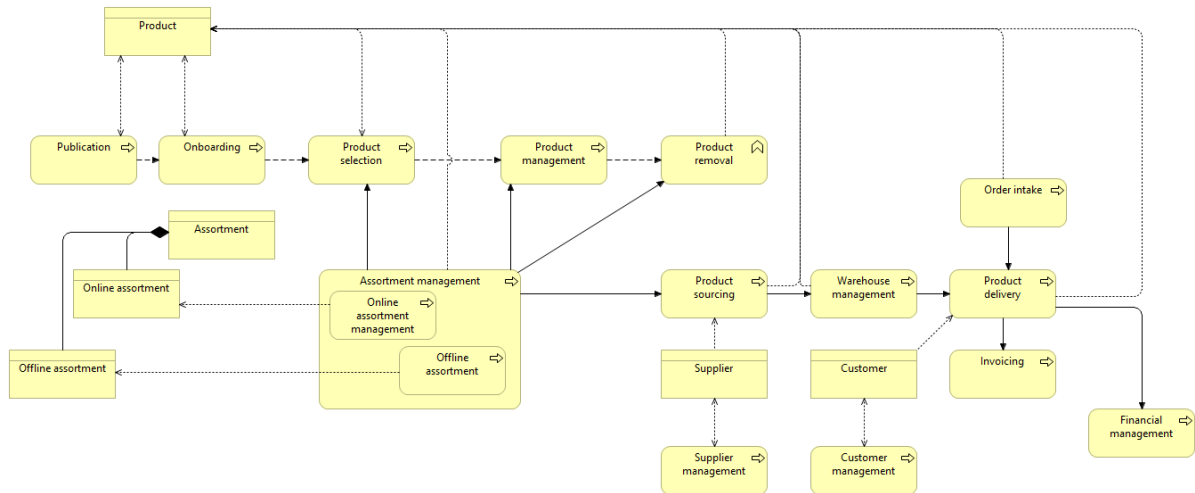


Figure 14. AsIs business processes for Master Data Management (MDM) and primary process

AsIs business processes will be optimised in the ToBe design by joining the online and offline assortment management business processes to one single Assortment management process as shown later.

5.4 ToBe business model

The ToBe model has little changes compared to the AsIs; only the difference between online and offline master data management is not present anymore in figure 15. The other processes are unchanged but will be serviced by other applications.

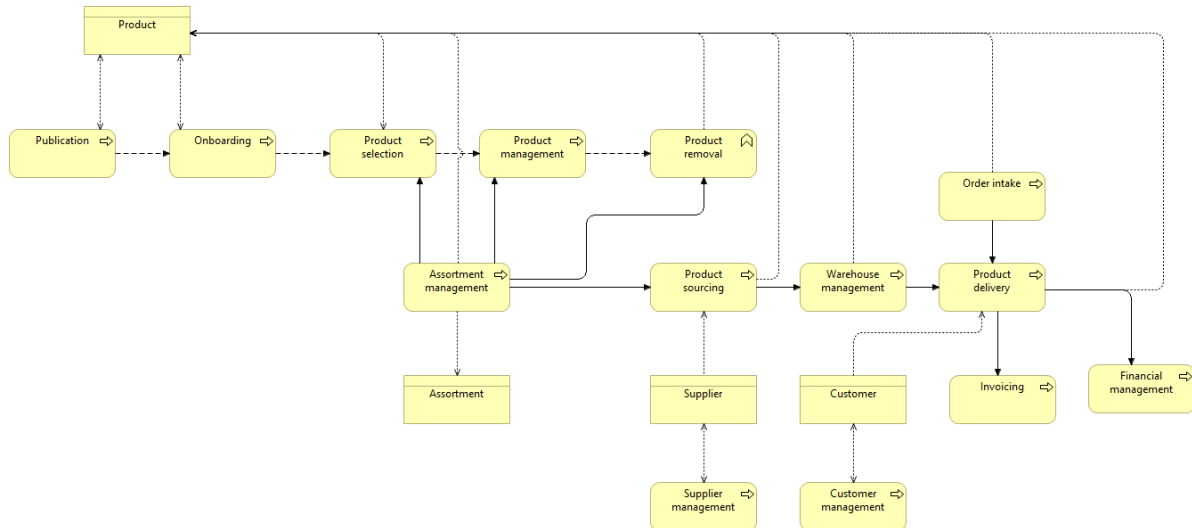


Figure 15. ToBe business processes for master data management and primary process.

5.5 AsIs application model

Figure 16 is the AsIs application model that does show the applications that take care of the mentioned business processes in figure 15. Because the size of the model is quite large, the view has been split into two views, one focusing on the MDM area and one of the legacy ERP area. The main parts are the Product Information Management system (PIM), system for purchasing and assortment management (SIAM) and the Integration platform that connects the legacy and the online platform. The 'leveranciers portaal' application function realises the application service 'Leveranciers portaal' that is used by product suppliers. The 'PIM interface' modelled in the integration platform services the communication to the 'PIM' application.

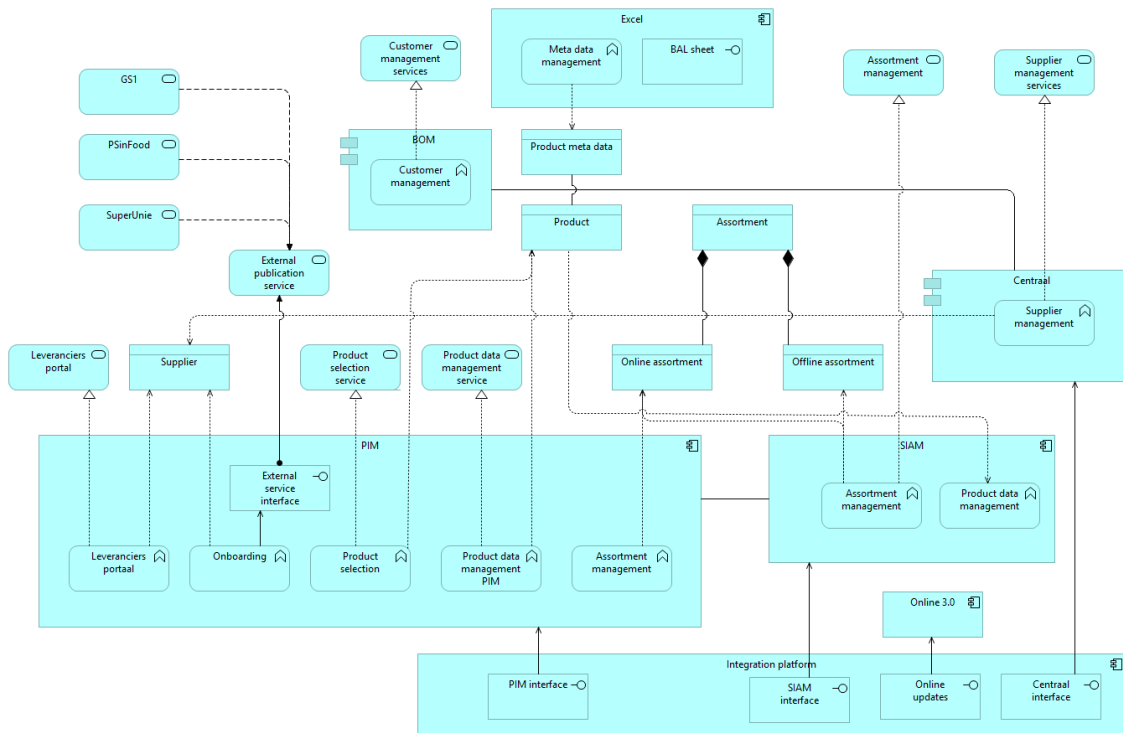


Figure 16. Application MDM AsIs view

5.6 ToBe application model

The required ToBe MDM architecture is shown in figure 17. The changes are mainly the introduction of the Stibo platform and SAP. SAP is modelled as a linking element to the other ERP in appendix E. The Integration platform remains in the ToBe design because of its current value to the business. Stibo delivers many application functions like ‘leveranciers portaal’ that is serviced by PIM in the AsIs design. ‘Assortment management’ application function services the assortment application service.

