Universiteit Utrecht

Utilizing change effort prediction to analyse modifiability of business rules architectures

MBI Graduation project – Thesis

Koen Smit

Version 1.4.1

"Adapt or perish, now as ever, is nature's inexorable imperative"

- H.G. Wells

MBI Graduation project – Thesis

Utilizing change effort prediction to analyse modifiability of business rules architectures

November 2014 / July 2015

Author:

Koen Smit

Student#: 4084063

Evenaar 105

3454 SK – De Meern

06-19 00 18 02

Utrecht University

Graduate School of Natural Sciences

Master of Business Informatics

Year 2, Semester 3/4

First supervisor:

Dr. R.S.Batenburg

R.S.Batenburg@uu.nl

Utrecht University

Second supervisor:

Prof. Dr. Sjaak Brinkkemper

S.Brinkkemper@uu.nl

Utrecht University

External supervisor:

Dr. Ing. Martijn Zoet

Martijn.zoet@zuyd.nl

Zuyd University of Applied Sciences

Utrecht, 9 July 2015

Universiteit Utrecht

Abstract

Currently, many organizations employ business logic, and therefore business rules, as part of their products and/or services to deliver added value to their customers. However, as the amount of business rules utilized increases, the cohesion between those business rules is not taken into account. This makes it practically impossible to perform proper impact analysis before changes need to be implemented. Furthermore, the configuration of the business logic in products and/or services containing business rules are generally not designed to cope with modifiability. This seems very counterintuitive as previous research already proved that business logic is characterized by the highest change frequency in combination with the highest amount of effort required to perform the changes when compared to other aspects of IS development and maintenance.

First, a literature review is conducted to provide an overview of the context of business logic, business rules, the curent state-of-the-art on business rule architectures, neighbouring fields, and modifiability as a measurable quality attribute of a architecture. The literature review was followed by the exploration of three rule-oriented architectures for application in the BRM domain. The three rule-oriented architectures are utilized to construct logical architectures for the cases Chronic Obstructive Pulmonary Disease (COPD) and Diabetes Mellitus (DM) of the NHS, which concern the remuneration of healtcare services. Both the exploration and constuction of logical architectures regarding the three selected architectural candidates is performed according to the design science research approach, including a multitude of validation cycles with subject-matter experts. This was followed by selecting an architecture evaluation method from the body of knowledge to evaluate the architectural candidates on effort required to apply modifications, and therefore a means to analyse modifiability.

Based on four criteria the ALMA-method from the software architecture domain was selected. As ALMA is dependent on productivity measures of individual cases, we conducted two interviews with NHS personnel who transform requirements from multiple stakeholders into business rules. The interviews resulted in two levels of experience, which we interpreted carefully as worst and best case productivity figures for the NHS as input for ALMA. Lastly, we gathered additional historic data regarding change history of the NHS business rule sets as applied in the clinical conditions COPD and DM concerning the previous eight years.

ALMA's results reveal that from all three included architectural candidates, the rule family-oriented architecture performs best on effort prediction. Analysis of the results shows that the case, the clinical condition COPD, containing less business rule sets, shows relatively equal modifiability performance concerning the included architectural candidates. This changes considerably when a larger case is analysed, which increases difference in modifiability performance between the included architectural candidates. We believe this difference is caused by the strutctural design decision of an architectural candidate which enforces the logical architectures either to utilize redundant elements or reuse existing elements.

Keywords: Business logic ,Business Rules Management, Business Rule Architectures, Modifiability, Architecture-Level Modifiability Analysis.

Acknowledgements

This thesis is the final part of my master program, Business Informatics at Utrecht University. In November 2014 I started my graduation period which was carried out at the HU University of Applied Sciences Utrecht (HU). This research is triggered by earlier research on the topic of business rules management that I performed for the chair Extended Enterprise Studies in the past two years.

I would like to express my gratitude to some special people who really supported me during the master program and graduation project. First, special gratitude goes out to Dr. Ronald Batenburg and Prof. Sjaak Brinkkemper for guiding me through the graduation process, but also providing me with feedback on how to perform 'proper' research, obtain my MSc, and deliver this thesis. Furthermore, I would like to thank Dr. Martijn Zoet for being my mentor in challenging times, but also as a colleague who is always there when I walk around with literally one hundred questions each day to fire at him. Also, I would like to thank Adri Kohler MSc as my father-figure at the office, who always provided me a sympathetic listening ear during this research project. Further, I would like to mention my thanks to Prof. Johan Versendaal who has been one of my role models in the past two-and-a-half years at my time working at the HU. Last but not least, I would like to thank Sam Leewis for supporting me with the practical aspects during my research project, also including his fresh view on the subject matter and accompanying challenges that I faced.

Furthermore, I would like to thank the NHS for their hospitality and effort taken to participate in this research project. Without their commitment it was surely impossible to deliver the research results in its current level of completeness and form.

Lastly, I would like to express my general appreciation to everyone which I have spoken to or who provided me with valuable hints and tips, either regarding the subject matter or graduating for a Masters degree in general. Furthermore, I would also like to thank my loving parents, family, friends, and my girlfriend for the much needed support, social interaction and distraction while this research project lasted.

Utrecht, July 9, 2015

Koen Smit

Table of Contents

List of figures

List of tables

List of equations

1 Introduction

1.1 Research triggers

In this section several triggers are identified which sparked the initiative to start this research project.

Scientific triggers

In the field of information systems the domain "Business Rules" or "BR" is a relatively young subject of study and steadily gained interest from researchers in the decade (Boyer & Mili, 2011; Liao, 2004; Morgan, 2002; Zoet, 2014). However, multiple knowledge gaps in this research domain are yet to be researched (Zoet, 2014), for example, how large sets of BR's are designed in the form of architectures.

The topic of architecture(s) is a heavily studied field in the IS domain. Different domains have their own architectures, for example; software architectures, database (data) architectures, data warehouse architectures, and business process management architectures. These fields are studied from multiple perspectives, for example: the business, application, and technology perspectives. If we look at the business process management domain, from a business perspective, a business process architecture is defined. Examples are: ETOM (TM forum, 2014), NICTIZ (NICTIZ, 2014) and NORA (E-overheid, 2014). On the application layer, BPMS architectures are defined. Shaw et al. (2007) and Lankhorst et al. (2009) state that each layer can be viewed from different aspects, for example: the passive structure, behavioural structure, and the active structure. The passive structure contains the architecture for the business objects itself while the behavioural structure contains the software application elements that modify these same elements. We know business rule architectures are utilized in practice, but few to none have written scientific literature on this subject (Nelson, Peterson, Rariden, & Sen, 2010)

In this research we focus on the field of business rules management and more specifically on business rules architectures from the perspective of formal model construct architecture and application architecture, as represented by the green squares in Figure 1. Formal model constructs are prefabricated construction elements from which a business rule architecture is build (Shaw et al., 2007). They are independent of means, techniques and the process of modelling. Instantiations of formal model constructs are business rule model in a specific language. For example, a business rule model specified in Semantics of Business Vocabulary and Business Rules, Decision Tables or Decision Trees. However, the design of an application influences the way in which instantiations of the formal model are dealt with. The design of a software application is depicted in its application architecture.

An application architecture is the composition of a set of architectural design decisions over a set of components and connectors (Bosch, 2004). In general, designing a software architecture can be viewed as a decision process. Each decision influences the formal structure of the software. The same applies to a software architecture of a business rules management system (BRMS). For example, Liao (2004) identifies over 20 different instantiations BRMS-software architectures. Each architecture has a different view on how to design, model and execute the formal construct of which a business rule model consist of. How these different design choices and the resulting architecture influences the modifiability of these architectures has not been investigated. This research will focus on identifying and analysing both formal constructs and application architecture constructs.

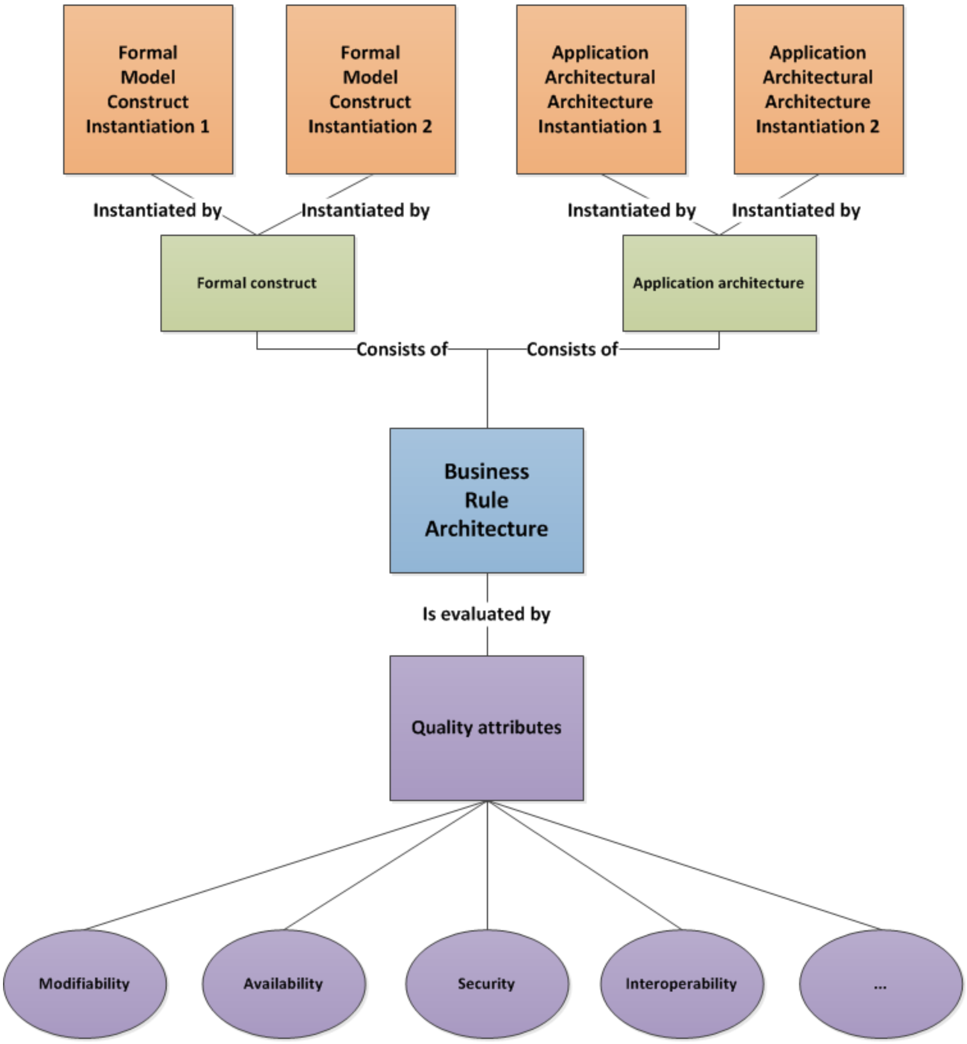


Figure 1 - Schematic overview of the problem space

In addition to defining the business rules architecture from the perspective of formal model construct and applications it is necessary to research the quality the business rule architectures, as represented by the purple objects in Figure 1. This is due to the need to be able to assess the appropriateness of different architectures. Quality measurements are widely applied in the IS domain, specifically for software architectures. Software architectures can be rated through utilizing quality attributes that define different perspectives, possibly from different stakeholders, to measure the architectural quality. Some examples of quality attributes concerning software architectures can be found in the work of Bass, Clements, and Kazman (2012). They mention several quality attributes ranging from availability, interoperability, modifiability, performance, security, testability and usability. Within the field of system lifecycle management, Ross, Rhodes, and Hastings (2008) also define quality attributes. Relevant examples are: changeability, flexibility, adaptability, scalability, modifiability, and robustness. In this research we focus on modifiability.

Optimal modifiability is defined as: "*The ability to incorporate anomalies to the business rule architecture made possible by the minimal number of changes.*" (Len Bass et al., 2012). Literature reveals that the knowledge on topics like software and database architectures is enriched by numerous studies that can be utilized as a starting point for this direction of research.

Triggers from practice

Nowadays, business rules affect large parts of the society in western countries due to the increasing demand for automation of enforcement of laws and regulations, which aims for effectiveness and efficiency at large governmental bodies. In practice, the automation of business rules implies that laws and regulations (naturally written in judicial language) need to be transformed to products and services which take into account those laws and regulations. A practical example of such a service in the Dutch context is the Dienst Uitvoering Onderwijs (hereafter: DUO). The DUO is responsible for students receiving monthly compensation when they are studying. Based on a business rule architecture containing eight thousand rules, the student can login and utilize the web portal to determine the height of the compensation. The height of the study compensation in euros is calculated using the student's variables like student income, parent(s) income, number of sisters/brothers also receiving study compensation, and the type of education in which the student is enrolled: MBO or higher education (Dienst Uitvoering Onderwijs, 2014). Practically speaking, it is imperative that laws and regulations which are utilized by automated services are valid on the moment they are used in decision making for consumers of the service. In practice, this is called compliance. Compliance is defined as "Acting in accordance with established laws, regulations, standards, and specifications" (Tarantino, 2006, 2008). In practice, this proven to be harder than it seems due to transformation of business language (the earlier mentioned judicial language) into IT services (technical language), which always have been a problem area in IS development (Darke & Shanks, 1996; Kim, Lee, & Gosain, 2005). This research does not address the transformation aspect itself, but the improvements regarding efficiency of implementing BR in modifiable BR architectures.

Another observation is the trade-off that is made in practice between efficiency & effectiveness vs. compliancy. It seems that businesses unintentionally, but often intentionally take risks when it comes down to this trade-off. Adhering to laws, regulations, protocols, standards, and specifications can pose considerable financial risks for businesses. Examples of this are: Sarbanes-Oxley and HIPPA (Tarantino, 2006).

In practice, laws and regulations are constantly subjected to changes due to certain laws or parts of it getting obsolete (for example: laws dated back to the 1900's), changing stakeholder parties in the governmental bodies which decide on changes in laws and regulations (i.e. the Dutch parliament), social pressure on existing stakeholder parties, and/or societal issues that arise (Blankena, 2012). Another problem arises where western countries are affected by different levels of law and regulations. Concerning the Dutch context, the life of Dutch civilians is affected by local law and regulations, national law and regulations, but also supranational (European Union level) law and regulations (Tarantino, 2008). The existence of these three layers of laws and regulations is not the only indicator of complexity. Complexity of laws and regulations adds up when it is being implemented and maintained situationally, meaning that each of the layers potentially differ from each other (for example: per country, municipality, etc.) (Tarantino, 2008).

Business triggers

In the Netherlands alone, examples can be seen rising as problems due to mismanagement of business rules and the architectures that the rules are built in. These problems result in huge cost factors since products and/or services cannot be delivered on time or with the appropriate quality (Algemene Rekenkamer, 2013; ANP, 2014; NVZ - Nederlandse Vereniging van Ziekenhuizen, 2013).

In general, the business benefits from agility when properly implemented. This is also the case for businesses that are characterized by using business rules. Not only the outside world, being the customers, will benefit from agility, but also the internal operations that need to operate the business rules that define the way their business behaves will benefit due to more efficient and agile working with changes in their business rule architectures. The business is seeking for answers concerning the agility of their, often situational, business rules architectures, and how agile modifiability can contribute to the products and/or services they deliver.

Businesses that are specialized in products and/or services which involve business rules to create added value are experiencing a lack of knowledge on business rule architectures. Examples of businesses that create added value utilizing business rules are Usoft (Urule), Everest (Cordys & Blueriq), IBM (IBM WebSphere ILOG JRules), and Oracle (Oracle Policy Automation). In general businesses have little fundamental knowledge on how to determine the potential impact of changes to business rule architectures, and architectural configurations with their influence on flexibility. It seems that the triggers which are addressed in earlier paragraphs are still too complex to solve for these businesses with their current knowledge on this topic.

1.2 Problem statement

In this paragraph, the problem statement and an objective are formalized for this study. Based on the research triggers we can conclude that organizations recognize business rules as a separate area of concern as it plays a significant role as part of their products and/or services. However, organizations are faced with an increase in frequency and volume of changes to their business rules due to changing laws and regulations and policies (Chapin, Hale, Khan, Ramil, & Tan, 2001; Zoet, Smit, & Leewis, 2015; Zoet, 2014). Instead of only taking into account the implementation-aspect, which comprises the technical implementation of business logic into products and/or services, organizations should also take into account the management-aspect, which comprises, amongst others, the modifiability of business logic (Boyer & Mili, 2011). Taking the modifiability-aspect into account provides organisations the means to effectively plan upon upcoming changes, and process modifications in such a way that organizational resources are most effectively employed to do so. Effectively, we include the notion of a business rule architecture since it is an under-researched theme in the current body of knowledge (Zoet, 2014). Managing large amounts of business logic as part of products and/or services increases the complexity of the management-aspect concerning business logic as well (Chapin et al., 2001). Currently, organizations face challenges regarding impact-analysis of upcoming changes. Difficulties arise when changes impact multiple products and/or services as cohesion between business logic elements are not properly documented, communicated, and utilized in the maintenance process as part of the evolution of the organization’s information system landscape (Zoet, de Haan, & Smit, 2014). Summarized, we address the following research statement in this research project:

> *"Current BR architectures are designed and implemented in an inflexible way due to the lack of proper impact-analysis up front and the inclusion of solely the implementation aspect, while excluding the modifiability aspect to cope with future changes in the BR architecture"*

1.3 Research objective

The focus of current literature on business rule architectures mainly lies on the technical perspective (Boyer & Mili, 2011; Liao, 2004; Morgan, 2002), thereby adhering to the definition that a business rule architecture is equal to a BRMS-architecture. However, from an information systems perspective, business rules are part of the logical design of information systems as well (Zoet, 2014). From both research and practice, authors recognize an increase in the volume of business rules, while the management of these business rules is also challenged with an increasingly higher change frequency. This throws up the question on how to structure business rules in such a way that they can cope with change in the most effective way. However, currently such a question is not addressed from a design perspective. Therefore we state that, when designing products and/or services, organizations should take into account the modifiability of the architecture of their products and/or services. To be able to do so our objective is to evaluate three business rule architectures, also referred to as architectural candidates, regarding their performance on modifiability. In section 3.4 we selected a method to evaluate the architectural candidates, which is based on effort prediction. The results of the analysis will enable us to quantitatively compare the architectural candidates regarding predicted effort and analyse the differences to reveal their modifiability characteristics. Summarized, the objective of this study can be formulated as follows:

> *"Evaluate business rule architectures regarding modifiability by utilizing effort prediction"*

Due to the limited scope of this study a minimum of three business rule architectures are evaluated. In the next section, the formal research question and related sub questions for this research project will be outlined and elaborated upon.

1.4 Research questions

For this particular study, we aim to explore the quality attribute modifiability concerning business rule architectures. Modifiability concerns change and according to Bass et al. (2012) four main questions are relevant when studying change:

1. What can change?
2. What is the likelihood of the change?
3. When is the change and who makes it?
4. What is the cost of the change?

The answers to the questions stated above are important to build a stable and cost effective architecture, as no architecture can be built to be able to cope with every modification possible.

Therefore an anticipated set of modifications needs to be defined which an architecture needs to cope with. An anticipated modification is "*a modification that results into a known amount of impact on system primitives*" (Mannaert & Verelst, 2009). Additionally, when modifications are anticipated, organizations are able to perform impact analysis regarding upcoming changes in laws and regulations and policies. Combined, our research triggers, problem statement, and objective brings us to the following research question and its corresponding sub-questions:

How can Business Rule architectures be designed for modifiability, taking into account the concept of anticipated modifications?

To answer the main research question, multiple concepts need to be analysed and the relationship between them needs to be established. Therefore, the main research question has been further divided into sub-questions formulated below. Additionally, some domain-related questions are added to clarify context of the research area that is not directly influenced by the research question. The following sub-questions are formulated:

- **SQ1. What is a business rule architecture?**
  With this sub-question, the main element within this research context is introduced. Within the BRM domain our subject of a business rule architecture is explored, both in its technical and logical form. This sub-question aims to provide an overview on the state-of-the-art on business rule architectures from literature and practice.

- **SQ2. What is modifiability?**
  As little is known in the BRM research domain concerning modifiability we aim to explore how neighbouring fields cope with the aspect of modifiability.

- **SQ3. What type of modifications could occur to a business rule architecture?**
  To properly evaluate modifiability of architectures, neighbouring literature suggests the use of a set of modification types. We adopt this by exploring which modifications are possible in theory and practice.

- **SQ4. How can modifiability of business rule architectures be measured?**
  Based on the exploration of neighbouring fields as covered by SQ2 we are able to further explore the neighbouring domains on how to measure to what extent an architectural candidate is modifiable.

- **SQ5. What business rule architecture is most optimal in terms of required effort to process anticipated modifications?**
  Based on our findings from all previous research questions we are able to perform the selected method of analysis to measure to what extend the included architectural candidates are modifiable.

1.5 Thesis outline

The remainder of this thesis is structured as outlined in Figure 2.

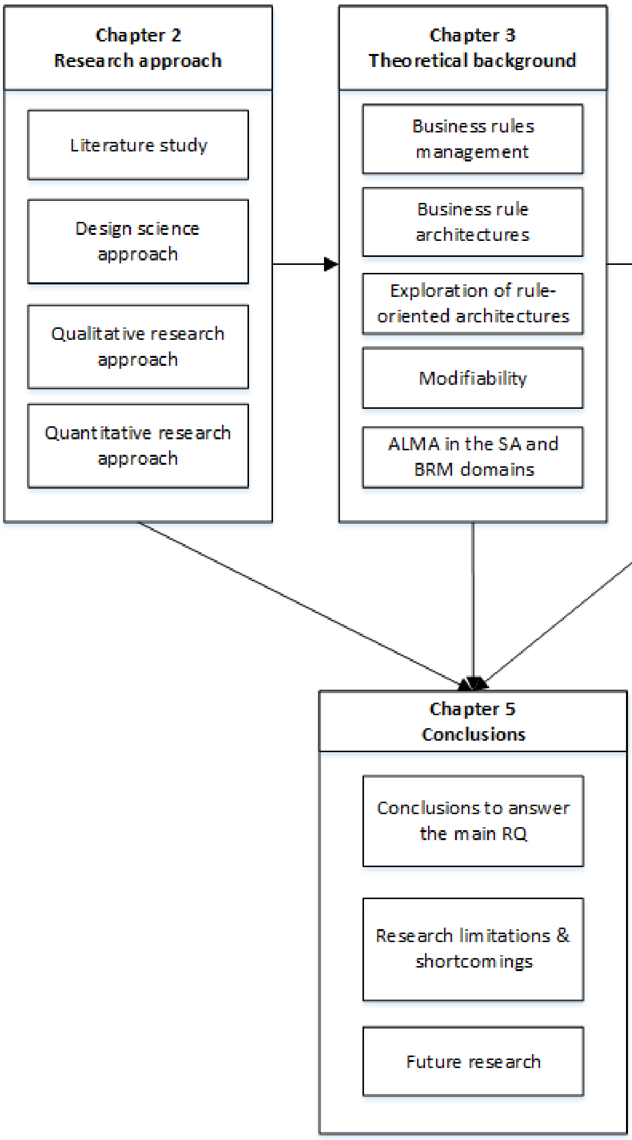


Figure 2 - Thesis outline

In chapter two we elaborate upon the research design of this research project, containing a mixed method approach. Chapter three comprises the theoretical background which contains a theoretical exploration of the BRM domain, an exploration and overview of the little available body of knowledge on business rule architectures, an overview of the concept of modifiability, and the transformation of data-oriented architectures into rule-oriented architectures for application in the BRM domain. Chapter four contains the method of analysis (ALMA) tailored for application in the BRM domain. Furthermore, the fifth comprises this study's conclusions, its limitations and opportunities for further research.

2 Research method

In this chapter, the research approach of this study is elaborated. In our research we are pursuing the validation of data architectures for appliance in the business rules management domain, exploring structures which are optimally serving a certain set of anticipated modifications. Also, we seek to adopt and adapt existing literature on analysing modifiability of architectural candidates that are applied as logical structures of products and/or services.

2.1 Literature study

An understanding of the existing knowledge base on business rules management and the architecture domain is necessary to properly elaborate our research questions and formulate relevant quantitative and qualitative research protocols. Furthermore, literature on the closely related concepts of evolvability, maintenance, and modifiability is added to determine what aspects should be examined or taken into account when performing analysis of modifiability of architectural candidates.

The search engines utilized for our literature retrieval were Google Scholar, Wiley Online Library, Taylor & Francis, PubMed, Springer Online, and IEEE Computer Society Digital Library. The consulted literature were limited to the English and Dutch languages. Depending on the research goals and approach an IS literature review type can be selected for this research. A recent study by Trudel, Jaana, Kitsiou, and Pare (2015), focusing on a typology of literature reviews, presents a rigorous overview of nine different literature study approaches, each aligned with a specific goal and research approach. From their work we identify the *Theoretical Review* to have the best fit according to the goal and approach selected and described in the preliminary research proposal for this study. Webster and Watson (2009) propose a theoretical review approach to be applicable to tackle an emerging issue that potentially would benefit from the development of new theoretical foundations. Such review approaches should be utilized not only for assembling and describing existing work, but also to develop conceptualizations or to extend current ones by identifying and highlighting knowledge gaps in existing literature (Eakin & Mykhalovskiy, 2003). This specifically fits our research since we already identified a knowledge gap in existing literature on the combination of business rules management and architectures, which constituted one of the main triggers for this research. For reliability purposes Table 1 depicts which search terms are utilized while querying the databases mentioned earlier in this section.