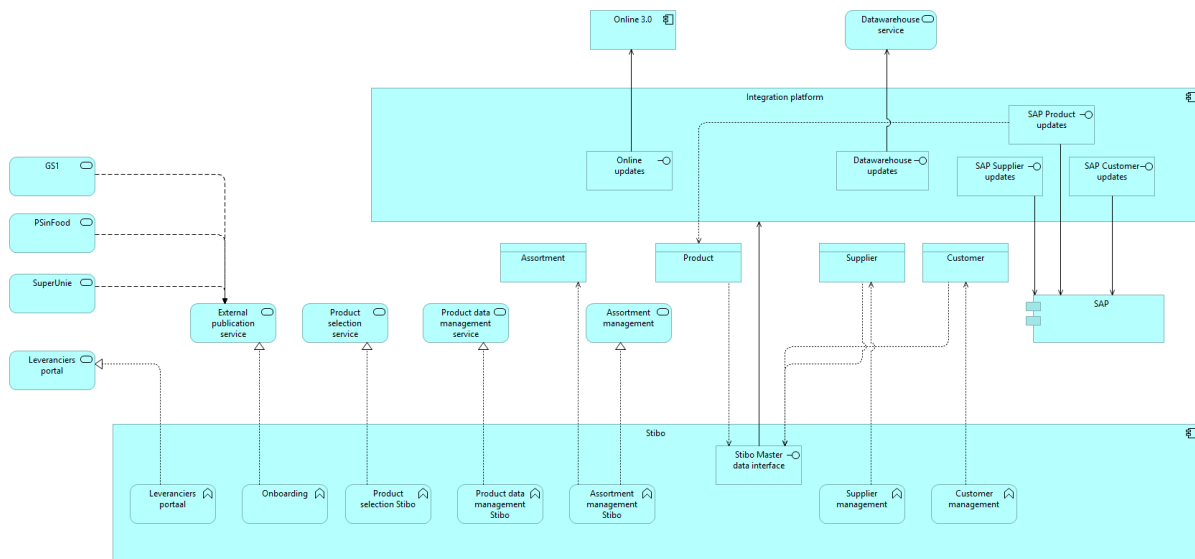


Figure 17. Application MDM ToBe view

5.7 GOC application layer

Figure 13 shows the improved GOC that has been used in the workshop. Creating many models for this paper led to a general applicable approach as a method for creating the GOC:

1. Create AsIs view relating all objects to the AsIs plateau
2. Create ToBe view relating all objects to the ToBe plateau
3. Copy layer from AsIs view (retaining id's) into a the GOC view
4. Copy layer from ToBe view (retaining id's) into this GOC view
5. Colour obsolete, new and changed objects in a creative procedure
6. Remove double objects (having the same objects id) and visualise relations from AsIs and ToBe
7. Remove objects that are unchanged and are no border objects
8. Colour the relations to the new objects if needed
9. Colour the relations to the obsolete objects if needed
10. Create extended by relations and probably nest the objects
11. Create replaced by relations

These steps have also been used for creating the GOC in figure 18.

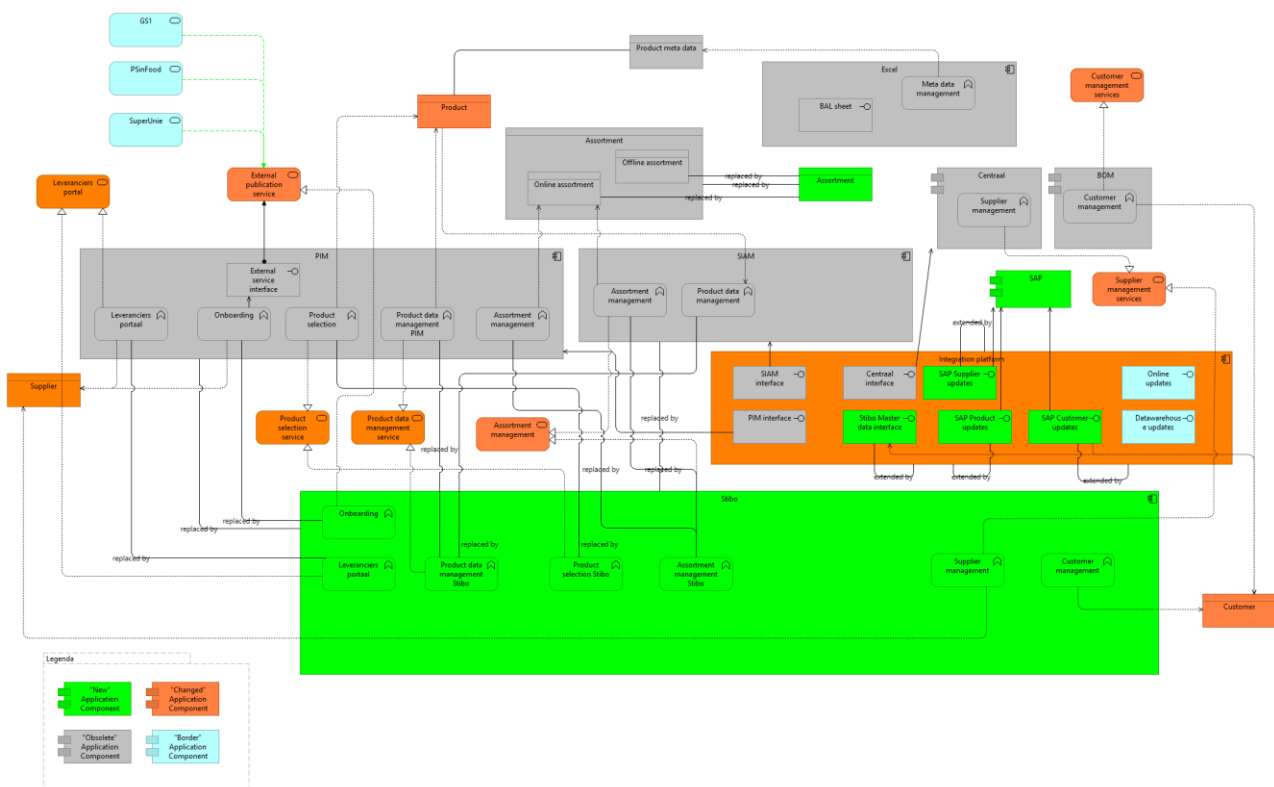


Figure 18. Improved GOC of application layer

This view has been tested during the workshop printed on A3 format. The view comprises obsolete (grey), new (green), changed (orange) and border (original colour) objects as described in the improved GOC definition. The actual changes and the elements comprising the GOC are listed in the appendices.

Some parts of the GOC are explained, the complete list is in the appendix. The Stibo application is a new object (green) while PIM and SIAM are obsolete (grey) application components. The Integration platform is changed (orange) because it remains in the ToBe architecture but services the new systems

SAP and Stibo. Both SIAM and PIM are replaced by Stibo, this is modelled using the ‘replaced by’ relation. Also the ‘Assortment management’ application functions from PIM and SIAM are replaced by application function in Stibo. This makes the ‘Assortment management’ application service changed. The external application services GS1, PSinFood and SuperUnie are border components (original blue application layer colour) because they service the changed ‘external publication service’ because it is now realised by the new Stibo ‘Onboarding’ application function.