Table 1 - Literature keyword utilization

Business rules management	**Neighbouring fields on logical architectures**	**Quality attributes and modifiability**

• Separation of concerns
• Business rules management
• Business rules
• Business rule architectures
• Modification types for BRM
	• Logical architectures
	• Enterprise architecture
	• Software architecture
	• Data architecture
	• Data warehouse architectures
	• Architecture evaluation method
	• Architectural stability
	• Function points
	• Function point analysis
	• Project delivery rate
		• Architecture quality attributes
		  ○ Evolvability
		  ○ Maintainability
		  ○ Modifiability
		• Modification types
		• Change types
		• Architecture level analysis
		• Architecture modifiability analysis

A semi-structured approach is chosen to guide the literature review process. First, literature is selected based on title relevancy on the first five pages of all mentioned databases. From these results the abstract is analysed and considered for further analysis in the literature review process. Then again, from the abstracts it is considered whether or not to include the literature for further analysis, being reading the full work. It must be noted that there was a minimum threshold of 100 citations for a potential source to be included in further literature analysis, but this was rather a preference since this is not always possible to adhere to, especially with very recent publications from the end of 2014 and/or less mature research fields where literature on some topics is scarce. The total literature review resulted in 162 relevant papers or contributions that are included in this study.

2.2 Design science approach

This research project includes multiple new artefacts to be created. Creating new artefacts within the IS research domain is often carried out utilizing Hevner's Design Science framework (Hevner, March, Park, & Ram, 2004) to ensure a rigorous and valid approach of creating and validating new artefacts. Both the creation and validation of artefacts are the two main activities that characterize the design science approach (Hevner & Chatterjee, 2010). In Figure 3 we apply the design science framework to our research context.

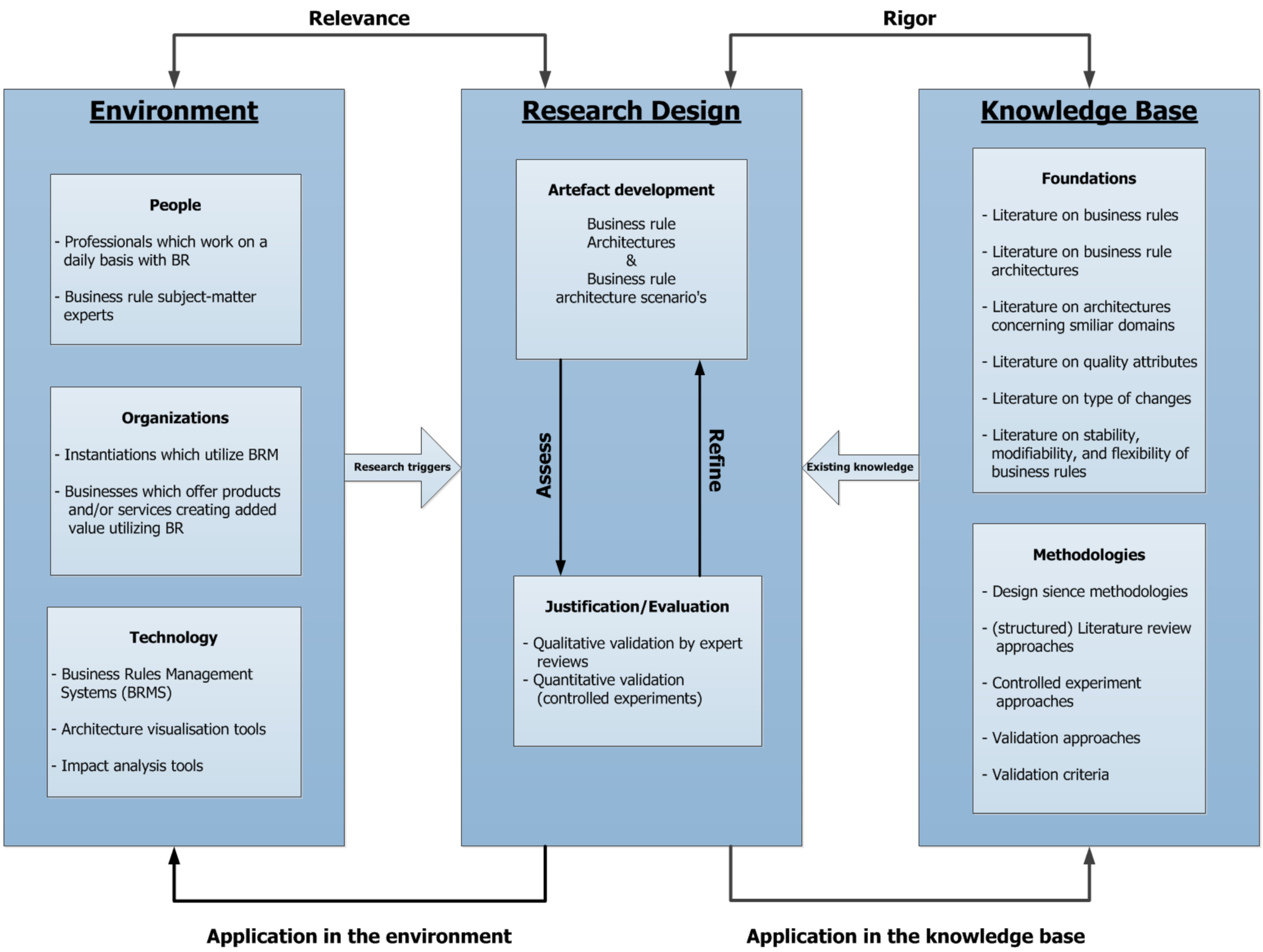


Figure 3 - Research design science framework, instantiated for this study (Hevner et al., 2004)

This research is conducted in an environment consisting out of organizations that employ professionals which are specialized in working with business rules. More specifically, this research focuses on the logical modeling of business logic by business or non-technical users. Furthermore, the current knowledge base consists of theories regarding business rules management, business rule architectures, architecture in the IS domain, quality attributes, and modifiability. Complementary to these theories we utilize well know research methods to guide the research project. Effectively, in this thesis we describe the design of three artefacts. The first artefact comprises the exploration of data-oriented architectures for application as rule-oriented architectures. The second artefact comprises the architectures instantiated regarding the selected case organization. The third artefact comprises an adapted measurement instrument to measure modifiability characteristics of business rule architectures. In this research project multiple cycles of validation are executed. Validation was performed utilizing two subject matter experts. The first expert has a PhD on the topic of BRM and has conducted seven years of research on the BRM domain. Furthermore, a second expert was included that conducted one and a half year of research on the BRM domain, especially on Business Rule Architectures. Results of the construction of the artefacts were discussed and validated on a weekly basis, in one hour sessions with both experts. Based on those meetings the created artefacts were shaped and adapted. Further (external) validation of the artefacts is further discussed in section 4.2.2.

2.3 Qualitative research

Literature analysis on modifiability shown that measuring modifiability of architectural candidates is very context-dependent and therefore requires various case-specific dimensions to be known. This requirement was not known at the start of the research project, but found very important to continue further analysis of architectural candidates. The targeted case-specific dimension concerns the productivity level (operationalized: Productivity scores) of business rules management experts employed at the case organization. Due to the nascent maturity of the business rules management domain we chose to acquire the missing case-specific data by utilizing a qualitative research method. The combination of the low maturity of the business rules management domain and the low amount of potential available respondents, consisting of only two individual business rule management experts processing case-specific matter into operational business rules, became the main arguments to select interviews as a technique to acquire the data needed to establish the productivity level dimension.

Generally, interviews provide greater detail and depth than the standard survey, allowing insights into hidden concepts and relationships. Additionally, interviews can be tailored specifically to the domain and the expertise-level of the respondent. According to Turner (2010) qualitative interview design can differ depending on different contextual characteristics. In this work, three different interview protocol archetype designs are elaborated upon: 1) informal conversational interview design, 2) general interview guide design, and 3) the standardized open-ended interview design. The data required to determine the productivity level dimension essentially comprise comparable ratio measurement scales (minutes/hours/days). This characteristic allows us to structure the interview as we want to compare the data and average the productivity levels for different modification types later on. However, we are also interested in contextual information like factors that can potentially or are known to influence the productivity levels of business rules management experts. Depending on the level of influence discovered, these factors could be taken into account or used to pinpoint future research directions.

Therefore the *Standardized Open-Ended Interview Design* is found to serve as the best fit for this research.

Standardized Open-Ended Interview protocol

The standardized open-ended interview is structured in terms of the wording of questions. Respondents are asked identical questions, but the questions are worded so that responses are open-ended (Gall, Borg, & Gall, 1996). This approach of interviewing provides data in a more rigorous way than compared to the lesser structured two alternatives, but still provides the researcher the possibility to probe follow-up questions to acquire fully the experiences and viewpoints of respondents.

Selection of respondents for this interview round is straightforward since only two respondents are applicable in the case of the selected case organization. The case organization employs three business rules experts as part of the QOF business rules team. Two of them are applicable since they really process requirements into operational business rules, thus performing the actual modifications. Due to the small number of respondents a pilot study is not desirable since it would consume too much time (Kvale, 2008).

The fact that organization that employs the respondents is based in Leeds, England, in combination with the provided resources for this study the mode of interview is confined to a digital webcam interview, i.e. Skype, or a telephone interview. While these modes of interview limit us to potentially miss non-verbal cues, we believe a telephone interview is an appropriate mode for this study. Trust and rapport is harder to establish when conducting a telephone interview. This is countered by extensive mail and telephone contact from the start of this research, as a mechanism to involve the respondents.

The full interview design can be found in Appendix A, containing 15 questions in total. Several factors have been important to take into account when creating the interview design. First, the skeleton of the interview design is made up out of 11 questions requiring the respondents to quantify the amount of time it takes to process a specific modification in their context. These questions are structured because they need to be quantitatively compared. For each of those 11 questions the respondents are allowed to elaborate on why it takes the specified amount of time and what factors do influence the process of processing the modification. Therefore the questions are initially positioned as structured, but are also characterized by an open-ended view for the respondents to elaborate more upon the specified time as an answer. Furthermore, the order of questions are designed in such a way that the respondents are first presented with simple questions (question 1 & 2), which provide some information about the experience of the business rule expert. The main body of the interview (question 3 – 13) consists of questions that are utilized to quantify how much time it takes to process a specific modification. These 11 questions are directly based on the work of (Zoet et al., 2015) on modification categories for business rules management, which utilized the same case organization thus being very applicable to serve as a foundation for this interview. The last two questions (question 14 & 15) are designed to be reflective on the factors that influence the answers given concerning the previous 11 questions and the interview itself. The last two questions are used as a means to provide the respondents room to comment on the topics, their answers or the researcher itself. Lastly, to ensure the respondents understand the modification types that are measured a case-specific example is created, featuring a business rules document with highlighted elements which are potentially subject to change, corresponding with the questions prompted.

The interview is prepared by sending the case-specific example three hours prior to the interview with the instructions to prepare for the interview and to keep it ready when the interview is conducted. This way the interviewer can guide the respondents trough the questions with the help of case-specific examples, increasing understandability of the topic itself, and decrease the chance that miscommunication occurs concerning the different modification types.

For recording purposes an audio recorder is utilized to record the telephone interview. Complementary to the audio recorder, the interviewer will take notes.

2.4 Quantitative research

Within the available body of literature several analysis methods to determine or compare certain architectural properties exist (Babar & Gorton, 2004; Dobrica & Niemela, 2002; Kazman, Bass, Klein, Lattanze, & Northrop, 2005). To determine to what extent an architectural candidate is modifiable the architecture-level modifiability analysis-method (ALMA) (Bengtsson, Lassing, Bosch, & Van Vliet, 2004) is utilized in this research. Arguments on the selection of ALMA for this research amongst other available methods is described in section 3.4. ALMA is a quantitative analysis based on architecture-level descriptions, which are generally characterized by high-level abstraction architecture designs. The analysis relies on realistic scenario's to determine the theoretical best-case modifiability, the theoretical worst-case modifiability, and the theoretical maintenance prediction for a given amount of time. The function of ALMA ideally suits as a technique to answer our research question as it provides us with the possibility to predict maintenance effort scores, which can be utilized for architectural candidate comparison. To be able to accurately utilize ALMA for BRM instead of software architectures several modifications to the method have to be made, which are described in section 3.3.5. Furthermore, to assess the three architectural candidates, several variables need to be identified as data collection challenges for this research. These variables are:

1) ***The size of a scenario;***
2) ***The weight of a scenario relative to the other scenarios;***
3) ***The total scenario profile;***
4) ***The total number of estimated modifications, and;***
5) ***The productivity scores for modification types.***

An overview on how we aim to acquire the data as described above is given in Table 2.

Table 2 - planned data collection strategies for each required ALMA dimension

Dimension 1: Scenario size	The scenario size is determined by function point analysis and expressed in function points for the business rules management domain. Function points are based on business rule complexity instead of lines of code as a standard measurement in the software engineering domain. Data collection to determine the scenario size will focus on the analysis of built business rule architectures in the three selected notations with their underlying structures. These business rule architectures allow us to determine the effect size, which is performed in step four of the ALMA-method, of a given scenario. To be able

	to create the business rule architectures two main sources are needed, subject matter knowledge of the business rules documents and subject matter knowledge on building architectures with the three selected architectural candidates. Essentially, creating the business rule architectures only requires literature so no further quantitative data collection is required for this dimension.
Dimension 2: Scenario weight	The scenario weight is determined by the amount of a specific modification occurrences in a given time period as observed in data which is divided by the total number of modifications occurred. The data comprises historic data on modifications made by the case organization concerning the business rules documents.
Dimension 3: Scenario profile	The scenario profile is determined by the total amount of scenarios as a result of the scenario elicitation phase of the ALMA-method, thus it is dependent on the sum of results from dimension 2 as described above. No further data collection is required to determine this dimension.
Dimension 4: Estimated modifications	The total amount of estimated modifications are dependent on the timeframe selected for maintenance prediction. For example, if a timeframe of two years is selected the prediction of maintenance holds for a period of two years. Depending on the available data the total amount of modifications are added together to determine the total amount of estimated modifications. This data will be collected through collecting and analysing change documents concerning the business rules documents published by the case organization.
Dimension 5: productivity scores	For the productivity scores we utilize interviews as described in section 2.3 in this chapter.

2.5 Research case description

As stated in chapter one, our objective is to evaluate three business rule architectures regarding modifiability by utilizing effort prediction. To be able to ground our findings we need to utilize realistic and relevant case data as input for our research. Earlier research performed on the topic of change management for BRM revealed that the National Health Service of the United Kingdom is an eligible case organization for further research regarding BRM (Zoet et al., 2015). This medical case was found applicable for this study for several reasons: 1) ***the size of the case***, averagely containing over twenty diseases accompanied by business rules documents, covering over 100 disease indicators, 2) ***the timeframe of available documentation***, concerning eight consecutive years (2006 – 2014), and 3) ***the completeness of the available documentation***.

Primary care in the United Kingdom

Until the 1980's, no real standards existed for general practices in the UK, where general practitioners (hereafter: GP's) almost entirely acted on their own conscience (Department of Health England, 2013; Lester & Campbell, 2010). However, in 1998 this had changed due to multiple policy drivers (Lester & Campbell, 2010). The necessity of the QOF was triggered by three main developments concerning primary care in the UK, namely: 1) the rise of evidence-based medicine, 2) a change in the professional culture of GPs towards recognizing variation in the quality of primary care, and 3) the underfunding compared to other countries (Allen, Mason, & Whittaker, 2014; Lester & Campbell, 2010). The debate in the UK continued for a number of years before a solid consensus could be reached with stakeholders (i.e. UK government, the British Medical Association, insurers, and patient organisations). In the 1990s, a first step was made by the introduction of audit as a requirement for general practices. Close after these measures were introduced in the 1990s, audits were performed rather poorly. Over time, the auditing of primary care practices grew in maturity while GPs, although slowly, got accustomed to measuring performance. In that time, research studies revealed widespread deficiencies in quality of care across the UK (Seddon, Marshall, Campbell, & Roland, 2001).

This triggered many GPs to measure their quality of care internally, only to conclude themselves that care could be improved (Gillam & Siriwardena, 2011). The adoption of IT, on a worldwide scale, also influenced the working methods of general practices in the UK at the end of the century. With primary care practices progressively investing in IT since the late nineties, GPs are provided with the possibility to actually measure quality of care, but more important, utilize the functionalities of digital reporting over manual labour concerning the quality of care delivered. The adoption of IT by primary care in the UK started utilizing computers for a recall system, prescribing and finally, clinical records. In the late 1990s, the UK government launched a plethora of initiatives to improve the quality of care across the NHS (Department of Health, 1998). A significant part of these fell under the umbrella of 'clinical governance' which focused on a more systematic approach of performing auditing. These developments ran parallel with the development of IT at both the primary care practices nationwide as the NHS (Gillam & Siriwardena, 2011). The early 2000s were characterized with primary care being accustomed to working with computers as well as measuring quality of care and working out ways of improving it. Then, in 2001, the government announced plans containing increasing the NHS funding to bring the UK to an average level (concerning developed countries) in terms of expenditure as a proportion of gross domestic product (GDP) (Gillam & Siriwardena, 2011). In the following four years, the UK government needed to establish a system or framework to increase the quality of care, while spending budget carefully in an efficient and effective manner. This comprised a complex process of negotiating the framework that is now defined as the Quality and Outcomes Framework (QOF), which will be elaborated in detail in the next sections.

The UK healthcare focuses on a system that is available to all, with access based on clinical need, rather than the ability to pay. Therefore basic healthcare is freely available to UK permanent residents. In most countries, unlike, e.g., the UK and the Netherlands, payment methods regarding healthcare is largely based on fees-for-service (Lester & Campbell, 2010). The implementation of the QOF as a P4P scheme is dependent on a number of contextual factors. Data entry and extraction are key drivers in the way P4P can be implemented. It is evident that such implementations require the P4P scheme to solely utilize electronic data entry and extraction following standardized read codes (Schoen et al., 2004). However, the level of digitalization of clinical information systems within general practices varies and is insufficient at, for example, in Australia, Germany, New Zealand, and the United States to use this method (Lester & Campbell, 2010). It is key that certain baseline data is available, supported by computerized general practice clinical systems to qualify for utilizing a P4P and reap its benefits (Gillam & Siriwardena, 2011).

Funding of healthcare related activities performed by providers is paid out of a fixed budget which is projected on a yearly basis by the British government in collaboration with the secretary of state for health (Department of Health England, 2013). The budget for healthcare for 2013/2014 was around 108,9 billion pounds (NHS England, 2015). The pay-for-performance scheme is utilized to remunerate providers in primary care according to the amount and quality of care delivered. The primary care system in the UK experiences a throughput of one million patients in a period of 24 to 36 hours (NHS England, 2015).

The National Health Service of the United Kingdom

The healthcare system of the UK is facilitated by the National Health Service (Hereafter: NHS), which was founded in 1948 (Department of Health England, 2013). It consists of four organizations: NHS England, NHS Wales, NHS Scotland, and NHS Northern Ireland. The NHS is the world's largest publicly funded health service (NHS England, 2015). The NHS is responsible for delivering care for more than 63,2 million people. Furthermore, the NHS employs over 1.7 million people (Department of Health England, 2013). Due to the scale of the NHS, some explanation of the scope is provided in context of the UK healthcare system. The NHS is made up of a wide range of organisations specializing in different types of medical services for UK permanent residents. Collectively, these (medical) services deal with over one million patients every 36 hours (Department of Health England, 2013). Providers of primary care are the first point of contact for healthcare related issues, in non-urgent cases. These include general practitioners (hereafter: GPs), dentists, opticians, and pharmacists. Concerning the GPs as a part of the NHS, over 36.000 are active in England, spread over 8,300 practices as of 2010 (General Medical Council, 2015).

The QOF is established to achieve a certain level of quality of healthcare provided by the healthcare system of the UK. Quality is defined by the National Health Service as: "*a combination of good medical outcome supported by evidence, safe care, and good patient experience*" (Department of Health England, 2013). Organizations that meet the NHS quality and financial standards are able to provide services which are funded by the NHS. These organisations are often referred to as 'providers'. For providers to be able to receive funds of the NHS, providers have to belong to NHS foundation trusts. These foundation trusts all have a board of directors which are responsible for the quality of healthcare delivered by the provider, while being directly accountable for their results. The performance of these trusts are externally monitored by members of the foundation trusts. Local population, for example, patients, service users, carers, staff, and family members of patients, can become members of the foundation trust so that it is represented by different relevant stakeholders. The foundation trust is represented by a council of governors, which are elected by the members of the foundation trust themselves. In a foundation trust, the council of governors oversees the performance of the provider, holding the board accountable for their organization. In this case study, boards are not to be taken literally, as a group of stakeholders. For example, GP practices usually do not have a board, but are rather represented by a person who is legally accountable. Important to note is the scattered situation concerning healthcare responsibilities. Within the UK, England, Northern Ireland, Scotland, and Wales each have their own, slightly different, governance and systems of publicly funded healthcare. This is largely due to the fact that these different regions having different policies and priorities. Regarding the QOF, all four regions apply the same system, but differ substantively. For example, Scotland and England both apply different achievement point publications (the height of achievement points available per disease indicator) or enabling different payment ranges (determination of achievement of practices fall within the thresholds or not) (British Medical Association, 2015). This situation is sometimes referred to as 'the devolved countries'.

The Quality and Outcomes Framework

The QOF is a P4P-scheme covering clinical, organizational, and patient areas in primary care. Clinical conditions are developed utilizing evidence-based-medicine studies, while the organizational areas are developed utilizing management practices and governmental policy changes. The patient areas represents patient satisfaction measurements regarding their practice (Lester & Campbell, 2010). The QOF is the largest P4P scheme for primary care in the world (Allen et al., 2014) and is based on five principles covering different aspects of care evaluation which represent the founding principles of the NHS. It involves (Department of Health England, 2013):

1) **The prevention of people dying prematurely;**
2) **Enhancing the Quality Of Life (hereafter: QOL) for people with long-term or chronic conditions;**
3) **Helping people to recover from ill health;**
4) **Ensuring that people have a positive experience of care;**
5) **Treating and caring for people in a safe environment and protecting them from avoidable harm (i.e. human errors in healthcare).**

Measurements within the QOF are performed utilizing 'Indicators'. These indicators represent the clinical, the organizational, or the patient domains while taking into account the domains as described earlier.

Although the QOF is a voluntary system, more than 99% of UK practices participated in 2010 (Gillam & Siriwardena, 2011). The exact remuneration height of a single achievement point is determined by the Review Body on Doctors and Dentists and is updated for each financial year based on inflation rate and other changes regarding healthcare budget commissioned by the UK government. The pounds per achievement points ratio was 160.15 £ / achievement point for 2015/2016 (British Medical Association, 2015). Each year, each practice must complete a report containing the recorded level of achievement together with evidence for that. The information provided by the practices is scrutinized and further evidence concerning the data may be sought utilizing random checking of records or qualitative interviews with practice staff regarding practice policies (HSCIC, 2012). The data is collected nationally by QMAS on 31th of March, the end of the UK financial year, which is followed up by calculation of the total achievement points for each practice on the 2nd of April on each financial year. This is sometimes referred to as 'National Achievement Day' (Simon & Morton, 2010). The final remuneration for care delivered according to the amount of achievement points is adjusted to standardize results. Therefore the total amount of achievement points per clinical indicator are converted to practice prevalence figures and are compared to national prevalence figures.

Prior research shows that the clinical domain is subject to a high frequency of chances compared to the organizational and patient domains (Lester & Campbell, 2010; Starfield & Mangin, 2010). Therefore only the clinical domain, together with its corresponding disease indicators are being addressed in this research. At the time of writing, the clinical domain contains 19 clinical conditions, with a large amount of underlying indicators, which make up for 80 percent of the commonly encountered health issues in primary care (Gillam & Siriwardena, 2011). The remaining 20 percent of health issues that are encountered in primary care is too volatile for indicator development (Gillam & Siriwardena, 2011). The most recent published QOF includes the following clinical conditions: Atrial fibrillation (AF), Secondary prevention of coronary heart disease (CHD), Heart failure (HF), Hypertension (HYP), Peripheral arterial disease (PAD), Stroke and transient ischaemic attack (STIA), Diabetes mellitus (DM), Asthma (AST), Chronic obstructive pulmonary disease (COPD), Dementia (DEM), Depression (DEP), Mental health (MH), Cancer (CAN), Chronic kidney disease (CKD), Epilepsy (EP), Learning disabilities (LD), Osteoporosis: secondary prevention of fragility fracture (OST), Rheumatoid arthritis (RA), and Palliative care (PC) (HSCIC, 2015). An example of a disease with underlying disease indicators is depicted in Table 3. A similar structure is adhered to for the other clinical conditions.

Table 3 - QOF clinical condition with corresponding clinical indicators

Disease	**Disease indicator**	**Description**
COPD	COPD 1	The practice can produce a register of patients with COPD.
	COPD 2	The percentage of patients with COPD (diagnosed on or after 1 April 2011) in whom the diagnosis has been confirmed by post bronchodilator spirometry between 3 months before and 12 months after entering on to the register.
	COPD 3	The percentage of patients with COPD who have had a review, undertaken by a healthcare professional, including an assessment of breathlessness using the Medical Research Council dyspnoea scale in the preceding 12 months.
	COPD 4	The percentage of patients with COPD with a record of FEV1 in the preceding 12 months [*FeV1 identifies another clinical disease code*]
	COPD 5	The percentage of patients with COPD and Medical Research Council dyspnoea grade ≥3 at any time in the preceding 12 months, with a record of oxygen saturation value within the preceding 12 months.
	COPD 7	The percentage of patients with COPD who have had influenza immunization in the preceding 1 August to 31 March.

Business rules in the QOF

As described earlier, the QOF of the new GMS contract is recorded on the national system QMAS. Business rules come into play when data is hold against certain logic to produce outcome, where facts regarding payment of provided medical services are created by business rules. In the QOF, business rules are utilized to report on achievement in primary care. The business rules detail the logic and sequence in which the numerator and denominator of a certain indicator is extracted to determine the indicator achievement. The business rules are created for each unique indicator that contains information required to identify:

1) **Patients that are eligible for inclusion on the disease register;**
2) **Patients that are eligible to receive care;**
3) **Patients which actually received care.**

Any modifications to the QOF business rules are subjected to a formal review process including representatives from the four UK health departments, NHS employers, the General Practitioners Committee (mostly IT committee), system suppliers, and other technical or coding experts (Gillam & Siriwardena, 2011). The QOF business rules are updated twice a year, usually in April and October (NHS Employers, 2014; NHS England, 2014). An overview of the process regarding the calculation of remuneration by the QOF is depicted in Figure 4.

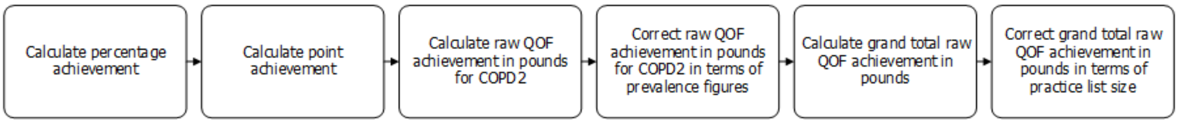


Figure 4 - Overview of the calculation of remuneration as part of the QOF

Following the COPD example shown in previous sections, a business rule set concerning COPD2 is provided in the following paragraphs. The example of COPD2 features, like other clinical indicators for most clinical conditions, four rulesets which process one single electronic patient record of a practice. The rulesets are:

1) **Patient registration status;**
2) **Diagnostic code status;**
3) **Denominator;**
4) **Numerator.**

The first ruleset as depicted in Table 4 comprises logic which identifies patients who are registered at the practice for GMS care at the point at which the data extraction is conducted (date of extraction is referred to as reference date). This determines the current patient population at the practice. To be included, a person needs to be registered for GMS care by having registered for this before or equal to the reference date. This is to ensure that patients are only registered at one practice at a time. Furthermore, the ruleset checks registered patients which registered before the previous reference date on deregistration after the previous reference date.

Table 4 - Patient registration status ruleset of the clinical condition COPD

Current registration status	***Qualifying criteria***
Currently registered for GMS	Most recent registration date <= (ACHIEVEMENT_DAT)
Previously registered for GMS	Any sequential pairing of registration date and deregistration date where both of the following conditions are met: registration date <= (ACHIEVEMENT_DAT); AND deregistration date > (ACHIEVEMENT_DAT)

The second ruleset as depicted in Table 5 comprises logic which determines the current patient population for a specified clinical condition out of the register of clinical records outputted by the previous ruleset. The outcome of this identifies the register size for the clinical condition. The example in Table 5 features logic to check whether a patient is registered as a patient diagnosed with COPD. This is performed by comparing the clinical codes from the patient records against the designated clinical codes -or code ranges, for example: H3...- under COPD.

Table 5 - Diagnostic code status ruleset for the clinical condition COPD

Code criteria	***Qualifying diagnostic codes***		***Time criteria***
Included	*Read Codes v2*	*CTV3*	Earliest <= (ACHIEVEMENT_DAT)
	H3...	H31..%	

	H31..% (excluding H3101, H31y0, H3122) H32..% H36.. - H3z.. (excluding H3y0., H3y1.) H5832	H32..% (excluding XaIQg) H3...% (excluding XE0YL%, H3122%) Xaa7C	

The third ruleset as depicted in Table 6 comprises logic of the denominator. Within a clinical condition, multiple denominators exist. Each denominator represents a clinical indicator and identifies which clinical records are entitled to receive care out of the register of clinical records outputted by the previous ruleset. For example, some people suffer from very mild form of a clinical condition. Another reason for a record to be excluded are patients who fail to attend reviews planned by their GP more than three times in the past year. Essentially, a denominator is the target population minus exclusions and exceptions. Exclusions are patients who are removed from the denominator of an indicator because they are ineligible for the care described in the indicator. For example, an indicator –and therefore its corresponding denominator- may refer only to patients of a specific age group, patients with a specific status (e.g. smoking status), or patients with a specific length (NHS England, 2014). Exceptions, are part of the exception reporting mechanism which allows GPs to remove patients who would otherwise be eligible for the care described in the denominator. Exceptions are only permitted when patient characteristics meet a certain set of strict criteria. More information on this can be found in the work of (NHS England, 2014).

Table 6 - Denominator ruleset of the clinical indicator COPD2

Rule number	Rule	Action if true	Action if false
1	If COPD_DAT >= 01.04.2011	Next rule	Reject
2	If COPDSPIR_DAT >= (COPD_DAT – 3 months) AND If COPDSPIR_DAT <= (COPD_DAT + 12 months)	Select	Next rule
3	If REG_DAT > (PAYMENTPERIODEND_DAT – 3 months)	Reject	Next rule
4	If COPDEXC_DAT > (PAYMENTPERIODEND_DAT – 12 months)	Reject	Next rule
5	If SPEX_DAT > (PAYMENTPERIODEND_DAT – 12 months)	Reject	Next rule
6	If COPD_DAT > (PAYMENTPERIODEND_DAT – 3 months)	Reject	Select

The fourth ruleset as depicted in Table 6 comprises logic of the numerator. Within a clinical condition, multiple numerators exist, but each is dependent on the outcome of its corresponding denominator. The numerator identifies the clinical records which did receive care out of the register of clinical records outputted by the denominator. The resulting register size represents the total register size of clinical records which are registered correctly, contain at least one of the corresponding clinical codes for the diagnosis of the clinical condition COPD, Are eligible for receiving care, and received care in the past financial year.

Table 7 - Numerator ruleset of the clinical indicator COPD2

Rule number	Rule	Action if true	Action if false
1	If COPDSPIR_DAT >= (COPD_DAT – 3 months) AND If COPDSPIR_DAT <= (COPD_DAT + 12 months)	Select	Reject

Remuneration of care
Care services within the context of the QOF are remunerated according to a payment scheme. The payment scheme of the QOF is activated once the numerator provides results as the register size, described in the fourth ruleset and is identical for all diseases. The payment scheme mainly consists of five main decisions: 1) *calculation of the practice prevalence*, 2) c*alculation of the percentage achievement*, 3) *calculation of the point achievement*, 4) *calculation of the raw QOF achievement in pounds* and, 5) *calculation of the final QOF achievement in pounds*. The payment scheme is executed per clinical indicator which adds up to the total remuneration per clinical condition. Finally, adding up all QOF achievement of all clinical conditions generates the total remuneration for a given period or financial year.

1. The calculation of the percentage and point achievement is performed utilizing a threshold system. Thresholds are (re)defined on a yearly basis by the NICE and are published by the department of health in the QOF guidance for the new GMS contract of the corresponding year. All indicators have a threshold value or lower and higher threshold assigned. When a threshold value is chosen, all points are rewarded upon achieving the stated threshold percentage. When a range between a lower an higher threshold is chosen, a minimum threshold and a maximum threshold are implemented, meaning three states for payment can be reached: 1) *no payment due to the percentage achievement is too low*, 2) *proportional payment because the achievement is between the lower and upper threshold*, and 3) *maximum payment due to reaching the upper threshold or higher*. For example, the practice has 2255 COPD patients in the denominator from which 2000 received care in the previous year. The achievement percentage regarding these figures are:

2000/2255 = 88.69%

2. The range of COPD2 for 2014-15 is set for a lower threshold of 40% and upper threshold of 90%. The raw percentage achievement is subtracted with the lower threshold and divided by the threshold range (upper threshold – lower threshold), so for this example:

(88.69% - 40% = 48.69%) /50% = 0.9738

The result of this formula is then multiplied by the maximum achievable points for the indicator, which is seven in this example:

0.9738 * 7 = 6.8

3. The calculation of the raw QOF achievement in pounds is performed by multiplying the baseline achievement in pounds per point achievement times the raw achievement score. For this example, we take the baseline value per of £150.55 per achievement point, resulting in:

6.8 x £150.55 = £1023.74

Depending on practice and national prevalence figures, the raw QOF achievement in pounds is corrected.

4. To correct the raw QOF achievement in pounds according to achievement against the national average, the practice prevalence is compared to the national prevalence. The national prevalence is a percentage provided by data from the department of health, which is in this example: 18.6%. The practice prevalence is calculated by dividing the register or target population by the total registered list size and converting it to a percentage. For this example we have a total list size of 10.566, while the denominator COPD2 list size was 2255, so:

2255/10.566 = 0.2134

This figure is converted into a percentage which results in 21.3% practice prevalence. The raw achievement is therefore multiplied by the results of dividing the practice prevalence with the national prevalence:

(21.3/18.6) = 1.14516 * £1023.74 = £1172.34

Lastly, practice versus national average list size correction is the last correction over the raw QOF achievement in pounds performed, which takes the scale of the practice into account. The national average practice list size is generated on a yearly basis and published in the Statement of Financial Entitlement. In practice, this calculation is applied when all indicator's achievement in pounds is calculated and practice/national prevalence correction was performed. The national average list size for previous year is 5844. The list size of the example practice is 6578. For this example, calculation of the clinical indicators Diabetes1, Diabetis2, and Asthma5 resulted in a total of £5639.88 achievement by the practice. Adding all clinical indicator achievement up, including the calculated example of the clinical indicator COPD2 as shown in the previous steps elaborated, the grand total QOF achievement in pounds results into:

£5639.88 + £1172.34 = £6812.22

Now that the requirement to be able to correct the grand total achievement in pounds have been met (all calculations have been performed, a grand total of clinical indicators is created) the practice list size can be compared with the national average practice list size, meaning:

6578 / 5844 = 1.12

Lastly, this ratio is multiplied with the grand total achievement in pounds:

£6812.22 x 1.12 = £7629.68

2.6 Case selection

For this research it is not necessary to include all business rule documents of all clinical conditions in the QOF. Therefore as part of the NHS as our case organization we select two representative clinical conditions, which are defined as cases from now on. In this section the selection of those cases is further substantiated. The selection of cases, i.e. relevant and representative clinical conditions with corresponding clinical indicators, should be based on the group of individuals, organizations, information technology, or community that best represents the phenomenon studied (Strauss & Corbin, 1990). To determine this, four criteria are taken into account:

1. Case size – Amount of pages and amount of business rules per clinical condition.
   This criteria is selected to determine if the case is significant in size to represent the average size of the case population. The descriptive statistics for this criteria are presented in Table 8.

Table 8 - Case size descriptive statistics

	Pages Lowest (top 3)	**Pages Highest (top 3)**	**BR's Lowest (top 3)**	**BR's Highest (top 3)**	**Average**
V 8	1. Cytology (4) 2. Obesity (4) 3. Cancer (6)	1. Diabetes (29) 2. CHD (20) 3. COPD (12)	1. Obesity (13) 2. Cytology (16) 3. Dementia (16)	1. Diabetes (222) 2. CHD (153) 3. COPD (59)	11 pages & 63 BR's
V 19	1. Cytology (4) 2. Obesity (5) 3. Cancer (6)	1. Diabetes (39) 2. CHD (28) 3. CVD (15)	1. Obesity (13) 2. Cytology (16) 3. Cancer (19)	1. Diabetes (207) 2. CHD (111) 3. CKD (64)	14 pages & 58 BR's
V 29	1. Blood pressure (3) 2. Obesity (4) 3. Cervical screening (5)	1. Diabetes (36) 2. CHD (25) 3. CVD (17)	1. Obesity (14) 2. Blood pressure (16) 3. Cervical screening (16)	1. Diabetes (112) 2. COPD (74) 3. CHD (67)	15 pages & 40 BR's

2. Case modification history – Amount of modifications to the case in the total case timeline
   This criterion is selected to determine if the case is relevant to include based on the amount of modifications made to the case in the time period of 2006 until 2014. The descriptive statistics for this criteria are presented in Table 9.

Table 9 - modification history descriptive statistics

Modifications Lowest (top 3)	**Modifications Highest (top 3)**	**Average**
1. Blood pressure (22) 2. Cervical screening (39) 3. Obesity (41)	1. CHD (807) 2. Diabetes (766) 3. CVD (336)	240 modifications

3. Practical relevance - Relevance for the practical domain and its societal importance.
   The WHO is the world's largest authority for health, directing and coordinating health within the United Nations system. The WHO has a considerable influence on the world's health research agenda. In 2013, the WHO published their global action plan for the prevention and control of noncommunicable diseases, setting goals for the period of 2013 – 2020. Based on these findings we selected, amongst others, COPD, Diabetes Mellitus, CHD, and Cancer as relevant for the practice from an international perspective. For a national view we utilize the resources of the Nederlands instituut voor onderzoek van de gezondheidszorg (NIVEL), which collects, analyses, and reports on healthcare data to improve the Dutch healthcare services. Based on the extra attention of the NIVEL for a selection of six most important clinical conditions (NIVEL, 2015b) together with the registered prevalence and incidence statistics of 2013 (NIVEL, 2015a) and the Dutch national figures from the National Institute for Public Health and the Environment (2013) we selected COPD, Diabetes Mellitus, and CHD as high relevancy clinical conditions. These findings are similar to the healthcare situation in England (Office for National Statistics, 2013).
4. Scientific relevance – Relevance for the scientific domain and the amount of research conducted for the specific clinical condition.
   To be exact about the scientific relevance regarding the included clinical conditions a full literature research has to be conducted. Instead, we only take into account the top three largest cases, highest amount of business rules, and highest amount of identified modifications from our descriptive statistics as presented in Table 9. From the top three diseases, Diabetes is a very relevant disease for the scientific domain. Furthermore, according to Gillam and Siriwardena (2011) Diabetes Mellitus is the most studied QOF clinical condition, while COPD has also received significant attention from the scientific community.

Taking into account the criteria and the descriptive statistics as presented in Appendix G we select COPD and Diabetes as the most relevant case conditions to process into three different business rule architectures. The following three stages will describe the process of architecture description creation for the clinical condition cases COPD and DM.

To be able to predict the total maintenance effort of the included architectural candidates certain descriptive statistics are needed. First, the total amount of modifications for both clinical indicators need to be derived. Next, the total amount of modifications need to be distributed across the eleven identified modification types from earlier research (Zoet et al., 2015), see Table 10 concerning COPD and Table 11 concerning Diabetes Mellitus. The total Modification types are derived from a sample of eight consecutive years in total.

Table 10 - Descriptive statistics for the clinical indicator COPD

Clinical indicator: COPD										
CD	DD	UD	CBR	DBR	CC	DC	UC	CFV	UFV	DFV
6	0	17	3	1	4	3	69	73	67	49
Total MO's		292								

Table 11 - Descriptive statistics for the clinical indicator Diabetes Mellitus

Clinical indicator: Diabetes Mellitus										
CD	DD	UD	CBR	DBR	CC	DC	UC	CFV	UFV	DFV
51	58	3	50	34	2	169	100	175	103	21
Total MO's		766								

Chapter conclusion

In conclusion we utilize a mixed method approach to analyse the modifiability of three architectural candidates as used in the BRM domain; the rule family-oriented, fact-oriented, and decision-oriented architectures. A literature review is conducted to explore the current state-of-the-art regarding business rule architectures. Furthermore, the literature review is utilized to give meaning to the concepts of business logic and business rules, architectures, quality attributes, and modifiability. Lastly, we used the literature review to ground our selection of available methods to analyse modifiability of architectural candidates. Based on this we selected ALMA as our method of analysis, which is grounded on the criteria presented in section 3.4. Utilizing ALMA requires the collection of several variables needed to quantify how much effort is required to process the predefined scenarios containing modification types. As the architectural candidates were not created we selected the design science approach (Hevner et al., 2004). By adhering to the distinctive cycle of creation and validation of artefacts we managed to employ eight rounds of validation with two BRM experts before our architectural candidates were deemed valid for the analysis of modifiability. Furthermore, ALMA is dependent on the productivity of the professionals processing the modifications as part of the scenarios. To ground our research and improve generalizability we did not choose to utilize general figures from earlier research, but instead designed and conducted two semi-structured interviews to fully reveal the context-specific productivity levels and its contingency factors at the NHS. Next to the productivity levels as input for ALMA we utilized quantitative data from an earlier study on the NHS's QOF (Zoet et al., 2015). Additionally, some variables required by ALMA are manually gathered and/or calculated. Analysing the full QOF as part of the NHS remuneration program would not fit in the scope of this research. Therefore we utilized four criteria, as described in 2.6, to derive two relevant cases as ground for the creation of our architectural candidates. Based on those criteria we selected the clinical conditions COPD and Diabetes Mellitus.

In the following chapter, comprising the theoretical background for this study, we present the results of the literature review as elaborated upon in section 2.1.

3 Theoretical background

The results of the first phase of this research, the literature review, are elaborated upon in this chapter. The findings of the literature review described in this chapter provide a foundation for further research into the domain of business rule architectures. To ground our described concepts we constructed a meta-level overview of the concepts described in this study together with their interrelationships, depicted in Figure 5.

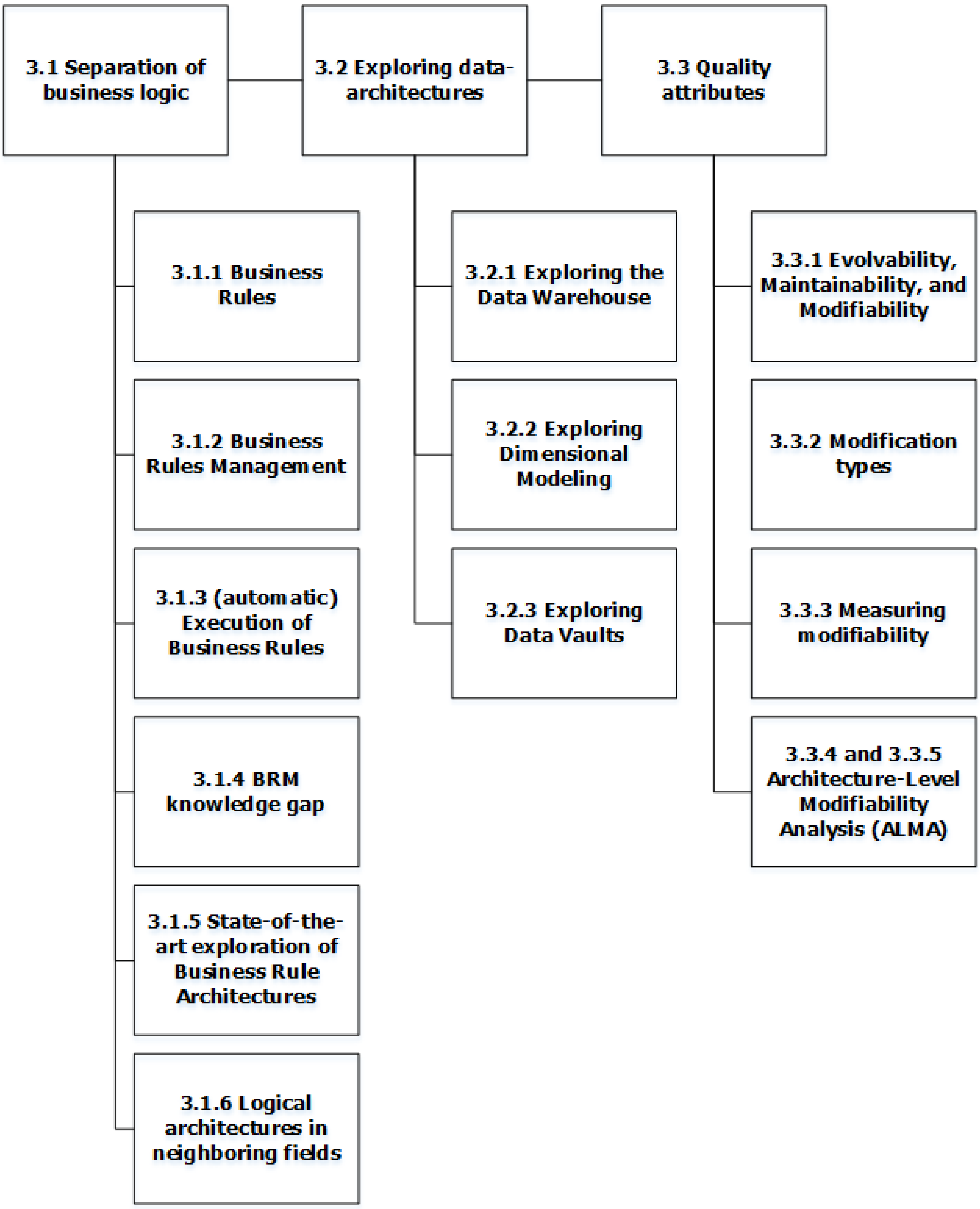


Figure 5 - Conceptual overview of the subjects presented in the theoretical background chapter

3.1 Separation of business logic

The primary goals of software engineering are to improve software quality, reduce software production costs, and facilitate maintenance and evolution. To achieve these goals, organizations constantly seek for development technologies and methodologies that add value in terms of complexity reduction, increase comprehensibility, promote reuse, and facilitate evolution of software systems (Tarr, Ossher, Harrison, & Sutton Jr, 1999). As these goals are all contributing to the overall perceived quality of software systems, mechanisms utilized for these goals are often conflicting of nature. This creates problems that complicate software engineering further. In their work, Ossher & Tarr (2001) indicate that these problems are related to separation of concerns, as coined by Dijkstra back in 1974 (1974). The ability to achieve the goals that represent different concerns depends on the ability to keep and manage separate all concerns of importance of software engineering. In his work regarding workflow management systems, van der Aalst (1998) also discussed the concept of separation of concerns, introducing the notion of Workflow Management Systems (WFMS) as a logical next step in separation of concerns from applications. He argues that researchers should look back in history to observe the evolution of the concept of separation of concerns. In the sixties, information systems where no more than a number of stand-alone information systems built upon operating systems. In the seventies, the advantages of separate data management gained attention from research and practice. This generally led to the separation of Database Management Systems from the application layer as a unique entity in software engineering. In the 80-ties, a similar observation can be made for the management of GUI's, which led to the separation of User Interface Management Systems from the application layer. Following these general observations regarding software engineering evolution, van der Aalst (1998) proposed the separation of processes from the application layer as well, introducing the concept of Workflow Management Systems, stating: "*In our opinion WFMSs are the next step in pushing generic functionality out of the applications. The 90-ties will be marked by the emergence of workflow software, allowing application developers to push the business procedures out of the applications*".

Although the separation of the flow (workflows) was a logical step in separating concerns we argue that separating business logic from applications is similarly logical and imperative as a next phase of separation of concerns. Business logic is defined as:

> *"Business logic is that portion of an enterprise system which determines how facts are transformed, calculated, and/or routed to people or software systems"* (Von Halle & Goldberg, 2009)

Applications and services must be flexible in order to cope with the ever changing business situations, policies and offered products (Vanthienen, 2001). Obtaining and maintaining this flexibility poses a challenge for businesses. This especially holds for business logic containing business rules. Currently, in many situations it is hard to modify the underlying rules (as part of business logic) of businesses due to the fact that these are often hard-coded or buried far into programming code (Charfi & Mezini, 2004; Vanthienen, 2001; Zoet, 2014). Next to rules being buried in programming code, Zoet (2014) argues that rules are also embedded in minds of employees, (parts of) procedures, manuals, schemes, and business processes. Furthermore, the increasing number of rules (among others due to increased legislation), the increased frequency in which they change, the different types of rules, the increased necessity of rule execution due to efficiency by automation measures, and the proof of consistent execution to third parties produce many challenges to organizations (Boyer & Mili, 2011; Graham, 2007).