The Excel used for Meta data management is an example of an object that is obsolete and not replaced by another object.

The Assortment data object has replaced both the online and offline assortment data object. This is possible because the data object is now created by a single system (Stibo) and not two systems (PIM and SIAM).

Some Obsolete relationships are not always clear because objects can be nested, e.g. the ‘onboarding’ application function of services by the PIM application. The relationships are not visible in this nested construct but they exists. So when a nested object is obsolete also the relationship is obsolete.

A tendency to model only services arises because replacing underlying interfaces and functions becomes crowdie. Using ArchiMate as a language, and not a static dogma, the view will be changed with less objects. Maybe using services using 1 view might be possible. This means that the GOC may have more objects and relationships than show in the views.

5.8 Number of changes

The following changes comprises the GOC:

- obsolete object 25
- new object 13
- unchanged object 8
- changed object 13
- obsolete relation 41
- new relation 26
- replaced by relation 12
- extended by relation 4
- border relation 7

Characteristic for the legacy replacement is the great amount of new and obsolete, both objects and relationships. This change characteristic depends also on the size of the repository related to the change because when the repository contains many unchanged systems these new and obsolete figures would be relatively smaller.

5.9 Improved GOC

As defined before, the actual GOC has been analysed using the software. Table 3 shows a random sub-set of the GOC’ objects, for each element it’s type is indicated using a Pandas DataFrame in the Notebook by Dijkstra (2018). The complete list is in appendix B.

Table 3. First members of the GOC

Seq	Name	Type	New	obsolete	border	changed	unchanged
BO00	Supplier management services	ApplicationService	False	False	True	False	True
CO03	External publication service	ApplicationService	False	False	False	True	False
CO08	BOM interface	ApplicationInterface	False	False	False	True	False
CO12	Customer	DataObject	False	False	False	True	False
NO02	Stibo interface	ApplicationFunction	True	False	False	False	False
OO00	PIM	ApplicationComponent	False	True	False	False	False
OO02	SIAM interface	ApplicationInterface	False	True	False	False	False
OO17	Excel	ApplicationComponent	False	True	False	False	False
OO17	Excel	ApplicationComponent	False	True	False	False	False
OO19	Offline assortment	BusinessProcess	False	True	False	False	False

5.10 Workshop setup

The GOC has been tested using the following workshop design.

Workshop setup

1. Explain the AsIs model and the goals of changes.
2. Next 20 min of the Workshop.
Give the ToBe model and as-is model.
Ask: Would you, please, find out the changes presented in ToBe? Explain the changes using the names of the concepts and relations on the AsIs and ToBe. Use scoring form as shown in appendix C.
3. Last 20 min of the Workshop.
Give the gap model, AsIs and ToBe.
Ask: Would you, please, find out the changes presented by gap? You may use the AsIs and ToBe. Explain the changes using the names of the concepts and relations on the AsIs, ToBe and gap. Use scoring form as shown in appendix C.

Analyse the answers about AsIs and ToBe:

Was a participant able to identify all the changes using AsIs and ToBe?

What kind of changes were not identified?

What kind of changes were identified?

Does the explanation of changes use the in terms of AsIs and ToBe? Does it use other terms and relations?

Is the explanation of changes by participants uniform or different? What are the differences?

Analyse the answers about AsIs, ToBe and gap:

Was the participant able to identify all the changes using AsIs, ToBe and gap?

What kind of changes were not identified?

What kind of changes were identified?

Is the explanation of changes uses the terms of gap?

Is the explanation of changes by participants uniform or different? What are the differences?

Having defined the Gap of Changes the following table lists this set partly. It comprises in total 51 objects and 6 border objects. Together with the relations 140 changes could be found.

The categories of the found differences are:

- Obs obj Obsolete object (25)
- New obj New object (13)
- Chd obj Changed object (13)
- Obs rel Obsolete relation (41)
- New rel New relation (26)
- Bor obj Border object (6)
- Repl Replaced by relation (12)
- Ext Extended by relation (4)

5.11 Workshop results

The workshop was executed with 5 participants and gave the following result. Participant are coded P1 through P5 as show in appendix D.

The listed differences are coded and categorized as related to a specific object or relation of the GOC. The used codes are ‘A’ through ‘AP’, ‘Remark’ and ‘x’. Lines coded as ‘Remark’ and ‘x’ did not contain a usable measurement. An example is the remark “SAP elementen nieuw”; this is not related to a specific element, but to some. ‘x’ indicate reported changes that are no part of the GOC and are not reported by the software.

Table 4 comprises the result coding for the participant using only the AsIs and ToBe view.

Table 4. Scoring classification for AsIs and ToBe

Showing AsIs and ToBe						P1	P2	P3	P4	P5
	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
1	AO	A	G	A	A	OR28	OO00	OR16	OO00	OO00
2	G	B	AF	B	C	OR16	OO04	OR08	OO04	NO00
3	AP	C	AG	C	R	OR28	NO00	OO08	NO00	OO17
4	V	D	V	I	O	RBR09	x	RBR09	x	OO15
5	W	E	W	J	T	RBR08	x	RBR08	OO12	OO10
6	Y	F	X	AA	L	OO05	x	RBR07	NO04	OO21
7	X	G	AH	AB	B	RBR07	OR16		NO12	OO04
8	P	H	P	AC	Z	NO02	OR23	NO02	x	OO02
9	A	I		R	N	OO00	x		OO17	OO24
10	E	J	Q	Remark	U	x	OO12	NO01		OO06
11	Remark	K	AL	AN	I		OO20	OO19	CO13	x
12	U	L	AM	AG	AA	OO06	OO21	OO15	OO08	NO04
13	B		L	AK	Y	OO04		OO21	x	OO05
14			Z	N	K			OO02	OO24	OO20
15					E					x
16					M					x
17					Remark					
18					AD					x
19					AB					NO12
20					Q					NO01
21										

Table 5 comprises the result coding for the participant using only the AsIs, ToBe and GOC view.

Table 5. Scoring classification for AsIs, ToBe and GOC

Showing AsIs and ToBe and GOC										
	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5
1		A		A	A		0000		0000	0000
2		B		B	C		0004		0004	N000
3		C		C	R		N000		N000	0017
4				I	O				x	0015
5				J	T				0012	0010
6		F		AA	L		x		N004	0021
7				AB	B				N012	0004
8				AC	Z				x	0002
9				R	N				0017	0024
10		J		Remark	U		0012			0006
11		K			I		0020			x
12		L			AA		0021			N004
13		S		AK	Y		x		x	0005
14		M		N	K		x		0024	0020
15		N		S	E		0024		x	x
16		O			M		0015			x
17		P			Remark		N002			
18		Q			AD		N001			x
19					AB					N012
20					Q					N001
21					AJ					RBR06

The coding leads to the following totals:

Table 6. Counting results

Showing AsIs and ToBe									
	Obs obj	New obj	Chd obj	Obs rel	New rel	Bor obj	Repl	Ext	Total
P1	6	2	0	1	0	0	3	0	12
P2	7	4	0	0	0	0	0	1	12
P3	6	2	0	1	0	0	3	0	12
P4	7	4	1	0	0	0	0	0	12
P5	11	6	0	0	0	0	0	0	17
Total	37	18	1	2	0	0	6	1	65

Showing AsIs and ToBe and GOC									
	Obs obj	New obj	Chd obj	Obs rel	New rel	Bor obj	Repl	Ext	Total
P1	0	0	0	0	0	0	0	0	0
P2	7	4	0	0	0	0	0	0	11
P3	0	0	0	0	0	0	0	0	0
P4	6	4	0	0	0	0	0	0	10
P5	11	6	0	0	0	0	1	0	18
Total	24	14	0	0	0	0	1	0	39

Two participants did have a significant use of the GOC. One participant reported a changed object as an border object (CO13). No new relations were reported.