Separating business logic from applications is in line with earlier work of (Boyer & Mili, 2011; Graham, 2007). The independent treatment of rules by organizations also implies a different approach than process and data management (Jörg Hohwiller, Schlegel, Grieser, & Hoekstra, 2011). While the mentioned domains share various characteristics, they also differ substantially on some characteristics (Zoet, 2014). When analysing the existing body of knowledge it is clear that application, business process and data practices are widely researched and adopted by the practice while the business rule-research field is rather young and immature (Von Halle, 2007). These differences can be communicated in a clearer manner by utilizing maturity levels, assigning them to individual research-fields, in order to compare them. According to Edmonson & McManus (2007), research field maturity can be categorized into nascent, intermediate, and mature. In recent work, Zoet (2014) stated that the maturity of BRM research from an IS perspective is in its nascent phase, relatively lower than the process and data research fields within the IS research domain. The nascent phase is the lowest level of maturity and is characterized by (Edmondson & Mcmanus, 2007):

1. **Little theory exists;**
2. **More open-ended research questions;**
3. **Focus on exploring the field and its underlying phenomena and relations while avoiding hypothesizing specific relations between variables;**
4. **Serves as an analytical journey towards exploring what themes are important for further research, acting like a filter mechanism for following studies.**

The separation of business logic from, for example, processes is demonstrated in an example based on the work of Hohwiller, Schlegel, Grieser, and Hoekstra (Hohwiller et al., 2011). Their work containing a discussion on why and how business processes, data, and business rules are separated concludes upon stating that these practices are good measures to increase efficiency and flexibility. However, it is important that these practices are seen independently from each other while still complementing each other. In Figure 6 a process is depicted visualizing a business process in which the business process and business rules practices are not separated.

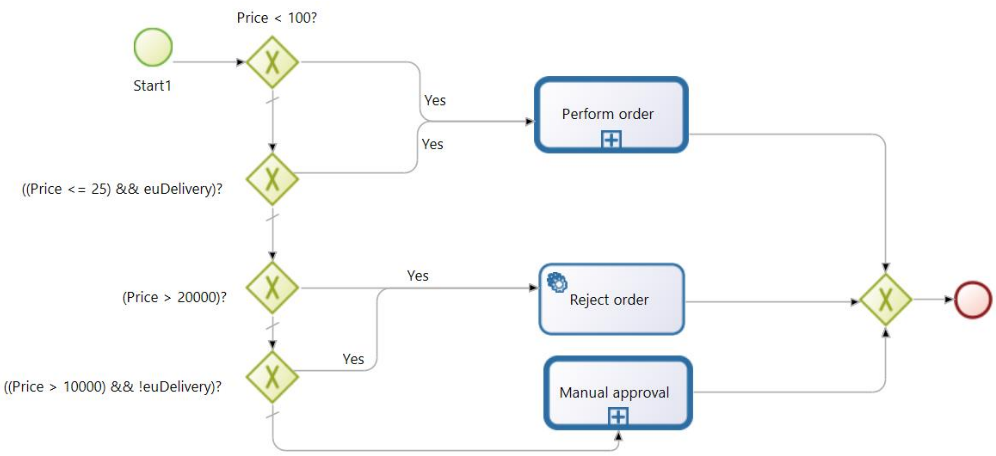


Figure 6 - A business process model intertwined with rule logic (Hohwiller et al., 2011)

The example depicts a business process concerning the decision on which customer orders are performed, rejected, or manually approved. At least half of the modelled content within the business process represents business logic which makes it a bad design and hard to maintain (Hohwiller et al., 2011).

3.1.1 Business rules

The concept of 'Business' in business rules represents that businesses, instead of IT, are in the lead when designing solutions, ensuring more business-IT alignment (Business Rules Group, 2003). However, the business perspective of the rules is often neglected or comes second place compared to information technology perspective (Kovacic, 2004). Morgan (2002) defined a business rule as:

> *"A statement that defines or constrains some aspect of the business intending to assert business structure or to control the behaviour of the business".*

It is evident that Morgan gave priority to the business perspective over the information technology perspective in this definition, in which the business is leading, and information technology plays a supportive role. In this thesis, we adhere to the definition given by Morgan. To instantiate the definition of a business rule as provided by Morgan (2002) the following examples are included:

1. *An order over $1,000 must not be accepted on credit without a credit check.*
2. *A customer is always considered a gold customer if the customer places more than 12 orders during a calendar year.*
3. *Delivery of products is not allowed before the customer has paid the bill.*

A more complex example of a business rule usually consists of multiple conditions and operators:

1. *If a bank account is over $7000 and more than 5 transactions have taken place in the last year, and the account holder's age is > 25 and is on the same job for more than 12 months, then accept the application, else reject the application.*

Literature suggest multiple types of business rules that are defined in theory and used in practice. For example, a well-applied theoretical overview of business rule types comes from Ross (2003), which separates constrainment type business rules, behavioural business rules, and definitional business rules. Furthermore, Zoet (2010, 2014) defined some additions to these types as Process, Actor, Data, Decision, and Event business rules. Taking into account all types of business rules available in literature and practice would not fit within the scope of our research project. Therefore we focus on decision type business rules only.

3.1.2 Business rules management

Business Rules Management (BRM) is the practice of managing business rules, usually centered around the elicitation, design, verification, validation, deployment, execution, audit, and/or monitoring of business rules (Zoet, 2014). These activities are commonly found in practice and are part of a process which is traversed to translate internal policies and external laws and regulations into products and/or services. Advantages of BRM are:

- It lowers the costs incurred in the modification of business logic due to the separate management of business rules;
- Shortens development time because of the reduction of time spent on rule requirements inside systems design;
- Due to the separate approach, business rules are easily shared among multiple (external) applications;
- Changes can be made faster and with less risk due to less dependencies as rules are not embedded in the system's design itself;
- A structured approach of documentation of business logic. Business logic is a type of knowledge which is not always documented appropriately, causing (vital) knowledge to go lost due to highly experienced employees leaving businesses (Morgan, 2002).

An important characteristic of BRM is that it stands for the elicitation of such business logic from employees in order to make it explicit and store it for further use. Further use can either be in the form of training new employees to apply the knowledge in their daily work or automation in which the business logic is digitized in order to be processed by machines (Zoet, 2014).

To be able to utilize business logic in an automated manner businesses have to transform their business logic into IT-language. Business language comprises the business logic (the '*specifications*'), while the IT language is represents events and data operations (the '*implementation*') which is interpreted by machines. Business language is generally equal to natural language and best describes how organizations operate and deal with business situations, like mentioned in the definition of Morgan (2002), asserting structure and/or control behaviour of the business and its operations. The '*implementation*' language, interpretable by IT systems logically is a more formal language, often putting heavy constraints on freedom of movement on the creation of business rules (Zoet, 2014). An example of the differences between business and IT statements and their transformation is provided in Table 12.

Table 12 - Example of business decision logic and IT decision logic

(r1) "If customer has spent more than 1000$ in the last year then customer is a bronze customer. (r2) "If the customer is a bronze customer he will get a discount of 5%."
(r1) discount (Customer, 5%) :- bronze(Customer). (r2) bronze (Customer) :- spending(Customer, Value, last year) , Value >1000.
Business fact: spending (Peter Miller, 1200, last year).

Research performed by Zoet (2014) provides insights on translation issues between business and IT, due to the transformation of natural language into functional language interpretable by IT systems. Zoet describes a process of business rule management wherein two transformations are made to support the creation and execution of business rules. First, business users translate their business logic into an implementation-independent business rule language to ensure universality of execution by the large amount of BRMS and rule engines currently available in the market. The second translation is performed by transforming the implementation-independent business rules into implementation-dependent business rules, thus specifically tailored to suit the linguistic requirements of the chosen BRMS or rule engine.

The benefits of utilizing business logic in an automated manner are best exemplified by the following situation: the Dutch Tax and Customs Administration. This organization used to employ resources to manually calculate and send tax administration data to citizens of the Netherlands. Nowadays, this is performed more efficient, where Dutch citizens utilize an online digital portal in which data is inserted. The services of the Dutch Tax and Customs Administration then automatically store and process the data to be calculated by systems. The systems then produce a given amount of tax return for each Dutch citizen, which is automatically printed and send via letter mail. As described in detail in the beginning of this chapter, business logic is often processed intertwined with other aspects of IS/IT design. We often observe that business logic is written in code, for example Java or C#. However, we argue that practitioners are not able to profit from the benefits of BRM as described earlier in this chapter.

3.1.3 Rule engines & Business Rules Management Systems (BRMS)

Utilizing business rules requires them to be written into specific formats not only for explicitation of knowledge and communication but also to automate them. Automation of business rules can be configured and performed in various ways, by utilizing Rule-engines or BRMS's. In general, business rules can be automated by 1) *hard-coding them into existing information systems,* 2) *utilizing a business rule engine,* and/or 3) *utilizing a BRMS* (Zoet, 2014). Due to scope constraints we do not focus on the definitions and differences between these variations.

Utilizing (automated) business rule execution usually offers the following functionalities (Primatek, 2009):

- Tools to allow a business user to create, update, or delete business rules in a business language (i.e. non-technical and for example English, using terminology of the business).
- Tools to help manage the rules (history of changes or versions, control on who can change which rules, rule life cycle, etc.).
- Tools to let applications use these business rules as executable pieces which allows the rules to be positioned for change.

As already identified in the extensive work of Zoet (2014) on methods and concepts for BRM, there are several unexplored opportunities for further research. In the next two sections we elaborate more upon the current knowledge gap in BRM and will continue with a state-of-the-art exploration of business rule architectures.

3.1.4 Knowledge gap in business rules management

The literature on business rules and business rules management presented in the previous sub-sections show an increasing maturity of the business rules domain for both researchers and practitioners. We identify that many researchers and practitioners focus on the technical aspect of BRM within the context of automated information systems and its architecture. This focus leads us to a gap of knowledge on logical architectures within the BRM domain. We seek to extend on the little amount of research which has been performed on business rules architectures with a focus on its logical design by van Thienen et al., (1993) and Zoet et al., (Zoet, Ravesteyn, & Versendaal, 2011). With the results of this study we try to set one step closer towards practitioners utilizing architecture just as it is used in other neighbouring domains, guiding planning and communication before changes are made to a product and/or service. This can potentially increase effectiveness of working with larger business rule sets utilized by many automated products and/or services.

3.1.5 State-of-the-art exploration of business rules architectures

While the business rules research field is currently getting increasingly popular amongst researchers and practice, the concept of a business rule architecture is relatively under-researched compared to other sub-topics in the business rule research area. The business rules research area is – as described earlier – still in its nascent phase of maturity. Analysis of the available literature on business rules in general and business rule architectures shows that business rule architectures are always placed in the context of a technical architecture. Technical architectures - especially in BRM-specific literature - often focus on the technical specifications and characteristics of software applications rather than the business logic defining the semantics representing the internal policies and/or external laws and regulations on which the product and/or service is built. The work of Boyer and Mili (2011) on agile business rule development is one example of dedicating work on architectures in the BRM research field, but rather on the technical design of business rule engines and business rule management systems. The same phenomena can be identified in the work of Paschke and Bichler (2008), Xiao and Greer (2009), Biletskiy and Ranganathan (2008), Manuel and AlGhamdi (2003), Mohagheg (Mohaghegh, 2005), Nammuni et al. (2004), O'Brien (2008), O' Brien et al. (2008), Curti et al. (2005), and Pons (2003).

As far as the authors are aware, the only method of modelling business logic, and therefore business rules, into a logical architecture is The Decision Model (TDM) created by Von Halle and Goldberg (2009). TDM is a notation that is platform and technology independent (OpenRules®, 2014) which features a relatively simple set of symbols to represent business logic. The notation is used by a number of Dutch governmental institutions, i.e. the Dutch Tax and Administration office and the Sociale Verzekeringsbank, as well as other national and international organizations. The advantage of TDM is that, due to its (perceived) simplicity, business users can model knowledge that can be interpreted and executed by several widely used BRM-systems. Examples of this are: OpenRules (2014), Blueriq (2015; 2013), and Drools (2015).

Since there is no unambiguous and uniform definition of a Business Rule Architecture we aim to construct and propose a definition for a Business Rule Architecture ourselves. We propose and adhere to the following definition of a Business Rule Architecture:

"A Business Rule Architecture is a formal description of a cohesive set of business rules and their relationships to provide knowledge about its structure, dependencies, and design principles."

In the next paragraph we consider neighbouring fields to see if certain concepts are common for these fields and that of BRM. Available knowledge on logical architectures within neighbouring fields will serve as a theoretical foundation for the concept of BRA's.

3.1.6 Neighbouring fields on architectures in IT

Neighbouring fields for BRM on logical architectures are the fields of Enterprise Architecture, Software Architecture, and Data/data warehouse architectures. Literature analysis reveals that there are other fields that focus on architectures as well, but our scope includes only the most popular and largest adopted architectural fields utilized by researchers and especially practitioners as it guards the practical relevance of the included literature. In these subsections we provide an overview of their definitions and their main concepts. Furthermore, knowledge gained from this literature overview on neighbouring domains provides us with the means to select on which field shows the most similarity which will be further analysed for research on a modifiable business rule architecture.

3.1.6.1 Enterprise Architecture

Enterprise architecture (EA) is concerned about the guidance of organizations towards higher maturity levels, taking into account business administration, information science, and computer science. Other than the software architecture and data architecture domains, EA focuses on the organizational view, including relationships between domains which support the organization's processes. The notion of architecture is increasingly applied in non-technical domains. Enterprise architecture is a product of the emerging enterprise engineering domain (Jonkers et al., 2006). Enterprises are confronted with an increasing variety of new technologies, business models, and other potentially disruptive options. Such developments results in enterprises having have a strong need towards innovation to adapt quickly to the ever changing environment in an attempt to create new business opportunities. This trend demands enterprises to act quickly which is dependent on the management's decision making. Enterprise architecture promises to provide organizations with insight and overview to harness complexity, while potentially offering blueprints for solutions in a coherent an integral fashion. EA offers a medium to achieve a shared understanding and conceptualization among all stakeholders involved (Ross, Weill, & Robertson, 2007). EA mainly thanks its creation and foundation due to the fact that practitioners became aware that the development and evolution of Information Technology should be done in conjunction with the development in the context in which it was used around the 1990's (Zachman, 1997). This insight led to the identification of the business/IT alignment problem. A popular description with a corresponding definition comes from Lankhost (2009), which states that EA views enterprises as a whole as purposefully designed systems that can adapted by the use of architectures. Within that context, Lankhorst (2009) defines EA as follows:

> *"Enterprise architecture is a coherent whole of principles, methods and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems and infrastructure."*

Within the context of the concept of architectures in general we identify important pillars within this definition. First, *a coherent whole of principles, methods, and models*, is mentioned, defining that enterprise architectures, thus architectures, must be coherent sets of objects. Furthermore the aspect of '*design*' is included, which refers to the utilization of architectures to plan and guide realization afterwards. An EA-specific notion are the aspects of '*organizational structure, business processes, information systems, and infrastructure*'. The first two aspects, organizational structure and business processes are not technical-oriented but rater business-oriented, while the last two mentioned aspects, information systems and infrastructure are IT-oriented. However, this totally depends on the instantiation of views utilized to express these aspects at an organization. Blenko, Mankins, and Rogers

(2010) argue that business logic containing decisions as a separate view is often ignored or put away in other aspects of EA. They state that organizations, that are willing to utilize their decisions to in pursuit of better organizational performance, should take into account business logic as a separate aspect to manage the organization's effectiveness and efficiency. Based on this, we believe a logical architecture containing business logic could be taken into account next to other architectural aspects that are used to represent organizations in the EA practice.

3.1.6.2 Software Architecture

Since late 1960s and 1970s researchers were observing an increase in software size. To be able to cope with this, Randell (1979) coined the concepts of decomposition and modularization in software planning. Later on, in 1996, Shaw and Garlan (1996) identified several more issues concerning the design of the ever increasing complexity of software systems. They state *"… the design problem has gone beyond algorithms, and data structures of computation"* In here, these issues are defined as part of software architecture design. In the beginning of the 1990s software architecture got larger adoption and attention in the software engineering community and also drew the attention of an increasing amount of researchers. Back in 2002, Bengtsson (2002) stated that software architecture has become an accepted concept, also noticeable by the adoption of the role of software architect by many organizations.

Literature on Software Architecture contains multiple, but similar definitions of the same concept. A popular definition is proposed by Bass, Clements (2012), that define the concept of Software Architecture as follows:

> "*The software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both*."

To simplify communication regarding SA, scoping should be applied to determine which particular structure or structures we are working with. Architectures can exist out of multiple structures, with multiple viewpoints for different stakeholders. The notion of a viewpoint can be described as a specific representation of a part of the architecture in which the given stakeholder(s) are interested. For example, when boardmembers need to make decisions concerning their products and/or services, they do not want to be presented with a very detailed architectural view including all interfaces, connectors, modules, functions, annotations, and components. In this case, a rather high-level architecture probably has sufficient granularity to support strategic decision making by boardmembers. However, software engineers building the same product and/or service will need more detail to support their operations regarding the development and evolution of that product and/or service. As business logic is often a key element in software engineering, we performed literature analysis on views for logical architectures containing business logic separated in the software engineering domain. However, this did not result in any relevant contributions. Similarly as in the EA domain, we believe that including views that separate and represent business logic within software architecture development is important as it is in line with the theory of separation of concerns (Dijkstra, 1974; Ossher & Tarr, 2001; Tarr et al., 1999) as described earlier.

3.1.6.3 Utilization of the domain of data architectures

Literature on logical architectures addressed in the previous sections contains numerous works from either the enterprise and software architecture domains. However, little literature suggest whether business logic or business rules are part of any of these and should be addressed or incorporated in the logical design of information systems. Little research is performed on business rule architectures and business rule structuring. Literature research identified earlier theories proposed by van Thienen (2001), TDM (2009), and Nijssen & Le Cat (2010) which suggest that the rule-oriented approach is most similar to the theoretical underpinnings of database models and database architectures. This is also further developed and demonstrated in the work of Zoet (2011). Analysis of the TDM approach (Von Halle & Goldberg, 2009) in combination with the work of VanThienen (1993) shows that both are based on relational algebra, where this is the same for Inmon's data warehouse concept in the data warehouse domain. Furthermore, analysis of the CogNIAM approach as proposed by Nijssen & Le Cat (2010) shows its similarities with Kimball's dimensional modeling utilized in the data warehouse domain, as both are fact-oriented. Next to the earlier mentioned two schools which show similar theoretical underpinnings between the data warehouse and BRM domains we identified Linstedt's data vault modeling as a data warehouse architecture that is not researched an does not have an equivalent from the BRM domain in terms of theoretical underpinnings.

Because of the similarities between data(warehouse) architectures and rule architectures the theories presented by VanThienen (1993), TDM (2009), Nijssen & Le Cat (2010), and Zoet (2011) are taken as appropriate reference to proceed with the database / data warehouse domain. Similarly as the previous two sections the next section contains a short overview of the database and data warehouse domain.

3.1.6.4 Database and data warehouse architectures

The third domain included in which architecture is a central means to express a design is the data architecture domain. Authors argue that data architectures are a sub-component of EA (Mario, 2010; Zachman, 1997). However, we interpret them as independent architectures due to the maturity of this domain and its available body of knowledge with maintainability and modifiability. Databases and database theory have been around a long time, see also the work of Inmon (2005). Early mentions of databases all focused on the utilization of single databases serving all known purposes of information processing at organizations. In most cases, the primary focus of the early database systems was operational, also referred as transactional, processing. In recent decennia, more advanced notions of database use has emerged (Kimball & Ross, 2013). Nowadays, we can recognize two fundamental database instantiations, 1) *Operational Databases - ODB* and 2) *Analytical Databases – ADB* (Inmon, 2005). Inmon (2005) argues that the distinction between ODB's and ADB's occurs due to two reasons: 1) the data serving operational needs is different than the data utilized for analytical needs. Data utilized for operational needs are single records that are processed in transactions, while data utilized for analytical needs comprises aggregated data from operational data to support decision making, and 2) the supporting technology for operational processing is different from the technology used for analytical processing.

Literature shows that the concept of an ADB is very similar to the concept of a BRA. Both concepts are utilized to support decision making, processing and deriving data to be able to do so. Instantiations of ADB's will be elaborated upon in the next section.

Data warehouse

There is little debate about the usefulness of structuring data in the form of data warehouses to support decision making (NESMA, 2012). A data warehouse is defined by Chaudhuri & Dayal (Chaudhuri & Dayal, 1997) as:

> "*A collection of decision support technologies, aimed at enabling the knowledge worker (executive, manager, analyst) to make better and faster decisions*"

We believe this definition is rather broad and focuses entirely on the knowledge worker to make decisions in a purely business context. Therefore we include another definition which has a more technical-oriented view on a data warehouse proposed by Widom (1995):

> "*A data warehouse encompasses architectures, algorithms, and technologies for bringing together selected data from multiple databases or other data sources into a single repository, suitable for direct querying or analysis*"

A data warehouse is built from two designs: the logical and physical design. The logical design is more a conceptual design and more abstract than the physical design. In the logical design the view is set at the logical relationships among the objects. In the physical design the view is set at an effective way of storing and retrieving the objects as well as handling from a transport and recovery perspective (Golfarelli & Rizzi, 1998).

Furthermore Widom (1995) identifies and elaborates on the architectural challenges data warehouses face. Information from each source that may be of interest is extracted in advance, translated, filtered as appropriate, merged with relevant information from other sources, and stored in a centralized repository. To be able to function as described by Windom, a data warehouse relies on different technologies enabling it to process data from very different technological platforms and standards and therefore must be technology and platform-independent

In general, utilizing a data warehouse could provide organizations with opportunities over 'regular' management of data (DBS's) and reporting practices to support decision making. For example, Kimball (2013) describes a set of general advantages of using a data warehouse as a supplement to the use of ODB's:

- Aggregate data from multiple data sources into a single database so single query engine can be used to present data in a uniform method;
- To ensure that information becomes more easily accessible, it can act as the authoritative and trustworthy foundation for decision making;

- Data from multiple sources is transformed into information and presented in a consistent way;
- Change requirements on data warehouses can be and are usually faster processed since it depends on ODB's but is not part of it.
- Maintain historical data also called data history, while not all source systems do so, or are able to.

Within the domain of data warehouses, three widely-used well-known schools exist: 1) the utilization of a *relational model adhering to the normalization principles* proposed by Inmon (2005), 2) the utilization of *star-and snowflake-schemes, focusing on dimensions of data* proposed by Kimball (2002), and 3) the utilization of *the relational data vault principle, based on hubs, satellites, and links* proposed by Linstedt (2009).

3.2 Exploring rule-oriented architectures

In this section the three data warehouse-architectures: Inmon's normalized data warehouses, Kimball's dimensional data warehouses, and Linstedt's data vaults and their underlying architectural structures are described and explored for applicability in the BRM domain.

To be able to evaluate the architectural structure of all three data warehouse architectures with underlying concepts we need to explore how the data-oriented architectures can be applied in a rule-oriented manner. For this we believe that metamodeling of the structures provide us the necessary handholds to support the exploration process. To ground this process we used the method engineering Process Deliverable Diagram-notation as proposed by Brinkkemper (1999; 1993). This notation consists of two components: 1) *the process flow* (containing activities) and 2) *the deliverable flow* (containing the deliverables as concepts). Since we are only focusing on the concepts of the data warehouse theories we choose to omit the processes needed to come to the included data warehouse architectures.

For the exploration of the data-oriented architectures, several common concepts are identified and utilized. In the BRM domain the concepts of: a decision, a conclusion, a condition, a fact, a fact value, and an operator are commonly utilized (Zoet, 2014). In Table 13 we elaborate upon these concepts more in-depth. Utilized concepts which are not shared across the architectural candidates are defined in their individual subsections.

Table 13 - Shared concepts and definitions (Ross, 2003; Zoet, 2014)

Concept	**Elaboration**
Decision	A conclusion which is answered based on the evaluation of conditions resulting in new knowledge or an action. For example, Accept applicant concerning the application for child benefits.
Conclusion	Represents a fact that is derived and concluded for one or multiple conditions. In other words, it is a represents a fact that is derived from other facts. For example, one of the conclusions that contributes to whether the applicant will be accepted or not will be Determine income from employment, which is dependent on the conditions 1) taxable income from other activities, 2) taxable periodic benefits and allowances, and 3) taxable notional rental value.

Condition	Represents a fact and is used in the evaluation to assert a conclusion. For example, 1) taxable income from other activities, 2) taxable periodic benefits and allowances, and 3) taxable notional rental value.
Fact	Represents a concept of data from the real world, an event or circumstance that has actually occurred. A fact can either be represented by a condition or a conclusion. For example we take the fact Employment income.
Fact value	The domain value of its parent fact. For example, for the fact Employment income we have the domain specific fact values: 1) <10000, 2) >30000, 3) >50000, etc.
Operator	An operator is a symbol that is utilized to evaluate a condition against another condition or another fact value. For example, an operator that has to test whether the applicant of the child benefit application meets the restrictions to be able to claim child benefits at all. The operator is utilized to evaluate if the Fact Employment income is lower than € 10.000,00, then continue the selection process, else, reject the applicant.

3.2.1 Inmon's normalized data warehouse approach

The data warehouse concept was first coined by Inmon in 1992, and further specified in later editions of his book (2005). His work defines the designed structure as a 'Data Warehouse'. The data warehouse concept as defined by Inmon is based on relational theory (Zoet, Ravesteyn, & Versendaal, 2011). Database normalization is designed as a technique to organize databases in a certain structure called normal forms, minimizing data redundancy while improving storage space resource utilization (Codd, 1970). Although five normal forms exist, the fourth and fifth normal forms are rarely used. The same can be observed in the data warehouse structure, solely utilizing normal form one, two, and three. A simple example of the database normalization technique can be found in (NH Computing, 2008). Inmon (2005) defines the data warehouse with four characteristics:

- Subject-oriented
  - The data in the database is organized so that all the data elements relating to the same real-world event or object are linked together;
- Time-variant
  - The changes to the data in the database are tracked and recorded so that reports can be produced showing changes over time;
- Non-volatile
  - Data in the database is never over-written or deleted - once committed, the data is static, read-only, but retained for future reporting; and
- Integrated
  - The database contains data from most or all of an organization's operational applications, and that this data is made consistent.

The Decision Model – A rule family-oriented BRA

The data warehouse architecture as defined by Inmon is similar to the architectural structure utilized by The Decision Model (TDM), as both architectures are based on relational theory (Zoet et al., 2011).

Since Inmon does not imply the use of a specific notation we selected TDM as the notation to utilize as a first architectural candidate, which complies with the design principles of Inmon's data warehouse concept, but tailored to the BRM domain as demonstrated in the work of Zoet et al., (Zoet et al., 2011). To compare the architectural structures and their design principles we present the meta-model of TDM in Figure 7. The TDM notation is further specified in Appendix C.

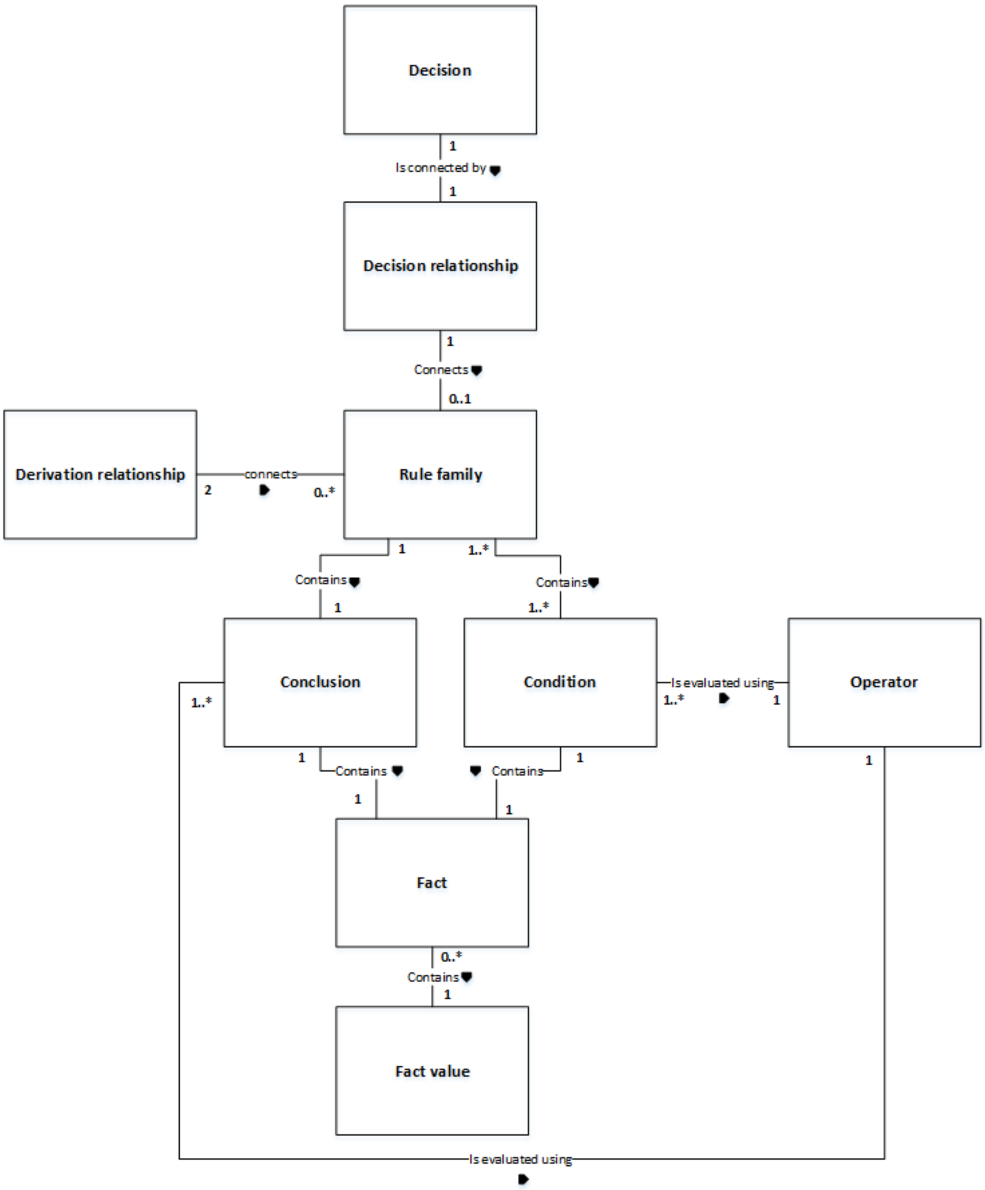


Figure 7 – Meta-model of the rule-oriented approach of the TDM normalized BRA

TDM utilizes the normalized design principle which is enforced by only allowing one conclusion to be part of a single rule family. The decision is always connected using a decision relationship to the highest rule family in its underlying rule family hierarchy. A rule family contains one or multiple conditions which can be derived from another rule family, which is connected by a derivation relationship, or non-derived, meaning the condition is atomic. Furthermore, all conditions and conclusions represent a business fact which can be instantiated with fact values. Normalization within TDM is applied using the first, second and third normal form. The first normal form enables the business logic to be represented and interpreted in one way possible. The second normal form eliminates redundancies in the business logic. The third normal form ensures that there is no hidden logic within the business logic as part of a conclusion (Von Halle & Goldberg, 2009).

3.2.2 Kimball's dimensional modeling approach

In reaction to the data warehouse concept presented by Inmon in 1992, Kimball (2002) tried to revolutionize the domain of data architectures with the notion of Dimensional modeling. In his work, a data warehouse is defined as:

> "*A data warehouse is a system that extracts, cleans, conforms, and delivers source data into a dimensional data store and then supports and implements querying and analysis for the purpose of decision making*"

Dimensional modeling is a logical design technique that seeks to present the data in a standard framework that allows for high-performance access oriented around understandability and performance. Dimensional modeling adheres to a discipline that uses the relational model with some important restrictions. Each dimensional model is composed of one table with a multipart key, defined as a fact table, and a set of smaller tables defined as dimension tables. The dimensional model is based on a star form and therefore utilizes the so called star-like schema with the dimensions surrounding the fact table (Kimball & Ross, 2013) Further normalization as defined by Inmon (2005) is possible and creates the so called snowflake scheme, but should be avoided to prevent performance loss (Kimball & Ross, 2013). This particular structural design decision makes it very different from the normalized data warehouse approach as proposed by Inmon since redundant data dimensions are very usual in the design of Kimball's logical design. Furthermore, the logical design also comprises the use of measures within the fact table entity. Measures are entities that perform certain operations over the fact data in the same fact table. Measures are static and consistent while analysts are using them to inform their decisions.

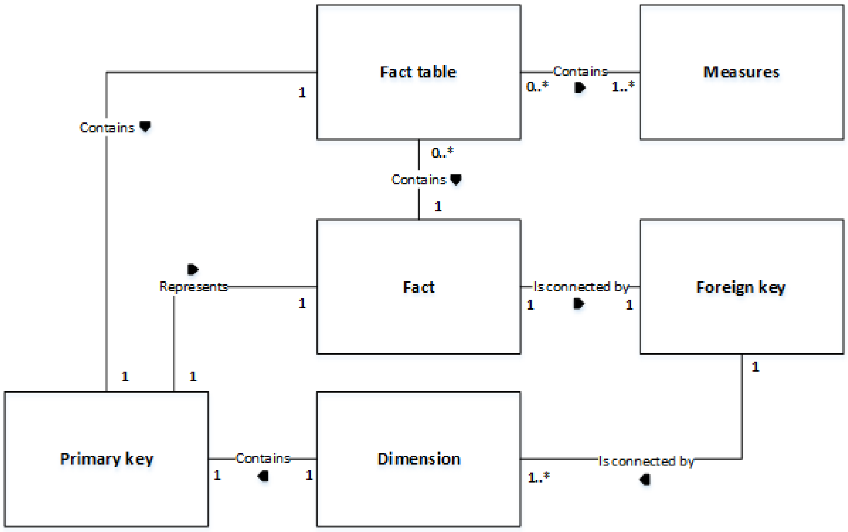


Figure 8 – Meta-model of the data-oriented approach of Kimball's dimensional modeling

To be able to transform Kimball's dimensional modeling approach, which is data-oriented, into a rule-oriented approach we modelled and presented it's meta-model in Figure 8. Based on this meta-model we constructed a rule-oriented meta-model as presented in Figure 9.

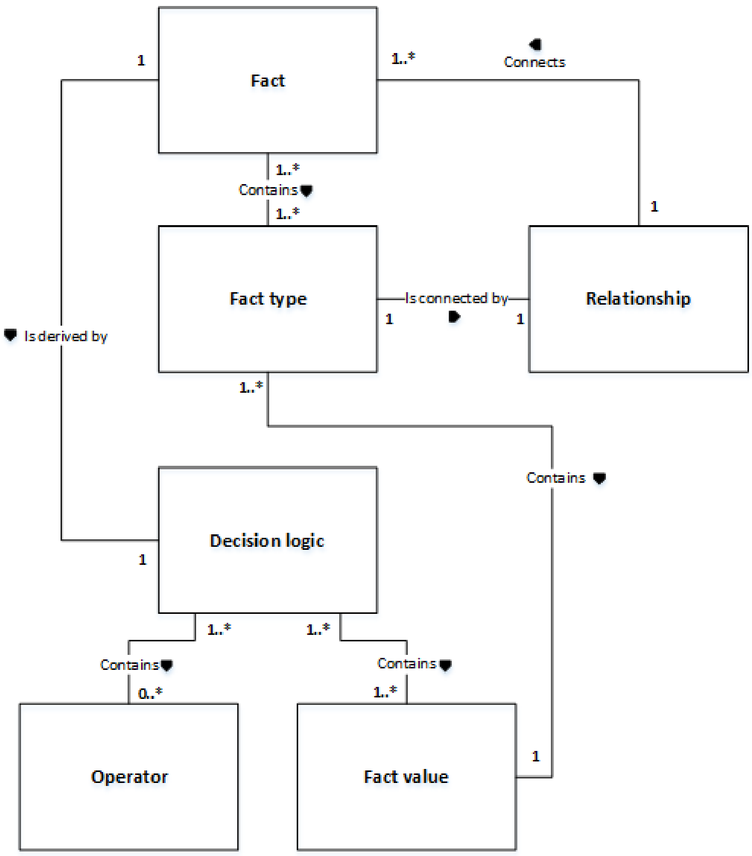


Figure 9 - Meta-model of the rule-oriented approach of Kimball's dimensional modeling

Kimball's dimensional modeling approach utilizes facts as core elements of the architecture, and can therefore best be categorized as a fact-oriented architecture. For the application in the BRM domain we see similarities with the Cognition enhanced Natural language Information Analysis Method (CogNIAM) approach (Nijssen & Le Cat, 2010) as this method utilizes a fact-oriented approach to manage business rules as well. Due to this, CogNIAM (Nijssen & Le Cat, 2010) is utilized to ground the application of Kimball's data-oriented approach in this research. The exploration of the data-oriented approach of Kimball's dimensional modeling for application in the BRM domain shows some differences between both domains. First, the use of tables is omitted since this is a typical data-oriented concept. If we look at the function of the original fact table design we see that it contributes to an information request from the business. Therefore we propose to change the fact table to a fact as also adhered to in CogNIAM. Furthermore, a fact is made up from one or multiple fact types, which is common to represent other facts in the BRM domain (Zoet, 2014). A fact is represented by a conclusion, while the fact types are the secondary facts (conditions) utilized to answer the conclusion for one decision. Fact types that answer a given fact in one decision are connected with their parent fact, and their underlying fact types, utilizing a relationship which is common in both the data and rule-oriented approaches. The derived fact within the same decision is represented by a dimension. Similarly as applied by CogNIAM, a fact is always derived by decision logic. Decision logic generally contains the logic to determine the conclusion, which includes fact values of the fact, its underlying fact types and their fact values, and operators.

3.2.3 Linstedt's data vault modeling approach

Data Vault modeling is a logical design technique mainly used in data warehousing. It was proposed by Linstedt (2009) and is specifically designed for storing data from different sources, with different definitions and reliabilities. Linstedt (2009) defines a data vault as follows:

> *"The Data Vault is a detail oriented, historical tracking and uniquely linked set of normalized tables that support one or more functional areas of business."*

As Linstedt positions its data vault modeling approach as a hybrid variant of both Inmon's and Kimball's data warehouse modeling approaches it is characterized by elements of both approaches introduced earlier in this thesis. The data vault consists of three core components, a hub, a link, and a satellite. Also, the notion of a reference table exists within the data vault, which is used when data is referenced in high amounts, to reduce redundant storage of data. A hub is a table containing a unique list of business keys, which form the core concepts of the information request. Furthermore, a hub contains surrogate keys that represent a relationship with a link, a load data time stamp, and a record source. The link is utilized to connect hubs with other hubs, or with shared satellites, which ensures traceability. The link stores the surrogate keys from the hubs it connects combined with date stamps and record sources. The hubs and links define the structure of the model, but lack any temporal attributes and descriptive attributes that define the detail of the (business) context regarding the hubs. This information is stored in separate tables defined as satellites. A satellite holds the contextual information separate from the hub due to different frequencies in which attributes change. A hub can be linked to multiple satellites as they can only provide contextual information for a specific level of information aggregation (Linstedt et al., 2009).

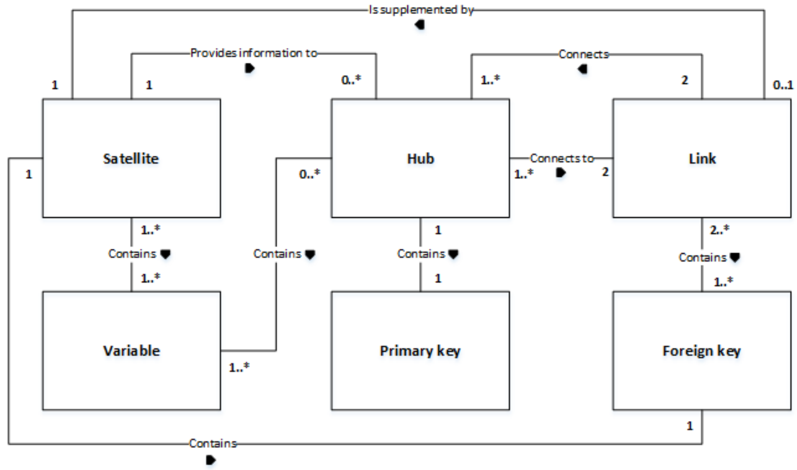


Figure 10 - Meta-model of the data-oriented approach of Linstedt's data vault modeling

To be able to transform Linstedt's data vault modeling approach, which is data-oriented, into a rule-oriented approach we modelled and presented it's meta-model in Figure 10. Based on this meta-model we constructed a rule-oriented meta-model as presented in Figure 11.

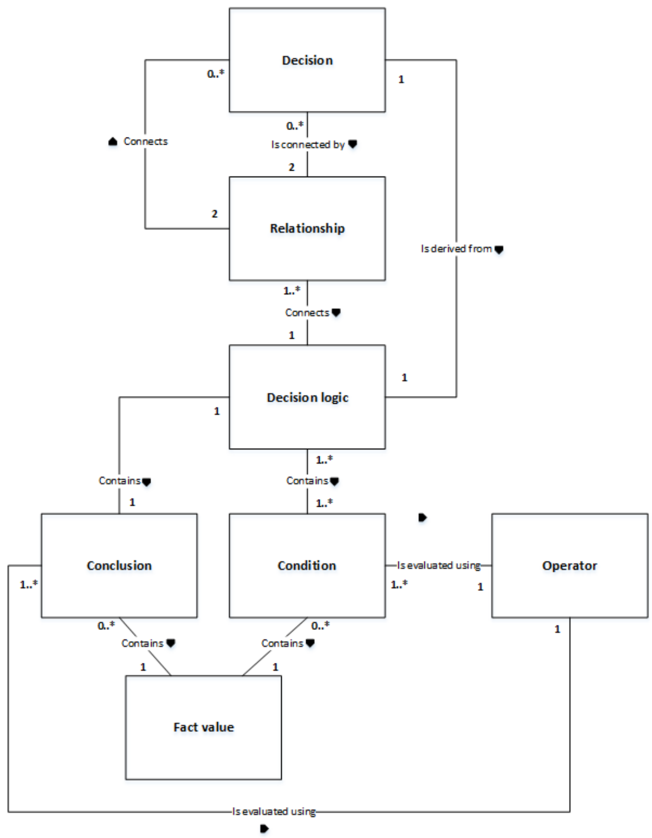


Figure 11 - Meta-model of the rule-oriented approach of Linstedt's data vault modeling

Contrary to the latter two architectural candidates presented we found no standard in the BRM domain that shows similarities with Linstedt's data vault modeling. Therefore we are forced to transform the data-oriented approach into a rule-oriented approach to be applied in the BRM domain. For this we decided to adopt certain element labels from the previous mentioned rule family-oriented and fact-oriented architectures. The rule-oriented architecture based on Linstedt's data vault modeling is decision-oriented as decisions are the main building blocks for its architecture. For the transformation of Linstedt's data vault modeling from a data-oriented approach to a rule-oriented approach several changes were applied. Similar to the exploration of the fact-oriented approach we retained the architectural structure, but made changes to match a rule-oriented approach. A hub represents a decision which is connected via a relationship to a link. The link is utilized similarly as we use it to express a relationship between two decisions to create a hierarchy between decisions. Furthermore, the satellite is also applied similarly, but instead of only providing contextual data we apply the satellite to provide the decision with decision logic, separating the decision logic from the decision itself. The decision logic contains the conclusion, the condition(s), and the operator(s) needed to process the decision. Both the conclusion and conditions represent business facts that contain fact values.

3.3 Quality attributes

Quality attributes have significant influence on the architectural design and design decisions taken by organizations (Klein et al., 1999). Furthermore, quality attributes are closely related to requirements management, where they are part of non-functional requirements concerning a certain system. Within the light of non-functional requirements, quality attributes capture many facets on *how* the functional requirements are achieved. In general, all but the most trivial systems will be characterized by non-functional requirements that are usually expressed in terms of quality attribute requirements.

For this study we adopt the widely used and cited work of Bass, Clements, and Kazman (2012) concerning quality attributes as proposed in the following definition:

"*A quality attribute (QA) is a measurable or testable property of a system that is used to indicate how well the system satisfies the needs of its stakeholders*".

Many different quality attributes are defined by researchers and practitioners in the past decades. One of the earliest quality model frameworks was suggested by McCall, Richards, and Walters (1977), containing a total of eleven quality attributes. This inspired the international community to develop a standard framework for software-quality measurement, the ISO 9126 standard. The ISO 9126 standard was further developed by researchers and practitioners into the ISO/IEC 25010:2011 standard which replaced the ISO 9126 standard completely in 2011 (ISO, 2011), as depicted in Figure 12. In this research, the ISO/IEC 25010:2011 standard with its corresponding quality attribute ontology is used as a central framework. This decision is made due to the high maturity of the ISO and IEC development and reporting on software quality knowledge. Furthermore, a lot of researchers and practitioners refer to the quality attribute ontology in the ISO/IEC 25010:2011 standard, proving its usefulness for both the academic and business domains.

How an architecture takes shape is very dependent on the requirements it has to deal with. With respect to the requirements engineering domain we do not elaborate on type of requirements and requirements management before considering architectures. In the context of this research we solely focus on quality requirements. However, we want to elaborate shortly on how requirements affect architecture design in such a way that the software product should have a certain achievement in relevant quality attributes. Another advantage of defining this relationship is the creation of operational definitions and measurements to determine to what extent a software product achieved a given quality attribute. Bass et al., (2012) states that a quality attribute requirement should be unambiguous and testable. To be able to do so, quality attribute scenarios are utilized. A quality attribute scenario commonly consists out of six characteristics which need to be defined: 1) *the stimulus*, 2) *the source of the stimulus*, 3) *the response*, 4) *the response measure*, 5) *the environment*, and 6) *the artefact*.

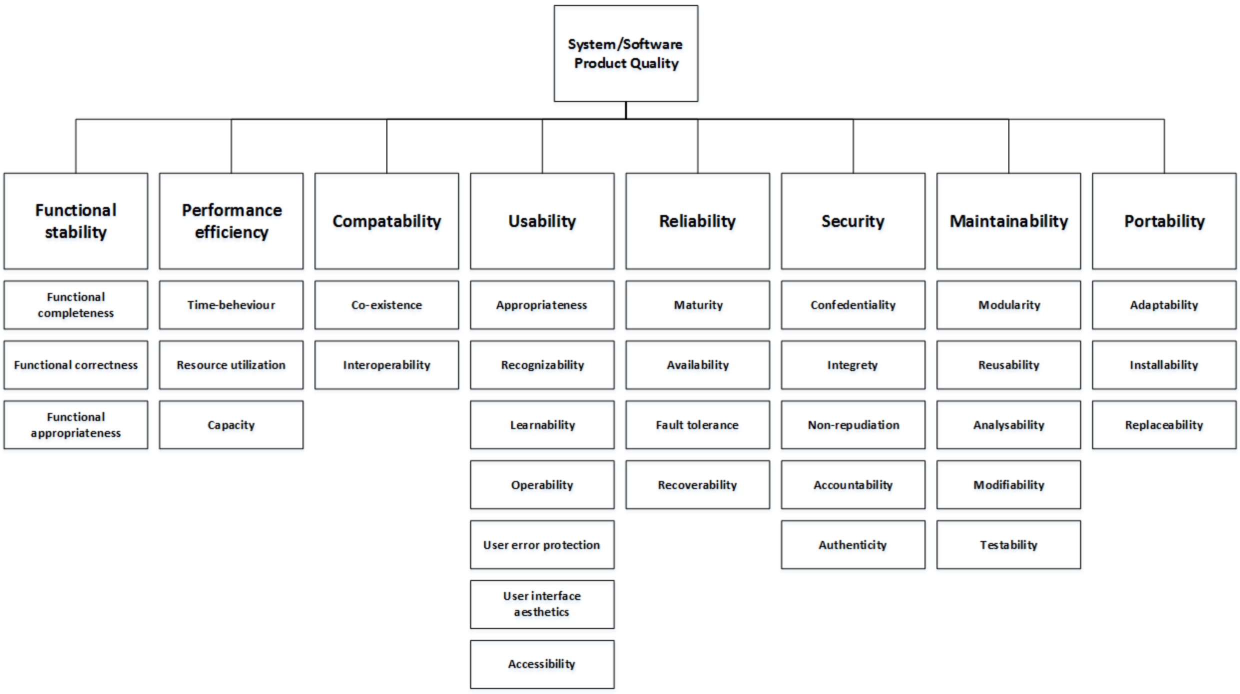


Figure 12 - The ISO/IEC FCD 25010 system/software quality standard (ISO, 2011)

Achieving quality attributes

To realize the business goals of the organization, the architecture has to achieve certain quality attributes based on quality requirements stated by involved stakeholders. To achieve a quality attribute, architectural tactics can be utilized. In other words, a tactic is a design decision that influences the achievement of a quality attribute response. The focus of a tactic is on a single quality attribute scenario, also defined as a quality attribute response by Bass et al., (2012), which is illustrated in Figure 13.

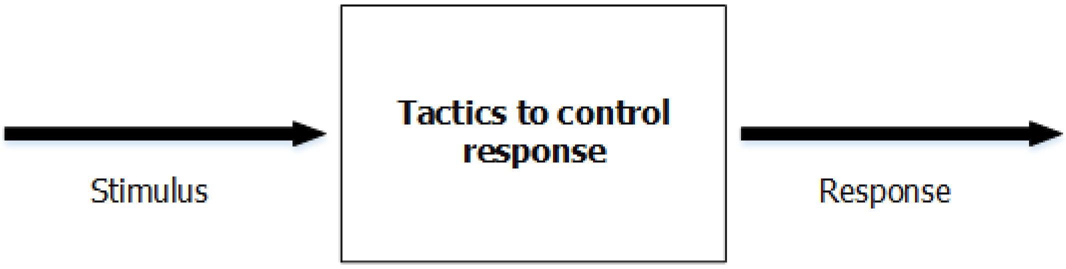


Figure 13 - The role of tactics to influence and control the response to a certain stimuli (Bass et al., 2012)

Tactics as design decisions can be categorized into the seven design decisions categories: 1) *Allocation of responsibilities*, 2) *Coordination models*, 3) *Data models*, 4) *Management of resources*, 5) *Mapping among architectural elements*, 6) *Binding time decisions*, and 7) *Choice of technology*. Examples of tactics are: Active redundancy (availability), Service discovery (interoperability), Encapsulation (modifiability), and Prioritization of events (performance) (Bass et al., 2012). The BRM domain shows a similar method of applying design decisions to achieve certain quality attributes. For example, the rule family-oriented architecture applies normalization to achieve a certain level of modifiability, while the fact-oriented architecture does not apply this design decision, utilizing more redundant data to ensure a higher level of performance. Therefore it is important, both for SA's as BRA's that design decisions are made taking into account all desired or needed quality attributes. This is referred to as a trade-off between quality attributes, often mentioned in literature on software architectures. However, for the BRM domain little is known regarding quality attributes and their trade-offs.