Other noticeable results from the workshop are:

- A participant (5) did compare exactly the AsIs and ToBe with the GOC and listed extra changes.
- A participant (3) made the remark that he did not have to functional knowledge to validate the indicated changes. He joined the company shortly and did not know ArchiMate before.
- Different approaches were used for listing changes. Obvious are the application approach (2, 4 and 5) and the relationship approach (1 and 3).
- Arguing the model itself (1) takes time and resulted in less listed differences.
- Participant 1 saw directly the changes in his domain.
- The participants concluded that thinking and talking about objects is easier than thinking and talking about relations.

Only high level changes were mentioned which were a small part of the total elements on the GOC.

6 Discussion and reflection

The GOC has been applied in this research in the case of legacy systems towards ERP/BOB while Bakelaar et al. (2017) did this for ERP towards BOB. The achieved results do come from the analysis of the definition of the GOC and the execution of the practical workshop. The created software acts as a link between the definition and the workshop. Through this link and GOC the architect became more aware of the actual change, other stakeholders can benefit from this increased awareness.

The method is repeatable for also other type of IT changes because the method is business type agnostic. This research can be validated by other researchers because all standards are open and available for free for non-commercial activities to third parties. Also the used software (Archi and Python Notebook) is available for future research. The coding of the results is a spreadsheet for which LibreOffice is available.

The use of the GOC is constrained by the type of EA repository. Only changes that are embedded into an instance oriented architecture is suitable. Because architectures containing architectural meta models and specialisation relationships are not suitable because the GOC requires objects that can be identified for the sake of change management; generic concepts have no identification in the real world.

The Gap of Changes (GOC) included in this case study two domains: MDM and ERP. Only the MDM domain has been subject of the workshop and analysis due to the time constrained workshop setup. The ERP domain is presented in the appendix and in AsIs, ToBe and GOC view, it is not analysed. This split caused a small uncertainty because an extra ‘border’ is created and decisions are required when objects are part of a certain GOC view.

The subject of ArchiMate modelling has been attractive for me from the beginning. However it turned out that using a concept is different from studying it. In this final report I have rewritten large part of the paper and changed the research artefact completely. Looking back this feels ok because I see that the first artefact was of limited scientific value and full of personal interpretations, the second and current feels more scientific based on logical reasoning.

7 Conclusion and Further Research

The analysis and workshop results show that the GOC can be applied in a change from legacy to ERP/BOB. Also the GOC has added value for both the architect and other stakeholders for architectural changes. The architect benefits from the GOC concept when modelling architectural changes because the changes in the model get a 'change' focus instead of a 'state' focus. Other stakeholders experience added value because of the unambiguous changes that are easily found in the GOC. However it turned out to be difficult to list all changes.

This research contributes the EA science domain by the following. The first result of this research is the actual application of the GOC in a legacy replacement case. The second result is an improvement of the definition of the GOC with a subset comprising only change related objects and border objects. This improved GOC is called GOC'. The third result is the software created for analysing the GOC' that tend to be large due to the type of project. Fourth result are the reported application restrictions for relation types and colouring when applying the GOC concept.

The research has been within the context of a migration from legacy to ERP/BOB solution in a FMCG whole sale company. Therefore the result of this thesis is relevant not only for Sligro Food Group but also for any other company that undergoes the transformation and Business Process Redesign (BPR) from legacy systems to a hybrid ERP / BOB within a product trading domain. However the application of the GOC method is not limited to this business and change type and can be applied for any IT change as long as the change is modelled in an repository containing instances of identifiable IT components.

This paper is written as part of a loosely coupled group of 4 peer researches. Analysing these reports together with the original report by Bakelaar et al. (2017) might be subject for new research.

A remaining challenge exists in defining a relation from an object to a relation that is possible since ArchiMate 3. Namely for given objects a, b and c, and relationship (a,b); the relationship from object c to a relationship (a,b) will be (c,(a,b)). This data construct is not part of the current definition of the GOC and is an option for future research because this constraints current EA modelling when using the GOC.

Appendix A Gap of Changes (GOC)

The following list the changes that comprise the GOC. The object sequence relates to the reported changed by the workshop participants in order to be consistent in the results.

Obsolete objects

object_seq	object_type	name
NO00	ApplicationComponent	Stibo
NO01	ApplicationInterface	SAP Customer updates
NO02	ApplicationFunction	Stibo interface
NO03	ApplicationFunction	Product data management Stibo
NO04	ApplicationInterface	SAP Product updates
NO05	ApplicationInterface	Stibo Master data interface
NO06	ApplicationFunction	Leveranciers portaal
NO07	ApplicationFunction	Onboarding
NO08	ApplicationFunction	Product selection Stibo
NO09	DataObject	Assortment
NO10	ApplicationFunction	Customer management
NO11	ApplicationFunction	Assortment management Stibo
NO12	ApplicationInterface	SAP Supplier updates

Obsolete objects

object_seq	object_type	name
OO00	ApplicationComponent	PIM
OO01	ApplicationFunction	Customer management
OO02	ApplicationInterface	SIAM interface
OO03	DataObject	Offline assortment
OO04	ApplicationComponent	SIAM
OO05	DataObject	Online assortment
OO06	ApplicationFunction	Meta data management
OO07	ApplicationFunction	Product data management
OO08	ApplicationInterface	External service interface
OO09	ApplicationFunction	Assortment management
OO10	ApplicationInterface	BAL sheet
OO11	ApplicationFunction	Assortment management
OO12	ApplicationComponent	Centraal
OO13	BusinessProcess	Online assortment management
OO14	DataObject	Assortment
OO15	DataObject	Product meta data

object_seq	object_type	name
OO16	ApplicationFunction	Product data management PIM
OO17	ApplicationComponent	Excel
OO18	ApplicationFunction	Onboarding
OO19	BusinessProcess	Offline assortment
OO20	ApplicationInterface	Centraal interface
OO21	ApplicationInterface	PIM interface
OO22	ApplicationFunction	Leveranciers portaal
OO23	ApplicationFunction	Product selection
OO24	ApplicationComponent	BOM

Changed objects

object_seq	object_type	name
CO00	ApplicationService	Customer management services
CO01	ApplicationFunction	Supplier management
CO02	ApplicationService	Leveranciers portal
CO03	ApplicationService	External publication service
CO04	ApplicationFunction	Supplier management
CO05	DataObject	Product
CO06	ApplicationService	Product data management service
CO07	ApplicationService	Assortment management
CO08	ApplicationInterface	BOM interface
CO09	ApplicationService	Product selection service
CO10	DataObject	Supplier
CO11	ApplicationComponent	Integration platform
CO12	DataObject	Customer

Unchanged objects

Mainly business objects are unchanged because the architecture has been designed to buffer changes on the application services

object_seq	object_type	name
UO00	ApplicationService	Supplier management services
UO01	ApplicationService	Datawarehouse service
UO02	ApplicationInterface	Online updates
UO03	ApplicationService	GS1
UO04	ApplicationComponent	Online 3.0
UO05	ApplicationService	PSinFood
UO06	ApplicationService	SuperUnie

object_seq	object_type	name
UO07	ApplicationInterface	Datawarehouse updates

New relations

New relations are introduced by the new Stibo as MDM solution replacing PIM, SIAM and BOM.