A trade-off relationship between quality attributes

Within complex software systems, quality attributes can never be achieved in isolation. As hinted in the previous section, the achievement of one particular quality attribute will have an effect on other quality attributes concerning the same system. This effect can either be positive or negative. For example, we take the widely used phenomena of different vessels of war built from the 17th trough to the mid-19th century. In this particular period, several types of (war)ships were built to serve different purposes. Take for example the *barque*, *cutter*, *frigate*, *sloop,* and *Ship-of-the-line* classes. Each of those classes are built on the format of a different structure, adhering to the achievement of different quality attributes. The *cutter* class is built to serve as, for example, coastal patrol, escort of larger ships, and small raids. Several structures can be identified in the architecture of the cutter class. 1) *The ship is small to medium sized*, 2) *The ship is lightweight*, 3) *the ship is single-masted*. Incorporating these structures into the architectural design of the *cutter* class has several advantages and disadvantages. Due to the combination of small size, lightweight properties of used materials, and placement of the mast utilized in the architecture design, the *cutter* class features the advantages of being very fast and agile, over the cost of less and lighter guns and lower damage tolerance. It is obvious that the largest ship class, the *ship-of-the-lines*, are characterized by the opposite of advantages and disadvantages. These advantages and disadvantages are simple ways to demonstrate the quality attributes of a ship class. Advantages like speed and manoeuvrability are achieved perks of the performance quality attribute, while the higher damage tolerance are achieved perks of the durability and/or resilience quality attributes. While there are ship classes which focus on performing average on all quality attributes (sometimes also referred to as an all-rounder), these are simply outclassed by extremes of smaller ships and bigger ships in most cases. Trying to achieve the highest of perks in all quality attributes when building a ship is practically impossible. Therefore quality attributes are to be placed in a trade-off relationship. This phenomena for in the context of software architectures is illustrated in an example provided by the SEI (Barbacci, Klein, Longstaff, & Weinstock, 1995) in Figure 14.

Returning to the software engineering domain, authors (Boehm & In, 1996; Breivold, Crnkovic, & Eriksson, 2008) state that a trade-off relationship is caused by conflicting concerns of stakeholders. For example, the organization that maintains the software product is primarily concerned with evolvability, maintainability, modifiability, and portability, while developers are more concerned with reusability and learnability (Boehm & In, 1996). Similarly, for the BRM domain these trade-offs are important for the construction of logical architectures. Depending on the desired traceability, volume, variety, and velocity of decisions that have to be processed the need for different quality attributes exist.

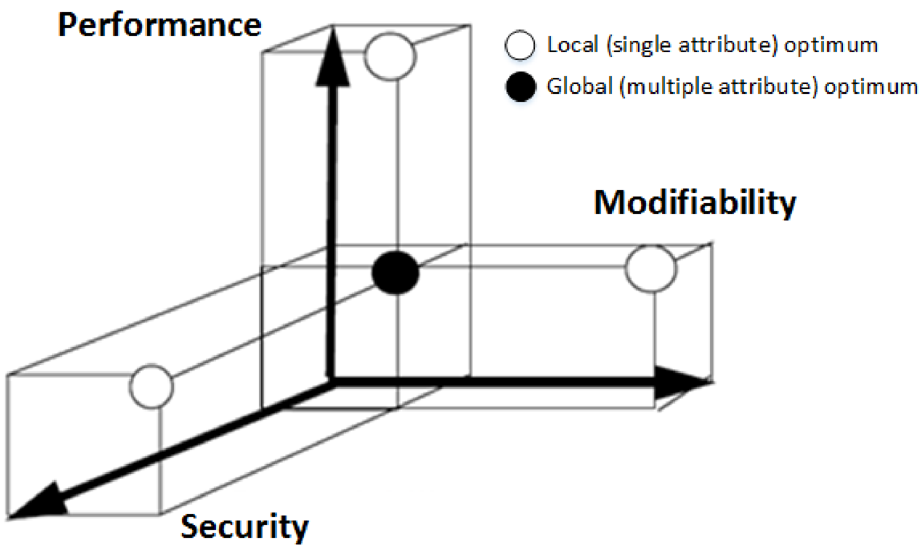


Figure 14 - Software quality attribute trade-off visualization based on the work of (Barbacci et al., 1995)

3.3.1 Evolvability, Maintainability, and modifiability

As described in our problem statement we focus on the concept of modifiability concerning logical architectures to be applied in the BRM domain. For this particular section we take the software architecture research field as the main source of inspiration due to the high maturity level of research concerning quality attributes, thus also on the concept of modifiability.

Authors agree on one phenomena constantly happening in the world of IS/IT, '*change*'. Change as part of evolution is necessary and inevitable, as products and/or services are influenced by (Rowe, Leaney, & Lowe, 1998):

1) **The ever-changing operating environment;**
2) **Changes in implementation technology;**
3) **Stakeholder needs which are either functional or quality requirements.**

In this section we elaborate upon the concepts of evolvability, maintainability, and modifiability, three quality attributes closely related to change.

Most literature suggest that Modifiability is a quality attribute which is part of the maintainability quality attribute, amongst modularity, reusability, analysability, and testability (ISO, 2011). A contradicting view on this is provided by Lundberg et al., (1999), who identifies maintainability and configurability to be part of the modifiability quality attributes. Another proposition of modifiability is presented by Andersson (2002), which states that both concepts are used interchangeably to denote the system's ability to be modified. Furthermore, a very similar concept is that of software or system evolvability. Similarly as the confusion in terminology between maintainability and modifiability as described earlier, theory shows that the concepts of evolution and maintenance are used interchangeably as well (Weiderman, Bergey, Smith, & Tilley, 1997). All three concepts concern how and to what extent an artefact is affected by change. According to Bass, Clements, and Kazman, (2012), changes to software products and/or services occur for various reasons, in different forms. For example, to add, change or retire features, fix defects, improve performance, enhance the user's experience, etc. The concept of evolvability is widely studied in the software engineering domain. Evolvability as a quality attribute is defined as:

> *"An attribute that bears on the ability of a system to accommodate changes in its requirements throughout the system's lifespan with the least possible cost while maintaining architectural integrity."* (Rowe et al., 1998)

Furthermore, Maintainability as a quality attribute is defined as:

> "*The ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment.*" (Radatz, Geraci, & Katki, 1990)

Another, somewhat broader, and more popular definition of maintainability is provided in the ISO 9126 standard:

> "*Maintainability is the capability of the software product to be modified. Modifications may include corrections, improvements or adaptations of the software to changes in environment, and in requirements and functional specification.*" (ISO, 2011)

In their work, Weiderman et al., (1997) focus on distinguishing the concepts of maintenance and evolution. They discuss that maintainability represents fine-grained and short-term activities focused on localized changes. Maintainability from this perspective results in system structure which remains relatively constant. Changes which are labelled as maintenance often produce few economic and strategic benefits as it is common to only respond to one requirement at a time. On the other hand, evolution is a coarser grained, higher-level of abstraction, structural form of change that aims to modify the software to ensure, for example, easier maintenance. Evolution of software often comprises a larger scope, responding to multiple requirements, by creating whole new capabilities.

Modifiability

To further scope on literature regarding modifying logical architectures we analyse the concept of modifiability. Modifiability is about change, and the cost and risk of making those changes (Bengtsson, 2002). Literature on modifiability in the software architecture domain contains a plethora of definitions, from which some popular are selected to define and scope context for this research. A very early definition of modifiability comes from Boehm et al. (1978):

> *"A software product possesses the characteristic maintainability to the extent that it facilitates updating to satisfy requirements. A maintainable software product is one which is understandable, testable, and easy to modify"*

Furthermore, a more recent and widely adopted definition of modifiability is provided by Bass et al., (2000):

> *"Modifiability is the ability of a system to be changed after it has been deployed"*

To refer back to the discussion regarding the interchangeable use of maintainability and modifiability Bengtsson (2004) argues that the existing definitions of maintainability all share a broad perspective on changes to software systems while modifiability of a software system can be defined as:

> *"The modifiability of a software system is the ease with which it can be modified to changes in the environment, requirements or functional specification."*

Furthermore, Bengtsson states that the definition provided above demonstrates the essential difference between maintainability and modifiability, where maintainability is concerned with the correction of bugs an errors, whereas for modifiability this is not the case (Bengtsson, 2002).

It becomes increasingly important to change with increased speed to keep up with evolving markets and with that, evolving requirements. Evolution of software systems is characterized by modifications to the software system which includes extending, deleting, adapting, and restructuring (enterprise) software systems (Chapin et al., 2001). Modification efforts can range from adding a functionality as part of a functional requirement in a single software system to the implementation of large structural improvements to the software infrastructure as often seen in enterprise information systems (Andersson, 2002). A factor that influences the complexity of software systems is that most of them are interconnected and interchangeably used to process data, information, and knowledge across enterprises and/or large chains of collaborating organizations (Lankhorst, 2009). This factor is one of the reasons modifiability should be taken into account when designing or modifying software systems. Modifications to one software system potentially cause a ripple effect among other, interconnected software systems, creating not only primary changes that were meant to be implemented, but also secondary changes that may not be taken into account (Lindvall, Tvedt, & Costa, 2003). More of such factors which potentially influence modifiability of a software system can be found in the work of Lagerstrom (2010) and Bass et al, (2012).

Bass, Clements, and Kazman (2012), propose four important questions for software architects to focus on when dealing with modifiability of software systems: 1) *What can change?*, 2) *What is the likelihood of the change?*, 3) *When is the change made and who makes it?*, and 4) *What is the cost of* *the change?* In this research we mainly focus on the first three questions. For this research, we omit the latter question for future research purposes.

Architectural stability and flexibility

To be able to control software evolution, Mannaert and Verelst (2009) propose the concept of *stability.* Stability, in the sense of software evolvability, is dependent from two dimensions, 1) *the time period* and 2) *the number of changes*. This means that the system becomes ever larger in the sense that the number of primitives, and the number of dependencies between them become infinite as T -> ∞. The concept of primitives is essentially a generic term for an instantiation of a construct in programming language, for example, procedures, functions, or classes. Mannaert and Verelst (2009) propose this theory as the *assumption of unlimited systems evolution*. In short, the concept of stability demands that the impact of the change is only dependent on the nature of the change itself. Changes that are dependent on the nature of the change as well as the size of the system are termed as combinatorial effects. Combinatorial effects can also be defined as ripple effects in software architectures, potentially impacting large parts of the system due to nested dependencies. Furthermore, Mannaert and Verelst (2009) state that combinatorial effects should be eliminated from the system structure, thus its architecture, in order to attain the concept of stability. Eliminating combinatorial effects from the system structure results in a normalized system, in which the authors are inspired by normalization theories concerning the database research field. The following definition of the concept of stability is provided and adhered to in this thesis:

> *"Normalized systems are information systems that are stable with respect to a defined set of anticipated changes, which requires that the changes results into a known amount of impact on system primitives"* (Mannaert & Verelst, 2009)

They therefore formulate the following postulate: "*An information system needs to be stable with respect to a defined set of anticipated changes*".

The exact opposite of the concept of *stability* is the concept of *flexibility*. We interpret this concept as the flexibility of a software product, and in particular to what extent this concept has a relationship with the domain of enterprise, software (system), and data warehouse architectures. While the concept of stability is determined by changes that are known to occur or planned, flexibility as a concept comprises the exact opposite. An exact definition is provided by Port and Liguo (2003):

> *"Flexibility is a quality property of the system that defines the extent in which the system allows for unplanned modifications"*

Another, slightly broader definition is provided by the IEEE (1990):

> *"The ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed"*

Flexibility is becoming more and more important due to the growing user-demand of software systems, provided as services. Nowadays, suppliers more often ensure their software is available on central servers on a pay-per-use basis. While this seems only a change in the delivery mechanism, the more fundamental change was how the software itself is constructed, especially larger software applications delivered as SaaS. Rather than software components are being developed and melted together into one single form, architectures are more frequently designed and developed as a pool of services which are only connected at the point of execution. Such designs allow alternative software components to be substituted between uses of the same system, allowing much more flexibility compared to the more conservative architecture designs (Bennett & Rajlich, 2000).

Literature on architectural stability and flexibility strongly stresses that neither a complete view of possible changes can be chosen without taking into account the opposite perspective (stability versus flexibility). Authors argue that software design is often a trade-off situation, similar to the trade-off concerning quality attributes of software architectures as described earlier in this chapter. In the following section we elaborated in more detail modifications as part of modifiability. Earlier research on modifications in the BRM domain will help us identify a set of anticipated changes to define our input for further analysis of the included architectural candidates utilizing modification scenarios.

3.3.2 Modification types

To be able to measure modifiability as a quality attribute of a given BRA we need to define what modifications are relevant to include in our analysis. First, a short literature overview is presented regarding the available literature on modification types presented in neighbouring research fields. This is followed by a literature overview concerning modification types specifically for the business rules research field.

Modification types for neighbouring research fields

First, we analyse literature on modifications in IS research. In the software engineering domain, a modification is defined as (Stark, 1996):

> *"A change that is not a part of the current system requirements"*

If we analyse literature on computer programming and database operations, Create, Read, Update, and Delete operations are not to be ignored. The concept of CRUD was introduced by Martin in 1983 as four basic functions of persistent storage for relational databases (Martin & Savant Institute, 1983). Since then, many variants were introduced with the purpose of adding or evolving the basic functions for persistent data processing. For example, *read* can be replaced with *retrieve* since it is not always applicable to literally only read a document, but a private instantiation of the document is required. Another example comes from the usage of the term *destroy* instead of *delete* where literally destroying a record is a requirement. A range of possible additions are: *Attach*, *Append*, *Aggregate*, *Anonymise*, *Find* (also referred to as *search*), and *Transfer* (Graves, 2013). Further elaboration of all possible modifications that is beyond the scope of this research. Literature on software engineering reveals similar concepts compared to the CRUD acronym. The domain of software evolution and maintenance also gained popularity in the past decades. The work of Chapin, Hale, Khan, Ramil, and Tan (2001) identifies four clusters of software evolution with twelve underlying cluster types which represent a certain mode of evolution and change of software systems. Furthermore, a widely used set of modification types for the software architecture domain is that of Kazman et al. (1994) based on the work of Oskarsson (1982), which identified four different classes of modifications relevant for analysing the properties of software architectures:

1) **Extension of capabilities,**
2) **Deletion of unwanted capabilities,**
3) **Adaptation of new operating environments, and**
4) **Restructuring.**

The theories addressed in this section are not exhaustive as we know more similar, less popular identifications, classifications, and ontologies exist (Gold & Mohan, 2003; Kemerer & Slaughter, 1999; Weber, Reichert, & Rinderle-Ma, 2008; Weissgerber & Diehl, 2006).

Modification types for business rules

Again, due to the low maturity of the business rules research field, little research is performed on modification types regarding business rules enactment. To be able to measure the modifiability of architectural candidates we search for possible modification types in the available literature. Chapin et al., (2001) identified business rules as a separate concern important in the evolution and maintenance of software systems. The authors state that among the other clusters (i.e. Support interface, Documentation, and Software properties) identified as concerns relevant for software evolution and maintenance, business rule modifications are the most frequent and significant, usually relying on extensive supportive use of activities from the other clusters. In their work they propose that business rules modifications are either 1) *Reductive*, 2) *Corrective*, or 3) *Enhancive* of nature. The first modification archetype, Reductive, comprises reducing the business logic implemented. The second modification archetype, Corrective, comprises refinement and making more specific of implemented business rules. The third modification archetype, Enhancive, comprises changing and adding upon the repertoire of software implemented business rules to enlarge or extend their scope. Another important (theoretical) observation is that due to increasingly complexity of customer's operations, the archetype Enhancive is usually the most common and most significant in terms of effort required. If we evaluate the identified modification types for business rules against modification types of neighbouring fields we recognize the similarities of the CRUD operations coined by Martin (1983).

Based on the meta-models as elaborated upon in section 3.2 we can determine the theoretical modification impact for each included architectural candidate needed for analysis later on. Earlier research as presented in (Zoet et al., 2015) aimed to discover modification types specifically for the case organization selected for this research project. We utilize these modification types as a framework to express impact per included architectural candidate on the meta-level, whereas later in this thesis the instanced architectural candidates will be prompted for impact as input for the analysis of modifiability. We believe the following modification types are applicable for further investigation in the analysis of modifiability of the three included architectural candidates:

<table>
<tr><td colspan="3">A. Create decision: This modification adds an additional decision or sub-decision to the already existing set of business rules. This includes all underlying variables such as new business rules and new fact values.</td></tr>
<tr><td>Rule family-oriented</td><td>Fact-oriented</td><td>Decision-oriented</td></tr>
<tr><td>The creation of a decision involves creating a new decision, accompanied with a minimal of one rule family below it.</td><td>The creation of a decision involves creating a fact with a minimum of one fact type, and decision logic containing the logic for the fact.</td><td>The creation of a decision involves creating a new decision accompanied with decision logic containing the logic for the decision. Furthermore, a relationship is created to link with other decisions.</td></tr>
<tr><td colspan="3">B. Delete decision: This modification deletes a decision that, for example became obsolete. This includes all underlying variables such as new business rules and new fact values.</td></tr>
<tr><td>Rule family-oriented</td><td>Fact-oriented</td><td>Decision-oriented</td></tr>
<tr><td>The deletion of a decision involves deleting a minimum of the decision and the underlying rule family.</td><td>The deletion of a decision involves deleting the fact, the minimum of one fact type used in the fact, and the decision logic accompanying the fact.</td><td>The deletion of a decision involves deleting the decision itself, the accompanying decision logic, and the relationship used to relate the decision to another decision.</td></tr>
<tr><td colspan="3">C. Update decision: This modification solely updates the name (label) of a specific concept without changing underlying logic.</td></tr>
<tr><td>Rule family-oriented</td><td>Fact-oriented</td><td>Decision-oriented</td></tr>
<tr><td>Update a decision involves the modification of solely the decision.</td><td>Updating a decision involves modifying the fact and its accompanying decision logic.</td><td>Updating a decision involves modifying the decision, its accompanying decision logic, and the relationship containing the name of the decision as a relation with another decision.</td></tr>
<tr><td colspan="3">D. Create business rule: This modification creates a new business rule within the business rule set of a given decision, including one or more conditions and one conclusion.</td></tr>
<tr><td>Rule family-oriented</td><td>Fact-oriented</td><td>Decision-oriented</td></tr>
<tr><td>The creation of a business rule involves the modification of the targeted rule family which will contain the business rule.</td><td>The creation of a business rule involves the modification of the decision logic which will contain the business rule.</td><td>The creation of a business rule involves the modification of the decision logic which will contain the business rule.</td></tr>
<tr><td colspan="3">E. Delete business rule: This modification deletes an existing business rule within the business rule set of a given decision, including one or more conditions and one conclusion.</td></tr>
<tr><td>Rule family-oriented</td><td>Fact-oriented</td><td>Decision-oriented</td></tr>
<tr><td>The deletion of a business rule involves the modification of the targeted rule family containing the business rule.</td><td>The deletion of a business rule involves the modification of the decision logic containing the business rule.</td><td>The deletion of a business rule involves the modification of the decision logic containing the business rule.</td></tr>
<tr><td colspan="3">F. Create condition: This modification creates a new condition to be used by existing or new conclusions.</td></tr>
<tr><td>Rule family-oriented</td><td>Fact-oriented</td><td>Decision-oriented</td></tr>
<tr><td>The creation of a condition involves the addition of a</td><td>The creation of a condition involves the creation of a fact. A condition is always used by</td><td>The creation of a condition involves the creation of the</td></tr>
</table>

condition to a single existing rule family.	another fact so it is added as a fact type as part of its parent fact. Furthermore, the accompanying decision logic should be complemented with the new fact type.	condition itself in the decision logic of its parent decision.
G. Delete condition: This modification deletes an existing condition from a given ruleset.		
Rule family-oriented	**Fact-oriented**	**Decision-oriented**
The deletion of a condition involves the deletion of a condition from the existing rule family.	The deletion of a condition involves the deletion of the fact, but also the fact type must be deleted from its parent fact and its accompanying decision logic.	The deletion of a condition involves he deletion of the decision, the accompanying decision logic, and the relationship.
H. Update condition: This modification solely updates the name (label) of a condition.		
Rule family-oriented	**Fact-oriented**	**Decision-oriented**
Updating a condition involves the modification of the targeted rule family.	Updating a condition involves the modification of the fact type, the parent fact which utilizes the condition, and the decision logic accompanying the fact.	Updating a condition involves the modification of the decision, the accompanying decision logic, and the relationship.
I. Create fact value: This modification creates a new fact value for its parent condition or conclusion.		
Rule family-oriented	**Fact-oriented**	**Decision-oriented**
The creation of a fact value involves the addition of a fact value to the targeted rule family.	The creation of a fact value involves the addition of a fact value within an existing decision logic element.	The creation of a fact value involves the addition of a fact value within an existing decision logic.
J. Delete fact value: This modification deletes an existing fact value from its parent condition or conclusion.		
Rule family-oriented	**Fact-oriented**	**Decision-oriented**
The deletion of a fact value involves the deletion of a fact value to the targeted rule family.	The deletion of a fact value involves the deletion of a fact value within an existing decision logic element.	The deletion of a fact value involves the deletion of a fact value within an existing decision logic.
K. Update fact value: Due to the granularity of this modification category updating or renaming a fact value is essentially the same.		
Rule family-oriented	**Fact-oriented**	**Decision-oriented**
Updating a fact value involves the modification of a fact value utilized by the targeted rule family.	Updating a fact value involves the modification of a fact value within an existing decision logic element.	Updating a fact value involves the modification of a fact value within an existing decision logic.

In the next section we present a literature overview of the measurement of the quality attribute modifiability, and further argument on the selection of a method to utilize in this research project.

3.4 Measuring modifiability

Now that we discussed the context of modifiability and how it is applicable for the BRM domain we elaborate in more detail on how we are able to evaluate and determine to what extent an architectural candidate is modifiable. In this section we continue our literature review on how to quality attributes of architectures are measured, with a strong focus on modifiability. Utilizing a method for this is important due to the structure it offers in the analysis process, guiding practitioners to rigorously perform the right steps and gathering the right information during analysis in the early stage of software engineering (Len Bass et al., 2012). It is important to perform architecture analysis in this early stage in order to detect problems related to quality attributes before they get unmanageable (Berander et al., 2005).

Literature on measuring quality attributes in the context of BRA's is absent. To be able to adopt a method for the analysis of modifiability in the BRM domain we analyse literature on architecture evaluation methods (AEM's) in neighbouring fields. Literature review reveals that a lot of research regarding architecture evaluation has been performed in the software architecture domain (Babar & Gorton, 2004; Dobrica & Niemela, 2002; Kazman, Bass, Klein, Lattanze, & Northrop, 2005; Mattsson, Grahn, & Mårtensson, 2006). Many of these works include either methods to evaluate quality attributes in general, in a trade-off situation (Kazman, Bass, Webb, et al., 1994; Klein et al., 1999), or performance, which is the most researched quality attribute in terms of AEM's (Mattsson et al., 2006). Examples of performance AEM's are RARE (Barber, Graser, & Holt, 2002), LQN (Petriu, Shousha, & Jalnapurkar, 2000), and SPE (Williams & Smith, 1998)

Since our scope does not extend beyond that of the evaluation of solely the quality attribute modifiability we only include relevant AEM's which solely focus on modifiability, also excluding any relevant trade-off relationships with other quality attributes. Literature research reveals six AEM's which (partly) focus on modifiability. The results are provided in Table 14.

Table 14 - Literature findings on AEM's for modifiability

Method label	**Description**	**Source**
EBAE	The AEM Empirically-Based Architecture Evaluation method was created in 2003 and focuses on quantitative measurement of architectural measurements to compare architectural candidates. EBAE consists of four activities: 1) *perspective selection*, 2) *define and select metrics*, 3) *collection of metrics*, and 4) *evaluate/compare architectural candidates*. EBAE requires its users to define the goal of the analysis, define qualitative, but primarily quantitative metrics based on the quality attribute analysed, and define criteria to analyse and interpret the results of EBAE. Depending on the metrics utilized, the results of EBAE can either be interpreted via qualitative or quantitative data.	(Lindvall et al., 2003)
SAAM	The AEM Software Architecture Analysis Method was created in 1994 and has a broad focus on multiple quality attributes of software architectures. SAAM consists of five activities: 1) *characterization of a canonical functional partitioning*, 2) *mapping of the functional partitioning to the architecture's structural decomposition*, 3) *selection of relevant QA's to involve for the*	(Kazman, Bass, Abowd, & Webb, 1994)

	assessment of the architectural candidate, 4) *selection of a set of tasks to test the QA's*, and 5) *evaluate the results*. SAAM requires its users to create a structure of the system, allocate functions, define scenarios, and manually analyse the effects on the system architecture. The results are interpreted in a qualitative way and ranking was applied, since no metrics were defined at that time to evaluate software architectures.	
ATAM	The AEM Architecture Trade-off Analysis Method was created in 2000 and has a broad focus while giving extra attention to the trade-off between different quality attributes. ATAM is a successor to SAAM, consisting of four main phases each covering two activities: 1) *scenarios and requirements gathering*, 2) *architecture and scenario creation*, 3) *model building and analysis*, and 4) *trade-off analysis*. ATAM requires users to collect scenarios and requirements, define and describe architectural views, and gather input data for the selected quality attributes. The analysis is performed manually in a qualitative manner. The results are expressed in risks, sensitivities, and trade-offs between quality attributes after analysis is performed.	(Kazman, Klein, & Clements, 2000)
ALMA	The AEM Architecture-Level Modifiability Analysis method was created in 2002 and focuses entirely on the analysis of modifiability. It is largely based on the AEM's SAAM and QASAR and features five activities: 1) *set analysis goal*, 2) *describe software architecture*, 3) *elicit scenarios*, 4) *evaluate scenarios*, and 5) *interpret analysis results*. ALMA requires input from stakeholders, especially from software architects, who need to create (high-level) architectural descriptions, coordinate the elicitation of scenarios with other stakeholders, and manually determine the effect of scenarios on the architectural descriptions. The results of the first three steps are entered into an equation which result in an estimation of total effort needed to process the included changes for a given amount of time.	(Bengtsson et al., 2004)
QUASAR	The AEM Quality-Driven System Architecting method was created in 2000 and consists of three main phases: 1) *preparation*, 2) *modelling*, and 3) *evaluation*. QUASAR has a broad focus, providing the possibility to assess most quality attributes. Furthermore, QUASAR requires its users to identify drivers and requirements as part of the problem definition, determine mechanisms to evaluate in the third phase, model the required architectural views, and defining scenarios to utilize in the analysis of the architectural candidates. QUASAR can be applied either in a qualitative or quantitative manner. Depending on the measures and parameters utilized, the results of QUASAR can either be interpreted via qualitative or quantitative data.	(Bosch, 2000)
ABAS	The AEM Attribute-Based Architectural Styles method was created in 2006 and has a very broad focus on the analysis of multiple concurrent quality attributes. ABAS utilizes architectural styles to apply reasoning in a framework as method of analysis. ABAS consists of five activities: 1) *problem definition*, 2) *define quality attribute measures*, 3) *define architectural styles*,	(Klein et al., 1999)

	4) *define quality attribute parameters*, and 5) *perform the analysis*. The reasoning framework can be applied either in a qualitative or quantitative manner. Depending on the measures and parameters utilized, the results of ABAS can either be interpreted via qualitative or quantitative data.	

Method selection

The purpose of this research allows for the evaluation of a single AEM to analyse three architectural candidates. This implies that we are to select one AEM from which we believe fits best for utilization in the BRM domain at this point. To ground the method selection process we predefined three knock-out criteria that exclude AEM's for further analysis: **1)** the method *focuses solely on the evaluation of modifiability* without including relationships and/or trade-offs with other or related quality attributes, **2)** the AEM must be *cited at least 100 times*, **3)** the AEM must be *utilized in further research* e.g. other (consecutive) research papers etc.

1. The first criterion is utilized to filter any AEM which has a focus that is too broad or relies heavily on trade-off analysis between different quality attributes. It is important that the selected AEM provides us the possibility to reveal in-depth information regarding modifiability to reveal components which increase modifiability for BRA construction. This criterion results in two complying AEM's: ABAS and ALMA. The other four AEM's are very broad in terms of involvement of different quality attributes other than maintainability and modifiability (SAAM, ABAS, and QUASAR) or general and focused on trade-off analysis between different attributes (ATAM). While the AEM EBAE focuses on the quality attribute maintainability, ALMA purely focuses on the quality attribute Modifiability, although both are not very different in terms of input data and method of analysis.

2. The second criterion is utilized to ensure scientific rigor by analysing to what extend an AEM is popular amongst researchers, which can be derived from the amount of citations in combination with the following criterion. All AEM's with the exception of the AEM EBAE (50 citations) comply with this criterion.

3. The third criterion is utilized to ensure scientific rigor and popularity in the architecture research domain as well. A literature search is performed to search for follow-up studies regarding the proposed theoretical AEM, either by the same authors, but preferably by other researchers that aim to validate the theoretical AEM in practice with case studies or with the utilization of realistic research data. Again, all AEM's with the exception of the AEM EBAE comply with this criterion.

Evaluating the results of the method selection guided by the three criteria as described we select the AEM ALMA to serve as the best fit for our analysis of modifiability in the BRM domain. Since the AEM ABAE is less popular amongst researchers and practitioners which is tested by criterion two and three we conclude this AEM as potentially less rigorous as the AEM ALMA. Furthermore, the AEM ALMA is largely based on the AEM's SAAM and parts of QUASAR, which are both general AEM's, but are specifically tailored for analysing the quality attribute modifiability of (high-level) software architectures. In the next section the AEM ALMA is fully described and made explicit to adept for utilization in the BRM domain as elaborated further upon in section 3.3.5.

3.4.1 ALMA for the SA domain

According to Bengtsson, Lassing, Bosch, and van Vliet (2004) Architecture-level modifiability analysis (ALMA) is a solution concerning the lack of scientific methods of architectural analysis methods specifically designed to focus on modifiability. ALMA is characterized by explicit assumptions and is designed to distinguish multiple analysis goals while providing repeatable techniques for performing the ALMA steps. ALMA is utilized to result into a modifiability prediction model, based on case-specific scenarios. The following paragraphs will elaborate more upon the steps that need to be performed when adhering to ALMA. ALMA consists of the following five steps:

1. **Set goal: determine the aim of the analysis**
2. **Software architecture description: give a description of the relevant parts of the software architecture**
3. **Elicit scenarios: find and document a set of relevant scenarios**
4. **Evaluate scenarios: determine the effect of the set of scenarios**
5. **Interpretation draw conclusions from the analysis results**

The first step is focused on determining the goal of the analysis. According to Bengtsson et al. (2004), the following goals can be pursued: 1) *Maintenance cost prediction*, 2) *Risk assessment*, and 3) *Software architecture selection* based on modifiability scores. Maintenance cost prediction aims to estimate the effort that is required to modify the system to accommodate future changes. Organizations want to be able to estimate, but more importantly, calculate as precisely as possible the costs of maintenance. Risk assessment aims to identify the types of changes for which the software architecture is inflexible. And lastly, ALMA can be utilized to guide the process of selecting a software architecture from multiple candidate solutions based on modifiability scores. The second step focuses on collecting or generating information concerning the software architecture candidates. This architectural information is required to evaluate scenarios with the help of ALMA. The analysis is utilized to identify the architectural elements affected by the change scenarios. Effect of these change scenarios can be categorized in direct and indirect effects. To measure this effects, the architecture descriptions need to be constructed. This step has three requirements towards the architectural description of the architecture candidate. The architectural description needs to include:

1. **The decomposition of the system in components;**
2. **The relationships between these components;**
3. **The relationships to the system's environment.**

Furthermore, ALMA does not imply the use of a certain technique or representation to capture architectural information or architectural knowledge, as long as the technique or representation covers the aforementioned requirements. Techniques to do so are further elaborated in the work of Medvidovic (2000). Once architectural descriptions of the system in their corresponding architectural candidates are created ALMA continues towards the third step. This step focuses on the elicitation of relevant scenarios used in the subsequent steps that include conducting the analysis with the help of selected scenarios. The elicitation of scenarios is performed by identifying stakeholders, eliciting the scenarios from surveys or interviews, and documenting the scenarios. One of the issues of the elicitation of scenarios concerning possible changes is that the number of possible changes is almost infinite (Bengtsson et al., 2004;

Bengtsson, 2002), this also results in an increase of the amount of possible scenarios for the analysis. To ensure feasibility and stability of the analysis and its results, two techniques are utilized to counter the infinity of possible changes in a certain situation: equivalence classes and classification of change categories. With equivalence classes all the possible scenarios are categorized into classes and a representative criteria is created to represent this class of scenarios best. This results in a significant decrease of scenarios that have to be considered in the analysis. The classification of change categories is based on the same principle, by creating certain change categories out of the infinite amount of changes which best represent them.

The process of eliciting scenarios can be guided following two approaches, the top-down approach or the bottom-up approach. With at top-down approach the process focuses on the use of predefined classifications of changes and scenarios to trigger and stimulate stakeholders to bring forward relevant scenarios. The bottom-up approach is designed to work the other way around, without predefinitions of change and scenario classifications, solely focusing on the elicitation of relevant scenarios by stakeholders. Both approaches have advantages and disadvantages, and it is possible to combine both in the process of eliciting scenarios. In this step, the following focus areas should be taken into account: 1) to estimate modifiability effort, ALMA needs scenarios which correspond to changes that have a high probability of occurring during the operational use of the system and 2) the scenarios selected need to expose risks of the architecture in relation to changes made in order to be able to perform risk assessment. For the analysis and its results to be accurate and feasible, it is important to identify saturation of input data for the analysis, essentially, when all relevant scenarios with corresponding possible changes are elicited (Glaser, 1978). This is the case when 1) all change categories are explicitly considered, and 2) new change scenarios do not affect the current classification structure. When it is decided that all relevant types of changes and scenario classifications are considered and documented the next step of ALMA is initiated. In this step, the actual architecture-level analysis takes place utilizing the scenarios as documented in previous steps. In general, impact analysis consists of three steps (Bengtsson et al., 2004; Bengtsson, 2002):

1. **Identification of affected components;**
2. **Identification of the effect on the affected components;**
3. **Identification of ripple effects.**

To calculate maintenance prediction, there is one more step to perform before the actual analysis, by the means of the main formula, is performed. The main formula of the analysis includes scenario weights which can be interpreted as the probability of scenario occurrence. The probability is transformed into weight, which is used relative to other weights in the formula. It is used to determine the scenario's influence on the end result. To be able to produce sound and relative results, first, stakeholders need to identify the importance or occurrence of each scenario containing specific change categories. This can either be done using estimated number of changes for each scenario can be expected or weights on a fixed scale, as long as the same scale is utilized for all respondents. After the process of the elicitation of weights per scenario has been completed, the weights need to be normalized. This is done by a rather simple formula depicted in Equation 1. The normalized value of a scenario, NW (S_n), is calculated using the estimated number of changes or weight, W (S_n), divided by the sum of the estimated changes or weights. The normalized weight, NW (S_n), should always be between zero and one, while the sum of all NW (S_n) always equals one. The list of scenarios with normalized weights is defined as the scenario profile which is used in the equation.

$$\mathrm{NW}(S_n) = W(S_n) \Big/ \sum_n W(S_n)$$

Equation 1 – Normalized scenario weight determination (Bengtsson et al., 2004)

After the change scenarios and their normalized weights have been determined, the next step can be undertaken. This involves the calculation to what extent an architectural candidate is modifiable, based on the scenarios as determined earlier in the process. To determine this, three equations are utilized. These three equations represent the architectural candidate's best-case modifiability score, its worst-case modifiability score, and the predicted maintenance effort, which represents the modifiability score somewhere in the range between the lower and upper boundaries of the analysis. The analysis is based on two assumptions made by Bengtsson (2002). The first assumption concerns that there is a difference in maintenance effort required which is due to differences in productivity scores, i.e. $P_{\text{new code}}$, $P_{\text{existing code}}$, etc. Note that these productivity scores include more than simply writing the source statements. For example, this also includes searching for the right source documents or systems, i.e. requirements, logging in, and/or publishing the new or updated code. The second assumption concerns the size invariance of modifications. The amount of code to be developed or changed for a scenario is relatively constant in size, independent of software architectural candidates. For example, when the same standard programming language is used, by the same programmers, in the same domain or context, modifications require fairly the same amount of code to be created or updated. This particular approach implies that impact analysis results from architectural candidate A are roughly similar for any other architectural candidate involved in the comparison. This allows researchers and practitioners to calculate the results from a hypothetical architecture without fully creating the architecture. Based on these two assumptions Bengtsson (2002) states that a best-case modifiability situation is determined by the software architectural candidate which allows practitioners to implement modifications at the highest productivity level. The overall productivity level of software engineers is taken as an average and can (theoretically) be deducted via two methods. The first method, being an analysis of the organization capacity in terms of software engineer productivity. The second method is referring to available scientific knowledge for figures on overall average capacity of software engineering on modifying (CRUD) the software system, for example, lines of code (J. E. Henry & Cain, 1997; Maxwell, Van Wassenhove, & Dutta, 1996). However, the latter method is simple and accessible, it is important that organizations utilize figures of productivity related to their own workforce dealing with modifying software systems due to the accuracy of the productivity figures over any scientific proof, also due to many possible contingency factors which render the scientific figures from other case studies mostly inaccurate.

To determine the effort required for each architectural candidate five dimensions are required to be known:

1. The **normalized weight** of the selected scenario, depicted by **j**
2. The **size** of the modification expressed as lines of code or function points, depicted by size ***S***
3. The **overall productivity level** of the software engineers performing the modification type, depicted by ***Pn***

4. The **sum of scenarios** in the scenario profile, depicted by **C(MP)**
5. The **total amount of estimated changes**, depicted by **CT**

The total maintenance effort can be predicted using Equation 2.

$$E_M = \frac{\sum_{AI}\left(\left(\sum_{CC_i} s_j\right)\cdot P_{cc} + \left(\sum_{NP_i} s_j\right)\cdot P_p + \left(\sum_{NC_i} s_j\right)\cdot P_{nc}\right)}{C(MP)} \cdot CT$$

Equation 2 – Total effort estimation as proposed in the work of (Bengtsson, 2002)

The prediction model proposed by Bengtsson et al. (2004) relies on types of modifications which are accompanied with their corresponding productivity levels. Each scenario comprises a minimal of one modification type together with the impact size estimate as elaborated upon earlier in this section. The individual sums of the product of the impact size estimates and the normalized weight of the scenario multiplied with their associated productivity level make up the total effort required for the modifications in the scenario profile.

3.4.2 ALMA for the BRM domain

To be able to utilize ALMA in the context of business rules and business rule architectures, multiple modifications have to be made to the method for it to be able to function appropriately, see also a visualization according to the PDD-modeling standard (Weerd & Brinkkemper, 2009) Appendix L. First, the fit of the main steps of ALMA need to be determined. Since the main steps of ALMA are very generic of nature as they do not refer to software architecture.

The first step comprises formulating a goal. The goal regarding the utilization of ALMA in this particular study is to compare three selected business rule architectures on effort prediction. As this is similar to the application of ALMA in the SA domain we did not modified this.

The second step requires the construction of architectures while taking into account the three requirements as stated for the SA domain. For the construction of business rules architectures the first requirement, which addresses the decomposition of components, is transformed into the decomposition of decisions and conclusions. The second requirement, the relationships between components, can be adopted without adaptation for the construction of business rules architectures due to the logical relationships that are needed between conclusions to come to a certain decision. The third and last requirement, the relationships to the system's environment, is omitted due to the single responsibility design principle we adhere to for the construction of the business rules architectures. The impact of this particular design decision is neglectable and only serves as an extra complexity measure to identify impact with system environment for SA architectures, thus less relevant for business rules architectures. Furthermore, omitting the last requirement for this particular step in the ALMA process does not impact the formula in any way.

The third step, elicitation of change scenarios, knows major modifications in order to function appropriately for the analysis of business rules architectures. While ALMA for SA aims to elicit relevant change scenarios utilizing qualitative interviews with stakeholders, this study aims to elicit relevant scenarios based out of large volumes of historical change data. For the application of ALMA for the analysis of business rules architectures we introduce the use of quantitative data analysis to determine relevant change scenarios based on historic data on changes made to a particular product or service. This does not exclude the use of qualitative data analysis for the elicitation of relevant change scenarios for business rules architectures in other cases, but merely provides room for an alternative method to perform data analysis for this particular step. We believe that quantitative data is more reliable over qualitative data, especially when historical change data is available over longer periods of time. For this study we use the QOF change documents as input data which holds information on which change scenarios are possible. The amount of QOF change documents available provides the possibility to perform quantitative data analysis, although primarily on a descriptive level. Descriptive level statistics are sufficient in terms of granularity to provide an overview of two important elements of information: 1) *what changes occur* and 2) *where do they occur*. This is in line with the goals of step 3 concerning ALMA for SA. Furthermore, categorization as applied in ALMA for SA is also applicable for the analysis of business rules architecture, therefore we adopt the categorization of type of changes and scenarios. For the categorization the top-down approach is used, predefining possible types of changes to a business rules architecture.

The fourth step, change scenario evaluation, is similar for both application in the SA and BRA domain. First, the output of the previous step needs to be normalized, which is adopted from equation

1. The weighting of the change scenarios is one again followed by the identification of affected components, the effect on those components, and possible ripple effects. Similarly as discussed in the previous section, this operation needs to be performed manually, simulating impact on the affected BRA. For the productivity scores the same approach is adopted from ALMA for SA. Again, experienced stakeholders can estimate these scores, which differ per type of operation, but it is desirable to base those scores on average in-house information regarding productivity.

For both SA and BRA, the accuracy of ALMA's results rely on the accuracy of the productivity scores as it is an important factor of total effort determination within the main equation of ALMA. The average productivity scores are not available for business rules management or business rules architecture related operations due to the immaturity of the business rules management research field. As described in chapter two, interviews will be conducted to acquire case-specific information regarding productivity of the BRM experts employed at the case organization. From the results we have to determine the productivity scores that will be utilized as essential input for the ALMA-method. Since two interviewees will provide estimations on the time it takes to process the given modification types we have to decide how the results will be interpreted. As both experts may vary in experience level it provides an average but accurate enough view on productivity. Due to this we interpret the averages per modification type calculated from the results of both interviewees. Furthermore, we expect the interviewees to differentiate in complexity of modification types as some modification types are characterized by a certain hierarchy, causing more secondary changes to occur than others, as experienced in earlier research on this topic (Zoet et al., 2015), which can influence how the productivity scores could be interpreted. Due to the amount of respondents available at the case organization in combination with the nascent maturity level of the BRM domain we choose to solely utilize arithmetic calculations to determine the productivity scores for each modification type. More advanced inferential statistics are not applicable in this situation and should be conserved for utilization when more explorative research is performed on this topic and larger sample sizes are available.

After the basic arithmetic calculations are performed the results are expressed as a given amount of time units rather than a given amount of function points. To be able to calculate the total effort predicted per architectural candidate it is essential that the input for the productivity scores equals the same measurement scale utilized for size. Therefore we need to translate the time units that will be provided by the interviewees into function points. Since the available body of BRM literature does not include any research on conversion rates or ratios for BRM-related activities or modification types to function points we need to search for neighbouring fields of study that reached a higher maturity regarding this topic. First, literature on data warehouse-architectures is queried for function point conversion. This resulted in zero relevant sources. The neighbouring field of software engineering is more accustomed to utilize function points for the determination of effort (Kan, 2002). In the software engineering field the conversion of function points to working hours or vice versa is often referred to as Project Delivery Rate (PDR) (Bundschuh & Dekkers, 2008). To determine a usable conversion of hours per function points we include scientific and non-scientific sources as industry standards derived from practice by experts are probably accurate. Due to scope constraints we included ten sources which include a statement on hours per function point, which are listed in Table 15. We note that these conversion rates can be dependent on various factors. This limitation is further elaborated upon in Section 5.1. For conversion of the average productivity scores that will be derived from the interview we selected the average of 12.59 hours per function point.

Table 15 - Literature overview of project delivery rates Hours/FP

Source	H/FP	Sample size
(Niessink & van Vliet, 1997)	10.26	1
(Longstreet, 2008)	26.33	Multiple but unknown
(Abran, Gil, & Lefebvre, 2004)	9.96	236
(Alexander, 2008)	4	1
(Herron & Garmus, 2000)	11.2	Multiple but unknown
(Carpers, 1996)	14.08	Multiple but unknown
(Kan, 2002)	14	Multiple but unknown
(Morris, 2002)	16.7	2
(Shepperd, Mair, & Forselius, 2006)	7.5	661
(Bundschuh & Dekkers, 2008)	11.2	241
Total	x	906
Average	12.59	x

Lastly, the size of the scenario and their corresponding modifications need to be calculated. The original ALMA equation mainly focuses on the use of Lines Of Code (LOC) or kilo LOC (kLOC) to express the total size of a scenario and the size per modification (type). If we analyse the elements that make up the structure of a BRA we cannot express size as LOC or kLOC. In their work, Bengtsson et al. (2004) refer to three possible ways to express size which allows the main equation to effectively calculate predicted total effort, 1) *LOC or kLOC*, 2) *Function Points*, and 3) *Object points*. As far as the authors are aware, no literature exists on transforming business rules into LOC or Object Points. However, in the light of the scarcity of information regarding transformation of business rules and the remaining concept of Function Points as described eligible to utilize in the ALMA equation by Bengtsson et al. (2004), the work of Felfernig and Salbrechter (2004) addresses multiple types of variables to measure effort estimation utilizing function point analysis.

As this contribution is unique in terms of the utilization of function points in the BRM domain we continue with taking function points as a unit of measurement for our analysis of the architectural candidates later in this thesis. First, we define the concept of function points and address how we can utilize them in our analysis. The notion of function points is relatively old and extensively researched. Function points were first coined by Albrecht in 1979 and is defined as:

> *"A measurement for relative size and complexity of a software system used to estimate and express relative size and complexity in early stages of development, analysis, and design"*

Traditional function point analysis comprises a set of five factors, namely: External input, External query, External output, Internal logical file, and External interface file (Albrecht & Gaffney, 1983). Advantages which arise when utilizing function points for the calculation of modifiability of business rules architectures are (Symons, 1988):

1. **The measure isolates the intrinsic size of the system from its environmental factors;**
2. **The measure is based on the users external view of the system, and is technology-independent;**
3. **The measure can be utilized early in the development cycle;**
4. **They can be understood and evaluated by nontechnical users.**

Felfernig and Salbrechter (Felfernig & Salbrechter, 2004) argue that conventional function point analysis techniques do not explicitly consider business logic or business rules as building blocks of software systems. In general, business rules make up a significant part of knowledge based systems, having a strong influence on development time and costs (Felfernig & Salbrechter, 2004). To be able to include business rules as cornerstones for function point analysis a definition was constructed to measure the complexity of business rules, see Table 16. It is based on Record Element Types (RET) and Data Element Types (DET) in conjunction with the amount of business rules per logical function (BRLF). Translating this towards the business rules architecture domain we define RET's as conclusions as part of a decision and DET's as conditions as part of the conclusion. Next, the amount of conclusions and conditions are summed so the vertical axis can be determined. Furthermore, the logical function equals a given decision within a business rules architecture. Then, the amount of business rules per logical function is determined. An example of a logical function for the SA domain is depicted in Figure 15.

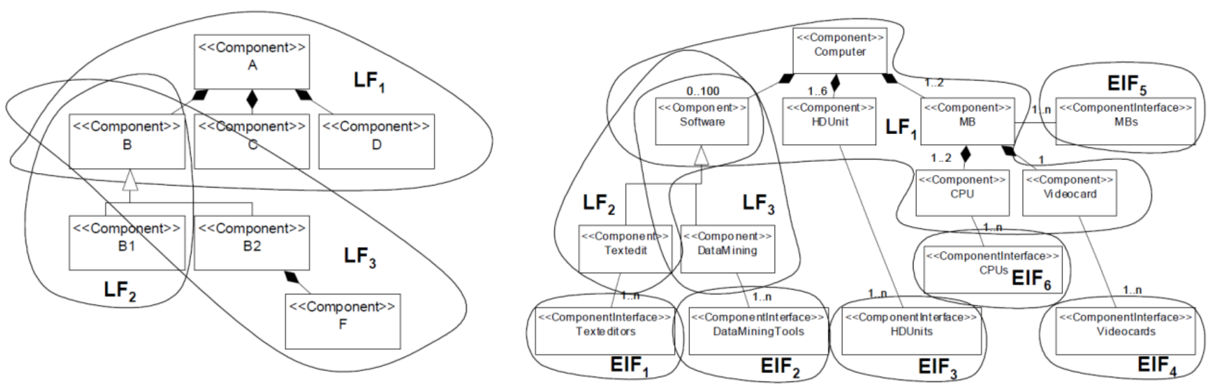


Figure 15 - Identification of logical files for SA components (Felfernig & Salbrechter, 2004).

The SA domain as represented in Figure 15 is, for example, represented by components and component interfaces, however, this is not the case for business rules architectures. As mentioned earlier, business rules architectures are primarily built upon decisions, including conclusions and conditions (Ross, 2003). Since the examples in Figure 15 do not provide a good fit with business rules architectures, a more relevant example is created to clarify the separation of logical functions within business rules architectures, which is depicted in Figure 16. In this example we utilized the TDM approach as an architectural description language. For the analysis of modifiability utilizing ALMA we chose to define a single logical function as one atomic architectural element, which applies to either the rule family-oriented, fact-oriented, and decision-oriented architectures.