relation_seq	source	source_type	target	target_type	relation_type
NR00	Customer management	ApplicationFunction	Customer	DataObject	AccessRelationship
NR01	Stibo	ApplicationComponent	Stibo Master data interface	ApplicationInterface	CompositionRelationship
NR02	Stibo	ApplicationComponent	Supplier management	ApplicationFunction	AssignmentRelationship
NR03	Stibo	ApplicationComponent	Assortment management Stibo	ApplicationFunction	AssignmentRelationship
NR04	Stibo Master data interface	ApplicationInterface	Customer	DataObject	AccessRelationship
NR05	Integration platform	ApplicationComponent	Stibo interface	ApplicationFunction	AssignmentRelationship
NR06	SAP Product updates	ApplicationInterface	Product	DataObject	AccessRelationship
NR07	Integration platform	ApplicationComponent	SAP Customer updates	ApplicationInterface	CompositionRelationship
NR08	Product selection Stibo	ApplicationFunction	Product selection service	ApplicationService	RealizationRelationship
NR09	Stibo Master data interface	ApplicationInterface	Product	DataObject	AccessRelationship
NR10	Leveranciers portaal	ApplicationFunction	Leveranciers portal	ApplicationService	RealizationRelationship
NR11	Assortment management Stibo	ApplicationFunction	Assortment management	ApplicationService	RealizationRelationship
NR12	SAP Customer updates	ApplicationInterface	Customer	DataObject	AccessRelationship
NR13	Stibo	ApplicationComponent	Product selection Stibo	ApplicationFunction	AssignmentRelationship
NR14	SAP Supplier updates	ApplicationInterface	Supplier	DataObject	AccessRelationship
NR15	Integration platform	ApplicationComponent	SAP Supplier updates	ApplicationInterface	CompositionRelationship
NR16	Integration platform	ApplicationComponent	SAP Product updates	ApplicationInterface	CompositionRelationship
NR17	Assortment management Stibo	ApplicationFunction	Assortment	DataObject	AccessRelationship
NR18	Stibo Master data interface	ApplicationInterface	Supplier	DataObject	AccessRelationship
NR19	Product data management Stibo	ApplicationFunction	Product data management service	ApplicationService	RealizationRelationship
NR20	Stibo	ApplicationComponent	Leveranciers portaal	ApplicationFunction	AssignmentRelationship
NR21	Onboarding	ApplicationFunction	External publication service	ApplicationService	RealizationRelationship

relation_seq	source	source_type	target	target_type	relation_type
NR22	Stibo Master data interface	ApplicationInterface	Integration platform	ApplicationComponent	ServingRelationship
NR23	Stibo	ApplicationComponent	Customer management	ApplicationFunction	AssignmentRelationship
NR24	Stibo	ApplicationComponent	Onboarding	ApplicationFunction	AssignmentRelationship
NR25	Stibo	ApplicationComponent	Product data management Stibo	ApplicationFunction	AssignmentRelationship

Obsolete relations

Obsolete relations are mainly caused by the removal of the legacy system.

relation_seq	source	source_type	target	target_type	relation_type
OR00	BOM	ApplicationComponent	Customer management	ApplicationFunction	AssignmentRelationship
OR01	Customer management	ApplicationFunction	Customer	DataObject	AccessRelationship
OR02	BOM interface	ApplicationInterface	BOM	ApplicationComponent	ServingRelationship
OR03	Centraal	ApplicationComponent	Supplier management	ApplicationFunction	AssignmentRelationship
OR04	Integration platform	ApplicationComponent	PIM interface	ApplicationInterface	CompositionRelationship
OR05	Integration platform	ApplicationComponent	Centraal interface	ApplicationInterface	CompositionRelationship
OR06	PIM	ApplicationComponent	Leveranciers portaal	ApplicationFunction	AssignmentRelationship
OR07	SIAM	ApplicationComponent	PIM	ApplicationComponent	AssociationRelationship
OR08	Onboarding	ApplicationFunction	Supplier	DataObject	AccessRelationship
OR09	PIM	ApplicationComponent	Product data management PIM	ApplicationFunction	AssignmentRelationship
OR10	Product data management PIM	ApplicationFunction	Product data management service	ApplicationService	RealizationRelationship
OR11	Centraal interface	ApplicationInterface	Centraal	ApplicationComponent	ServingRelationship
OR12	Assortment management	ApplicationFunction	Online assortment	DataObject	AccessRelationship
OR13	External service interface	ApplicationInterface	External publication service	ApplicationService	AssignmentRelationship
OR14	Assortment	DataObject	Online assortment	DataObject	CompositionRelationship
OR15	Assortment management	ApplicationFunction	Online assortment	DataObject	AccessRelationship
OR16	Leveranciers portaal	ApplicationFunction	Supplier	DataObject	AccessRelationship
OR17	PIM	ApplicationComponent	Product selection	ApplicationFunction	AssignmentRelationship
OR18	SIAM interface	ApplicationInterface	SIAM	ApplicationComponent	ServingRelationship
OR19	Meta data management	ApplicationFunction	Product meta data	DataObject	AccessRelationship
OR20	PIM	ApplicationComponent	External service interface	ApplicationInterface	CompositionRelationship

relation_seq	source	source_type	target	target_type	relation_type
OR21	PIM	ApplicationComponent	Assortment management	ApplicationFunction	AssignmentRelationship
OR22	Product data management	ApplicationFunction	Product	DataObject	AccessRelationship
OR23	Product selection	ApplicationFunction	Product	DataObject	AccessRelationship
OR24	Product data management PIM	ApplicationFunction	Product	DataObject	AccessRelationship
OR25	Assortment	DataObject	Offline assortment	DataObject	CompositionRelationship
OR26	Assortment management	ApplicationFunction	Assortment management	ApplicationService	RealizationRelationship
OR27	Assortment management	ApplicationFunction	Offline assortment	DataObject	AccessRelationship
OR28	Onboarding	ApplicationFunction	External service interface	ApplicationInterface	ServingRelationship
OR29	BOM	ApplicationComponent	Centraal	ApplicationComponent	AssociationRelationship
OR30	Excel	ApplicationComponent	BAL sheet	ApplicationInterface	CompositionRelationship
OR31	Integration platform	ApplicationComponent	SIAM interface	ApplicationInterface	CompositionRelationship
OR32	Excel	ApplicationComponent	Meta data management	ApplicationFunction	AssignmentRelationship
OR33	Product selection	ApplicationFunction	Product selection service	ApplicationService	RealizationRelationship
OR34	Leveranciers portaal	ApplicationFunction	Leveranciers portal	ApplicationService	RealizationRelationship
OR35	SIAM	ApplicationComponent	Assortment management	ApplicationFunction	AssignmentRelationship
OR36	Product meta data	DataObject	Product	DataObject	AssociationRelationship
OR37	Customer management	ApplicationFunction	Customer management services	ApplicationService	RealizationRelationship
OR38	PIM interface	ApplicationInterface	PIM	ApplicationComponent	ServingRelationship
OR39	PIM	ApplicationComponent	Onboarding	ApplicationFunction	AssignmentRelationship
OR40	SIAM	ApplicationComponent	Product data management	ApplicationFunction	AssignmentRelationship

Extended by relations

The extended by relations are the newly introduced relationships.

relation_seq	source	source_type	target	target_type
EBR00	Integration platform	ApplicationComponent	SAP Supplier updates	ApplicationInterface
EBR01	Integration platform	ApplicationComponent	SAP Product updates	ApplicationInterface
EBR02	Integration platform	ApplicationComponent	SAP Customer updates	ApplicationInterface
EBR03	Integration platform	ApplicationComponent	Stibo Master data interface	ApplicationInterface

Replaced by relations

The replaced by relations are the newly introduced relationships.

relation_seq	source	source_type	target	target_type
RBR00	Onboarding	ApplicationFunction	Onboarding	ApplicationFunction
RBR01	Product data management PIM	ApplicationFunction	Product data management Stibo	ApplicationFunction
RBR02	Leveranciers portaal	ApplicationFunction	Leveranciers portaal	ApplicationFunction
RBR03	Assortment	DataObject	Assortment	DataObject
RBR04	SIAM	ApplicationComponent	Stibo	ApplicationComponent
RBR05	Offline assortment	DataObject	Assortment	DataObject
RBR06	Online assortment	DataObject	Assortment	DataObject
RBR07	Assortment management	ApplicationFunction	Assortment management Stibo	ApplicationFunction
RBR08	Product data management	ApplicationFunction	Product data management Stibo	ApplicationFunction
RBR09	Product selection	ApplicationFunction	Product selection Stibo	ApplicationFunction
RBR10	Assortment management	ApplicationFunction	Assortment management Stibo	ApplicationFunction
RBR11	PIM	ApplicationComponent	Stibo	ApplicationComponent

Border relations

Border relations give context to the change. Because the complete system replacements the border that is left is rather small.

relation_seq	source	source_type	target	target_type	relation_type
BR00	SuperUnie	ApplicationService	External publication service	ApplicationService	FlowRelationship
BR01	Integration platform	ApplicationComponent	Online updates	ApplicationInterface	CompositionRelationship
BR02	Supplier management	ApplicationFunction	Supplier management services	ApplicationService	RealizationRelationship
BR03	Integration platform	ApplicationComponent	Datawarehouse updates	ApplicationInterface	CompositionRelationship
BR04	GS1	ApplicationService	External publication service	ApplicationService	FlowRelationship
BR05	PSinFood	ApplicationService	External publication service	ApplicationService	FlowRelationship
BR06	Supplier management	ApplicationFunction	Supplier management services	ApplicationService	RealizationRelationship

Appendix B Improved GOC

The following listing comprise the objects that are part of the GOC as defined in chapter 3. “True” and “False” values indicate whether the object is part of that change category conform the definition of the improved GOC.

Seq	Name	Type	New	obsolete	border	changed	unchanged
BO00	Supplier management services	ApplicationService	False	False	True	False	True
BO01	Online updates	ApplicationInterface	False	False	True	False	True
BO02	PSinFood	ApplicationService	False	False	True	False	True
BO03	GS1	ApplicationService	False	False	True	False	True
BO04	SuperUnie	ApplicationService	False	False	True	False	True
BO05	Datawarehouse updates	ApplicationInterface	False	False	True	False	True
CO00	Customer management services	ApplicationService	False	False	False	True	False
CO01	Supplier management	ApplicationFunction	False	False	False	True	False
CO02	Leveranciers portal	ApplicationService	False	False	False	True	False
CO03	External publication service	ApplicationService	False	False	False	True	False
CO04	Supplier management	ApplicationFunction	False	False	False	True	False
CO05	Product	DataObject	False	False	False	True	False
CO06	Product data management service	ApplicationService	False	False	False	True	False
CO07	Assortment management	ApplicationService	False	False	False	True	False
CO08	BOM interface	ApplicationInterface	False	False	False	True	False
CO09	Product selection service	ApplicationService	False	False	False	True	False
CO10	Supplier	DataObject	False	False	False	True	False
CO11	Integration platform	ApplicationComponent	False	False	False	True	False
CO12	Customer	DataObject	False	False	False	True	False
NO00	Stibo	ApplicationComponent	True	False	False	False	False
NO01	SAP Customer updates	ApplicationInterface	True	False	False	False	False
NO02	Stibo interface	ApplicationFunction	True	False	False	False	False
NO03	Product data management Stibo	ApplicationFunction	True	False	False	False	False
NO04	SAP Product updates	ApplicationInterface	True	False	False	False	False
NO05	Stibo Master data interface	ApplicationInterface	True	False	False	False	False
NO06	Leveranciers portaal	ApplicationFunction	True	False	False	False	False
NO07	Onboarding	ApplicationFunction	True	False	False	False	False
NO08	Product selection Stibo	ApplicationFunction	True	False	False	False	False
NO09	Assortment	DataObject	True	False	False	False	False
NO10	Customer management	ApplicationFunction	True	False	False	False	False
NO11	Assortment management Stibo	ApplicationFunction	True	False	False	False	False
NO12	SAP Supplier updates	ApplicationInterface	True	False	False	False	False
OO00	PIM	ApplicationComponent	False	True	False	False	False
OO01	Customer management	ApplicationFunction	False	True	False	False	False
OO02	SIAM interface	ApplicationInterface	False	True	False	False	False
OO03	Offline assortment	DataObject	False	True	False	False	False
OO04	SIAM	ApplicationComponent	False	True	False	False	False
OO05	Online assortment	DataObject	False	True	False	False	False

Seq	Name	Type	New	obsolete	border	changed	unchanged
OO06	Meta data management	ApplicationFunction	False	True	False	False	False
OO07	Product data management	ApplicationFunction	False	True	False	False	False
OO08	External service interface	ApplicationInterface	False	True	False	False	False
OO09	Assortment management	ApplicationFunction	False	True	False	False	False
OO10	BAL sheet	ApplicationInterface	False	True	False	False	False
OO11	Assortment management	ApplicationFunction	False	True	False	False	False
OO12	Centraal	ApplicationComponent	False	True	False	False	False
OO13	Online assortment management	BusinessProcess	False	True	False	False	False
OO14	Assortment	DataObject	False	True	False	False	False
OO15	Product meta data	DataObject	False	True	False	False	False
OO16	Product data management PIM	ApplicationFunction	False	True	False	False	False
OO17	Excel	ApplicationComponent	False	True	False	False	False
OO18	Onboarding	ApplicationFunction	False	True	False	False	False
OO19	Offline assortment	BusinessProcess	False	True	False	False	False
OO20	Centraal interface	ApplicationInterface	False	True	False	False	False
OO21	PIM interface	ApplicationInterface	False	True	False	False	False
OO22	Leveranciers portaal	ApplicationFunction	False	True	False	False	False
OO23	Product selection	ApplicationFunction	False	True	False	False	False
OO24	BOM	ApplicationComponent	False	True	False	False	False

Appendix C. Scoring form

Scoreformulier 1 - Views AsIs en ToBe (Applicatie MDM)

Vraag 1	#	Verandering
<p>Welke veranderingen zie je tussen de views (AsIs en ToBe)?</p> <p>Geef in de kolom hiernaast voor elke verandering per regel aan wat de verandering volgens jou inhoudt.</p> <p>Geef aan welke elementen en relaties er nieuw, verwijderd, vervangen of uitgebreid zijn?</p>	1	
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	10	
	11	
	12	
	13	
	14	
	15	
	16	
	17	
	18	
	19	

Naam :

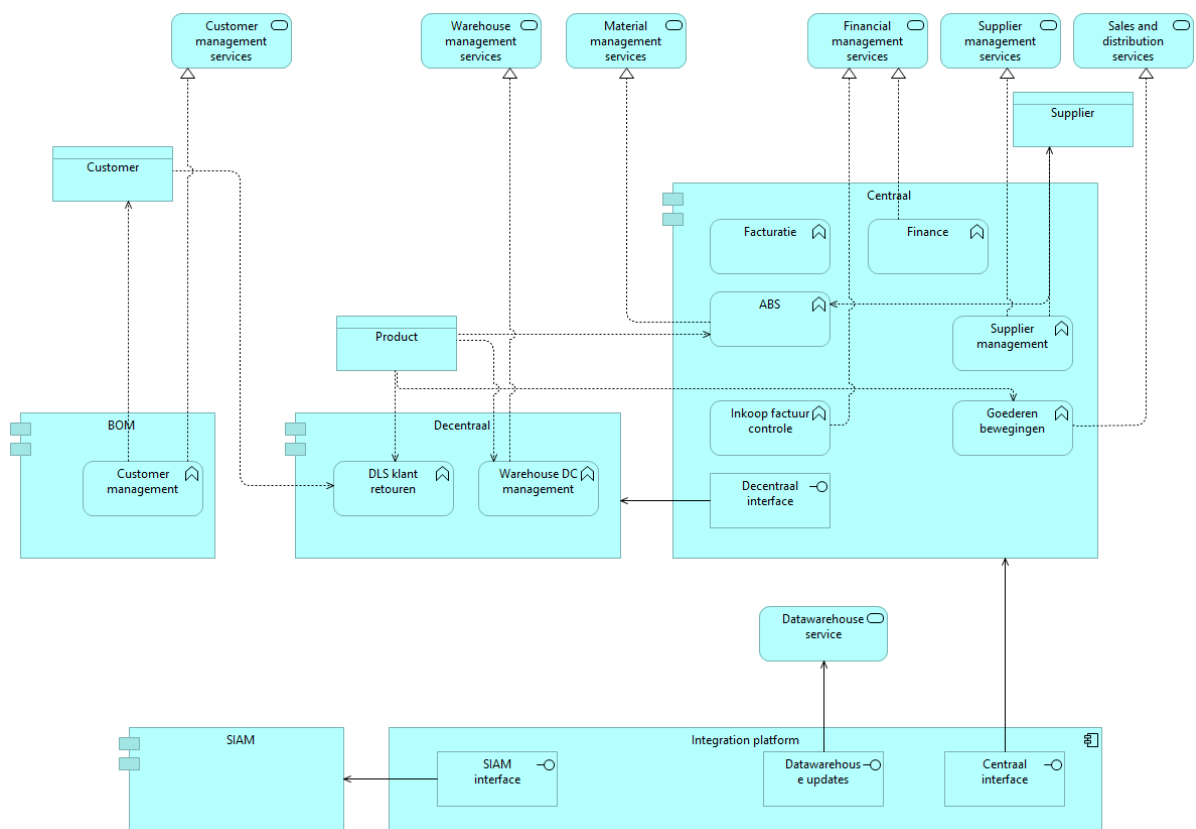
Appendix D. Participant result coding

Code	Type	Omschrijving	Seq	In AsIs/ ToBe	In AsIs/ ToBe/ GOC
A	Obs obj	PIM verwijderd	OO00	4	3
B	Obs obj	SIAM verwijderd	OO04	4	3
C	New obj	Stibo nieuw	NO00	3	3
D	Ext	Integration platform is uitgebreid met "Datawarehouse update"	x	1	0
E	New obj	"Data warehouse service" is nieuw	x	3	1
F	New obj	"Customer object" is nieuw	x	1	1
G	Obs obj	"Leveranciers Portaal functie" heeft geen relatie met "supplier object"	OR16	3	0
H	Obs obj	"Product selection function" heeft geen meer relatie met product object	OR23	1	0
I	New obj	SAP is nieuw systeem	x	3	2
J	Obs obj	CENTRAAL systeem is verwijderd	OO12	2	2
K	Obs obj	"Centraal interface" is uit integration platform	OO20	2	2
L	Obs obj	"PIM interface" is uit integration platform	OO21	3	2
M	Remark	Data warehouse staat niet als nieuw aangegeven	x	1	2
N	obs obj	BOM verdwenen	OO24	2	3
O	Obs obj	"Product meta data" is verdwenen	OO15	1	2
P	New obj	Stibo masterdata interface is toegevoegd	NO02	2	1
Q	New obj	SAP customer updates toegevoegd	NO01	2	2
R	Obs obj	Excel verwijderd	OO17	2	2
S	Remark	SAP is vervallen	x	0	2
T	Obs obj	BAL sheet verwijderd	OO10	1	1
U	Obs obj	Meta data management mist	OO06	2	1
V	Repl	Product selection vervangen door product selecton in Stibo	RBR09	2	0
W	Repl	Vervangen product data management PIM vervangen door PDM in Stibo	RBR08	2	0
X	Repl	Assortment management vervangen door Ass mgt in Stibo	RBR07	2	0
Y	Obs obj	Online assortment verwijderd	OO05	2	1
Z	Obs obj	SIAM interface verwijderd	OO02	2	1
AA	New obj	SAP Product updates nieuw	NO04	2	2
AB	New obj	SAP Supplier updates nieuw	NO12	2	2
AC	x	Supplier management services verwijderd	x	1	1
AD	x	Customer man services verwijderd	x	1	1
AE	x	Verwijderd relatie lev portaal - supplier (equals G)	x	0	0
AF	Obs rel	Verwijderd relatie onboarding - supplier	OR08	1	0
AG	Obs obj	Verwijderd extrenal services interface	OO08	2	0
AJ	Repl	Online assortiment vervangen door assortiment	RBR06	0	1
AK	Obs obj	Supplier verwijderd	x	1	1
AL	Obs obj	Offline assortiment verwijderd	OO19	1	0
AM	Obs obj	Product meta data vervallen	OO15	1	0
AN	Chd obj	External publication service	CO13	1	0

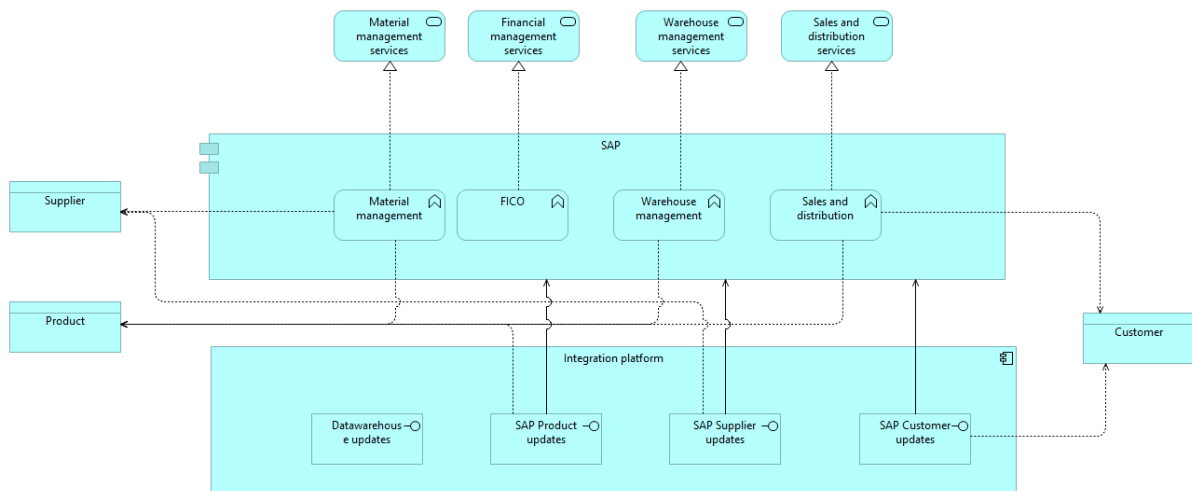
AO	Obs obj	Rechtstreekse relatie "Onboarding" en external publ serv.	OR28	1	0
AP	Obs rel	Onboarding 1 output minder	OR28	1	0

Appendix E. Financial and supply chain system replacement

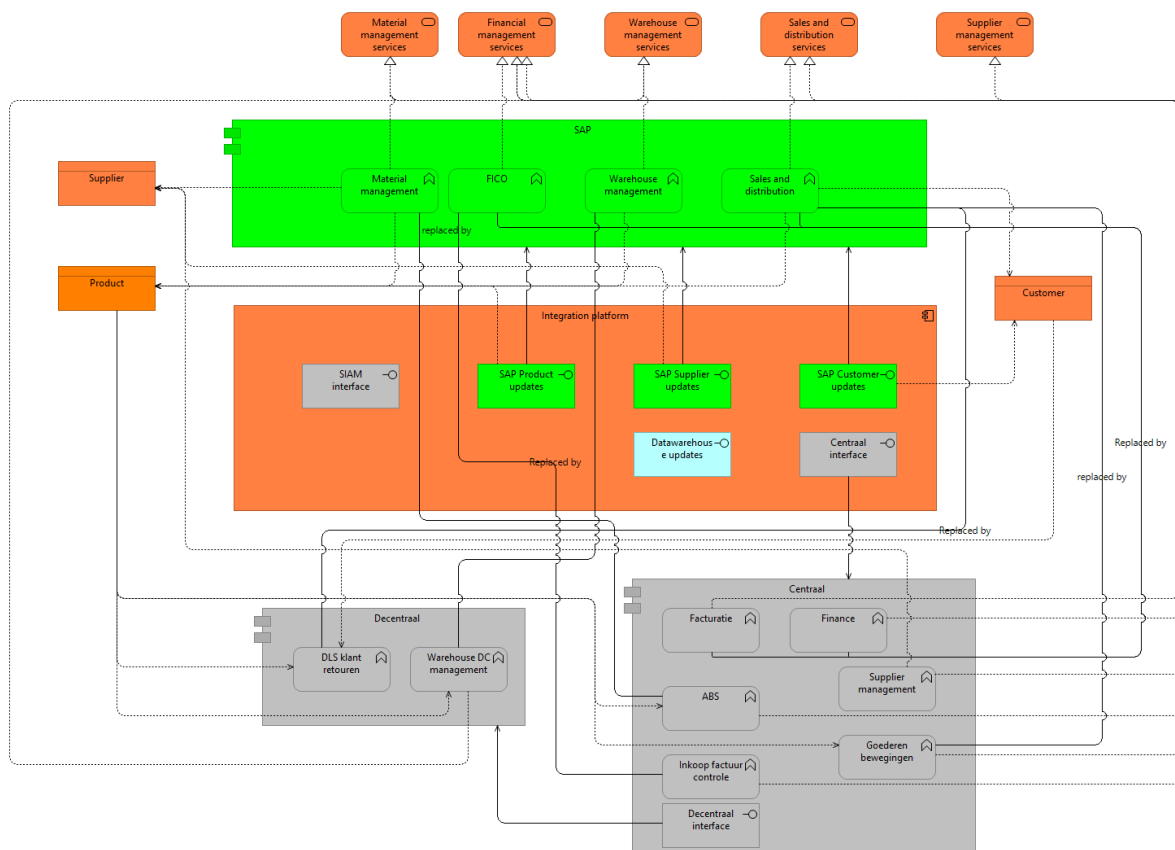
AsIs, financial and supply chain system



ToBe, financial and supply chain system



GOC, legacy financial and supply chain system



List of figures

Figure 1. Full ArchiMate framework from ArchiMate (2017)	9
Figure 2. Business layer fragment from case study	9
Figure 3. Typical ArchiMate fragment	11
Figure 4. Excel sheet model.....	12
Figure 5. Nesting ArchiMate elements	13
Figure 6. Grouping as diagramming technique.....	13
Figure 7. Derived relations	14
Figure 8. Distinction between standard and modified software.....	14
Figure 9. GOC relations.....	16
Figure 10. Colour and nesting issue.....	18
Figure 11. Full system replacement	20
Figure 12. Used modelling convention	21
Figure 13. Motivation for implementing ERP and MDM.....	22
Figure 14. AsIs business processes for Master Data Management (MDM) and primary process	23
Figure 15. ToBe business processes for master data management and primary process.....	24
Figure 16. Application MDM AsIs view	25
Figure 17. Application MDM ToBe view.....	25
Figure 18. Improved GOC of application layer	26

List of tables

Table 1. ArchiMate elements explained	10
Table 2. ArchiMate relationships explained	10
Table 3. First members of the GOC.....	28
Table 4. Scoring classification for AsIs and ToBe	29
Table 5. Scoring classification for AsIs, ToBe and GOC	30
Table 6. Counting results	30

References

Aldea A., Iacob M., Hillegersberg J. van, Quartel D., Bodenstaff L., Franken H. (2015). Modelling strategy with ArchiMate. Conference: 30th Annual ACM Symposium on Applied Computing (SAC) At: Salamanca DOI: 10.1145/2695664.2699489

ArchiMate. (2017). *ArchiMate® 3.0.1 Specification*. Retrieved from <http://pubs.opengroup.org/architecture/archimate3-doc/toc.html>

ArchiMate, chapter 5. (2017). *ArchiMate® 3.0.1 Specification, 5 Relationships*. Retrieved from <http://pubs.opengroup.org/architecture/archimate3-doc/chap05.html>

Archi. (2018). Retrieved from <https://www.archimatetool.com/>

Azevedo C., Iacob M., Almeida J., Sinderen M. van, Pires L., Guizzardi G. (2015). Modelling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate.

- Baghi E., Schlosser S., Ebner V., Otto B., Oesterle H. (2014). Toward a Decision Model for Master Data Application Architecture 47th Hawaii International Conference on System Sciences.
- Bakelaar, B., Roubtsova E., Joosten, S. (2017). A Framework for Visualization of Changes, Springer International.
- Buschmann F. (2009). Introducing the Pragmatic Architect. Siemens corporate Technology, IEEE Software.
- Dijkstra, R. (2018). <https://github.com/RichDijk/EAGOC>.
- Foorthuis, R., van Steenberg, M., Brinkkemper, S. et al. (2016). A theory building study of enterprise architecture practices and benefits. *Inf Syst Front* 18: 541. <https://doi.org.ezproxy.elib11.ub.unimaas.nl/10.1007/s10796-014-9542-1>
- Haag S, Eckhardt A. (2017). Shadow IT. *Bus Inf Syst Eng* 59(6):469–473 (2017) <https://doi.org/10.1007/s12599-017-0497-x>
- Jonkers, H., Lankhorst, M., ter Doest, H., Arbab, F., Bosma, H., Wieringa, R. (2006). Enterprise architecture: management tool and blueprint for the organisation. *Information Systems Frontiers* 8, 63–66.
- Jupyter Notebook. (2018). Retrieved from <http://jupyter.org/>
- Lankhorst, M. (2009). Enterprise Architecture at Work. ISBN 978-3-642-01309-6 e-ISBN 978-3-642-01310-2 DOI 10.1007/978-3-642-01310-2
- Light B., Holland C., Christopher P., Wills K. (2001). ERP and best of breed : a comparative analysis. *Business Process Management Journal*, 7(3), pp. 216-224.
- Maier N., Hoffman, R. (1967). “VALENCE IN THE ADOPTION OF SOLUTIONS BY PROBLEM-SOLVING GROUPS”. *Journal of Personality and Social Psychology* 1967, Vol. 6, No. 2, 175-182
- Makey D., Zundel M. (2017). “Recovering the Divide: A Review of Strategy and Tactics in Business and Management”. *International Journal of Management Reviews*, Vol. 19, 175–194 (2017) DOI: 10.1111/ijmr.12091
- Moody D. (2009). The “Physics” of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering, Daniel L. Moody, Member, IEEE, *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, VOL. 35, NO. 6, NOVEMBER/DECEMBER 2009
- Pandas DataFrame (2018). Retrieved from <https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html>
- Python. (2018). Retrieved from <https://www.python.org/>
- Schwaninger M. (2010). Model-based management (MBM): a vital prerequisite for organizational viability. University of St Gallen, St Gallen, Switzerland. *Kybernetes* Vol. 39 No. 9/10, 2010 pp. 1419-1428 Emerald Group Publishing Limited 0368-492X DOI 10.1108/03684921011081105
- TOGAF. (2017). <http://pubs.opengroup.org/architecture/togaf9-doc/arch/>
- Wierda G. (2015). Chess and the art of Enterprise Architecture, R&A, The Netherlands.