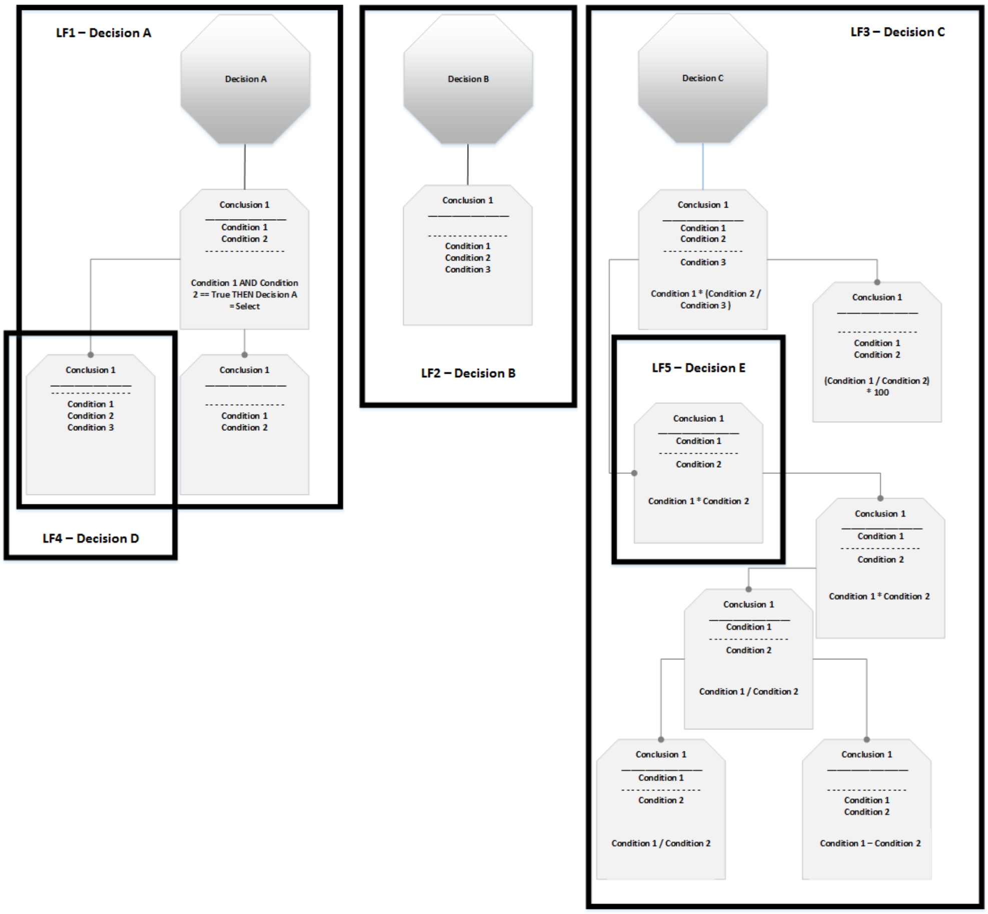


Figure 16 – Identification of logical functions for the business rules architecture domain.

Unadjusted function points per logical function

Once we determined the scope of base measurements within the determination of size as part of the analysis of modifiability, we define the complexity of each logical function to calculate the amount of function points per logical function changed later on. This measurement is imperative as it allows us to determine impact of a modification type and convert it into the function points measurement scale. The work of Felfernig and Salbrechter (Felfernig & Salbrechter, 2004) contains a definition of complexity classes for business rules as shown in Table 16.

Table 16 - Complexity of Business Rules (Felfernig & Salbrechter, 2004).

RET+DET	1-4 BRLF	5-9 BRLF	>9 BRLF
1 - 16	Simple	Simple	Average
17 - 40	Simple	Average	Complex
> 40	Average	Complex	Complex

Furthermore, for each complexity class a base value of the amount of function points needs to be determined. In their work, Felfernig and Salbrechter (Felfernig & Salbrechter, 2004) determined a set of unadjusted function points per complexity class. However, it is imperative that unadjusted function points are adjusted according to environmental or contextual factors. Function points that are adjusted to their environmental and contextual factors are also referred to as adjusted function points (Abran & Robillard, 1996; Felfernig & Salbrechter, 2004; Symons, 1988)

Adjusted Function Points

Literature (Abran & Robillard, 1996; Felfernig & Salbrechter, 2004) suggest the adjustment of function points. Function points are essentially unadjusted due to the lack of adjustment to organizational and system characteristics which can potentially influence analysis function point analysis results. For the adjustment of function points towards the case-organization we utilize the productivity scores derived from the qualitative interviews and the amount of time units per function point. With these measurements we are able to calculate the amount of function points per modification type which are adjusted to the organizational productivity measurements. As we expect the results of the productivity scores elicitation to be expressed in minutes per modification type we also transform the ratio of hours per function point into minutes per function point. For the calculation of adjusted function points we then divide the amount of minutes it takes to perform one modification type with the amount of minutes per function point.

ALMA's effort estimation

As all variables needed for the analysis of BRA's are made explicit, the main equation to calculate total effort applicable in the BRM domain is presented in Equation 3. Unlike the change code and create code modification types that are used in Equation 2 (ALMA for SA), we identified and distinguished eleven modification types for BRM maintenance in the work of (Zoet et al., 2015). These modification types are in line with earlier explorative attempts to theoretically distinguish modification types for BRM maintenance and evolution (Vanthienen & Snoeck, 1993; Zoet et al., 2011). We adhere to this standard and adapted the equation so that all eleven modification types are covered to measure the estimated maintenance effort adequately. As stated by Bengtsson (2004) Equation 2 can be dynamically adjusted depending on the modification types relevant. In we present a part of the ALMA equation as proposed in the work of Bengtsson et al. (2002) for application in the SA domain. This example contains two types of modifications: *Change Code* (CC) and *New Code* (NC). In the example we see that each modification type is represented by the calculation of its product and its accompanying productivity measurement (P_{cc} and P_{nc}).

$$\sum_{AI}\left(\left(\sum_{CC_i} s_j\right)\cdot P_{cc} + \left(\sum_{NC_i} s_j\right)\cdot P_{nc}\right)$$

Figure 17 - Example of the equation for SA featured by ALMA (Bengtsson et al., 2004)

In Figure 18 we present the adaptation of the equation for utilization in the BRM domain, but instantiated with the modification types as derived from (Zoet et al., 2015). In our example we replaced the original modification types as presented in the work of Bengtsson et al. (2004) with the modification categories of *Create Decision* (CD) and *Delete Decision* (DD) with their accompanying productivity measurements (cd and dd).

$$\sum_{AI}\left(\left(\sum_{CD_i} s_j\right)\cdot P_{cd} + \left(\sum_{DD_i} s_j\right)\cdot P_{dd} + \right.$$

Figure 18 - Part of the ALMA equation to be utilized in the analysis of BRA's

The example provided in Figure 18 is further extrapolated to cover all modification types in Equation 3. This increases the complexity of the calculation due to the inclusion of all eleven modification types as derived from earlier research (Zoet et al., 2015). To express the estimated effort for a given period we utilize $E_{M.}$ Furthermore, we utilize CD and cd for *Create Decision*, DD and dd for *Delete Decision*, UD and ud for *Update Decision*, CBR and cbr for *Create Business Rule*, DBR and dbr for *Delete Business Rule*, CC and cc for *Create Condition*, DC and dc for *Delete Condition*, UC and uc for *Update Condition*, CFV and cfv for *Create Fact Value*, UFV and ufv for *Update Fact Value*, and finally DFV and dfv for *Delete Fact Value*. Also, CT for estimated Changes Total and C(MP) for total Maintenance Profile are adopted exactly as applied in the SA domain. Finally, we decided to omit the calculation of best and worst case measurement of modifiability since we do not believe such theoretical data based on the lowest productivity figures provide added value concerning our calculation. Instead, we will utilize our data collected from the interviews conducted at the case organization to explore the differences of experience that will probably produce different outcomes regarding effort prediction. The scope of this study lies entirely on the calculation of total effort and the analysis and interpretation of differences between the included architectural candidates.

$$E_M = \frac{\sum\limits_{AI}\left(\left(\sum\limits_{CD_i} s_j\right)\cdot P_{cd} + \left(\sum\limits_{DD_i} s_j\right)\cdot P_{dd} + \left(\sum\limits_{UD_i} s_j\right)\cdot P_{ud} + \left(\sum\limits_{CBR_i} s_j\right)\cdot P_{cbr} + \left(\sum\limits_{DBR_i} s_j\right)\cdot P_{dbr} + \left(\sum\limits_{CC_i} s_j\right)\cdot P_{cc} + \left(\sum\limits_{DC_i} s_j\right)\cdot P_{dc} + \left(\sum\limits_{UC_i} s_j\right)\cdot P_{uc} + \left(\sum\limits_{CFV_i} s_j\right)\cdot P_{cfv} + \left(\sum\limits_{UFV_i} s_j\right)\cdot P_{ufv} + \left(\sum\limits_{DFV_i} s_j\right)\cdot P_{dfv}\right)}{C(MP)} \cdot CT$$

Equation 3 – Equation to predict the total effort based on the work of (Bengtsson et al., 2004)

Chapter conclusion

In conclusion we conducted our literature review based on three main research domains; business rules management, data warehouses, and software architecture. Since the theory of separating concerns in information systems development was introduced, multiple disciplines were identified as concerns that were separated to promote the separate management of information system quality goals. In the last decade the notion of logic or business logic gained increasingly more popular due to organizations employing business logic to guide or constraint the business based on external laws and regulations and internal policies. Research and practice increasingly define business logic as a separate concern in their information systems development lifecycle. As laws and regulations and policies change more often, evolvability of information systems should take into account the notion of modifiability of business logic as a separate concern. We utilized the data warehouse research domain as it is well known for its experience with modifiability of its data structures in large volumes. Within this domain we identified three architectural candidates which we want to utilize in the BRM domain and evaluate on modifiability; Inmon's data warehouses, Kimball's dimensional models, and Linstedt's data vaults. Next, we theoretically unravelled these three architectural candidates and identified two similar standards which could be applied in the BRM domain. The last architectural candidate, Linstedt's data vaults, is transformed to be applied in the BRM domain utilizing knowledge from BRM experts. As we defined our case organization and its underlying cases in the previous chapter in combination with an overview of both the BRM and Data warehouse research domains we concluded this chapter by first defining what we wanted to analyse (modifiability) and how this relates to other theories regarding for example evolvability and maintainability. As ALMA requires scenarios that are grounded on modification types we also reviewed literature on modification types and how these relate to information systems development and evolution. This was followed by a literature overview of methods to evaluate architectures. Literature suggest that the software architecture research domain is relatively mature in evaluating architectures, which led us to reviewing potential Architecture Evaluation Methods (EAM) from which we can select one as the method of analysing modifiability of the architectural candidates. Based on three criteria described in section 3.4 we selected ALMA as our preferred AEM. Lastly, this chapter concluded with a theoretical description of ALMA. However, several modifications were needed to be able to utilize ALMA for the analysis of modifiability concerning the three architectural candidates, which are elaborated upon in section 3.4.2.

In the following chapter, comprising the data analysis and results, we present the results of our research activities which were conducted to be able to utilize ALMA, followed by the results of ALMA as an AEM itself.

4 Data analysis and results

In this chapter the results of the data collection regarding the conducted interviews are presented. Furthermore, in this chapter we present the results of the implementation of the ALMA method in the BRM domain. The results are reported based on the four steps described in section 3.4.2.

4.1 Estimation of productivity

As elaborated upon in section 2.3 we conducted two interviews at the HSCIC which employs two business rule experts that translate requirements into business rules. In this section we present our findings. The interview transcripts are included in Appendix B.

Introduction

In total two interviews were successfully conducted, both with senior information analysts of the Health & Social Case Information Centre (HSCIC). Both interviewees are part of the business rule team which are responsible for the translation of stakeholder requirements into operational business rules which are implemented by another team further in the development process of data and IT systems for health and social care. Interviewee one has four years of experience as a business rule expert and is employed in the business rule team since its establishment. Interviewee two was less experienced, being employed for almost one year.

Both interviewees had difficulty with estimating to what extend their monthly activities were related to working with business rules regarding the QOF. This is due to the following two factors: 1) The *updates tend to be different in size* over the years, which influences capacity needed to process modifications to the QOF and 2) *Other business rule-driven services are also maintained* by the business rules team of the HSCIC. These services are also characterized by their own release cycle and release size, which is different from the QOF. Examples of these services are the Enhanced services and the Commissioning for Quality and Innovation (CQUIN) payment programme. Furthermore, stakeholders like the NICE also utilize the expertise and capacity of the HSCIC business rules team to perform feasibility tests on upcoming modifications to the services. Both interviewees stated that their activities comprise a constant cycle of incoming change requirements from different stakeholders in different sizes at different intervals applicable to one or multiple services the NHS delivers.

Productivity findings

The initial design of the questions regarding productivity scores were found insufficient during the start of the first interview so a different mode of question was adopted to ensure the respondents understood the concept of productivity for BRM. The initial design featured questions which seek to measure productivity per hour or day while the interviewees found it more applicable to define what time it takes to perform a single modification type. The resulting productivity scores from both interviews are reported in Table 17.

Table 17 - productivity scores results from both interviews

	Interviewee 1	Interviewee 2
Modification type 01: Create decision	One minute	Five to ten minutes
Modification type 02: Delete decision	Ten minutes	15 to 30 minutes
Modification type 03: Update decision	Two minutes	15 minutes
Modification type 04: Create business rule	Between five minutes and five days	Between five minutes and five days
Modification type 05: Delete business rule	15 minutes For each dependency add up 15 to 30 minutes	10 to 15 minutes With dependencies 60 minutes
Modification type 06: Create condition	Five to ten minutes	180 to 240 minutes Adding an existing condition (with existing cluster containing fact values) 60 to 120 minutes
Modification type 07: Delete condition	15 minutes	60 to 120 minutes With dependencies 180 to 240 minutes
Modification type 08: Update condition	15 minutes For each business rule document the condition label occurs add up 5 minutes	60 to 120 minutes
Modification type 09: Create fact value	Five minutes	Five to ten minutes
Modification type 10: Update fact value	Five minutes	Five to ten minutes For each dependency add five to ten minutes
Modification type 11: Delete fact value	Five minutes	Five to ten minutes For each dependency add five to ten minutes

Case-specific factors that influence productivity

Additionally, we extended the structured interview questions with space for open-ended elaboration on the provided productivity scores. Also, we included another question to determine factors that influence the processing of modifications to the business rule documents by the HSCIC business rules team. These findings are further elaborated in Appendix D.

4.2 ALMA for BRA outcomes

In this section we elaborate upon the implementation and outcomes from the four ALMA steps. The first step comprises the formulation of our goal. The second step comprises the creation of the architecture descriptions. The third step comprises the elicitation of scenarios, and finally, the fourth and last step comprises the evaluation of the scenarios by the implementation of Equation 3, and its results.

4.2.1 Step 1 – Goal formulation

The goal of study analysis is to determine which of the included architectural candidates scores best on effort prediction. Achieving this goal will provide insight into which architectural candidate requires the lowest effort, and therefore cost to maintain, what kind of modifications are needed to develop coming releases to the product and/or service, and where are the trouble spots in the system with respect to accommodating modifications.

4.2.2 Step 2 – Creating architecture descriptions

Once the goals are set, the architectures need to be described to be able to determine the impact of the modification scenarios in the following steps. For this research project we chose to create three different architecture descriptions of two clinical conditions within the QOF as described in section 2.6. This step is performed in three stages. The first stage comprises the construction of the clinical indicator COPD9 as part of the clinical condition COPD in all three architectural candidates. The second stage comprises the construction of all clinical indicators of the clinical condition COPD in all three architectural candidates. The third stage comprises repeating stage two, constructing all clinical indicators of a large clinical condition, Diabetes Mellitus, in all three architectural candidates. All three architecture descriptions are based on version 8, published on the 15th of March, 2006. In total, it took two months (January and February, 2015) to create all architecture descriptions of both included clinical conditions. As the results are validated each week we successfully completed eight cycles of validation of the architecture descriptions. Due practical issues, we included all architecture descriptions in Appendix E. Furthermore, to demonstrate examples of the architecture descriptions we included an example of a rule family-oriented BRA in Figure 19, an example of a fact-oriented BRA in Figure 20, and a decision-oriented BRA in Figure 21. All three figures represent the same part of business logic, that of the decision whether a patient is registered or not.

Architecture description construction

In this section we present the process of constructing the instantiated architectures in three stages. The architectures are constructed utilizing the three rule-oriented architectures as described in section 0. The first stage comprises the construction of a one indicator of the clinical condition COPD. For this first stage selected the first indicator, COPD 9, labelled as: *The percentage of patients with COPD in whom diagnosis has been confirmed by spirometry including reversibility testing*. The scope of the first stage is deliberately kept small to explore the representation and structure of each architectural candidate. The architecture representing the indicator COPD9 in all three architectural description languages can be found in Appendix E.

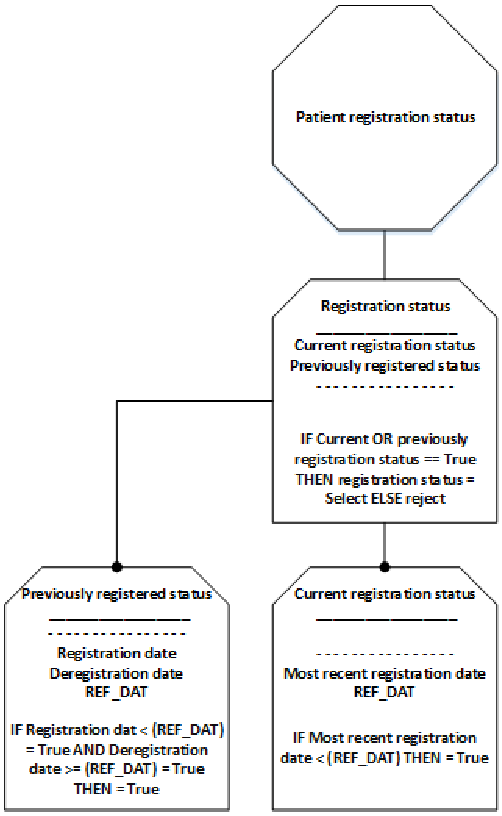


Figure 19- Example architecture description of the rule family-oriented architecture

The second stage comprises the construction of the full COPD case in all three architectural description languages. Instead of one single indicator as part of the clinical condition COPD we constructed the complete architecture with four indicators. The architecture representing the clinical condition COPD in all three architectural description languages can be found in Appendix E.

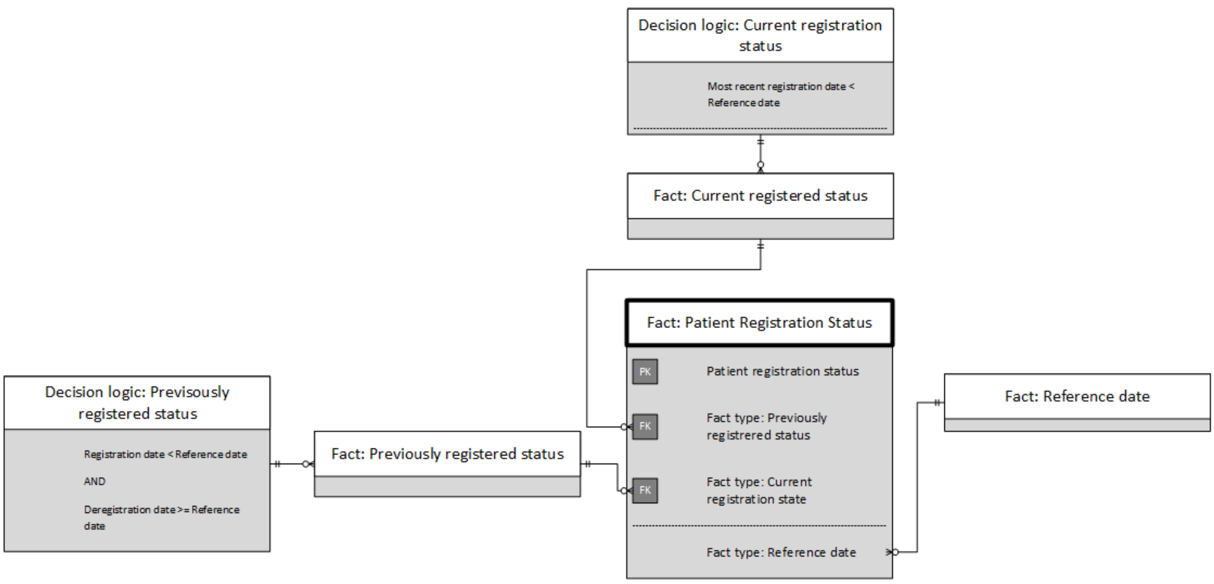


Figure 20 - Example architecture description of the fact-oriented architecture

For the third stage we scaled our production to create the full architecture for the clinical condition Diabetes Mellitus. This clinical condition is three times larger than the clinical condition COPD, including fifteen indicators, with larger amounts of underlying business rule sets containing more conditions as well. The architecture representing the clinical condition Diabetes Mellitus in all three architectural description languages can be found in Appendix E.

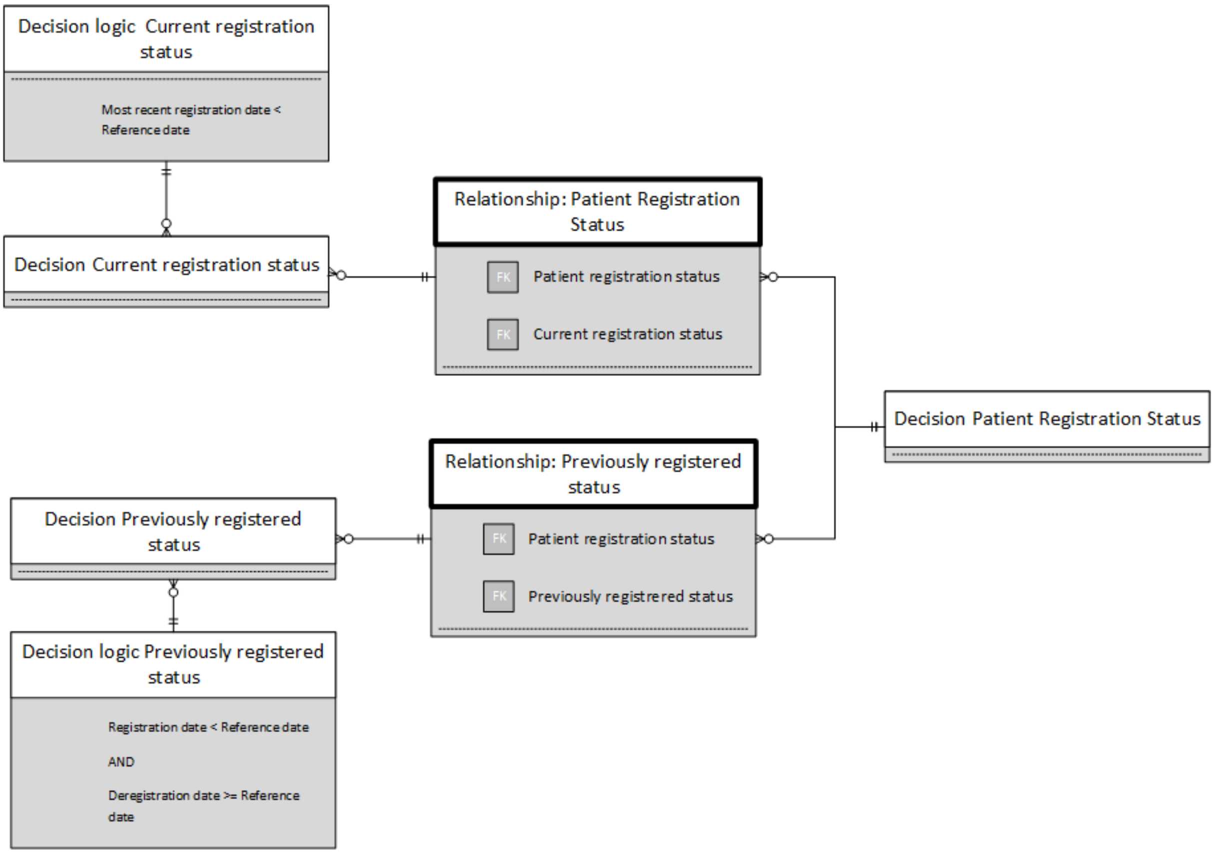


Figure 21 - Example architecture description of the decision-oriented architecture

The construction of the architectures was performed by the graduate student. However, the construction process was guided by two other researchers as described in chapter two. We managed to establish a series of weekly meetings that focused on validating the constructed architectures, which took around two months in total for all three stages combined. Due to space constraints we only include an example instantiation of one decision as modelled in the three architecture description languages as are depicted in Figure 19, Figure 20, and Figure 21.

4.2.3 Step 3 – Elicitation of scenarios

The third step comprises the elicitation of relevant scenarios. As described in the analysis chapter we utilized quantitative data to elicit relevant scenarios to be included in the ALMA equation. For the method the quantitative data is extracted and elicited we refer to section 2.4. The results of the extraction of data for scenario elicitation is included in Appendix F.

From the results of our top-down approach, as described in section 3.3.5, eleven relevant modification types were derived from earlier research results to include in the ALMA equation (Zoet et al., 2015). Each scenario elicited represents a unique modification type which should be taken into account while performing the analysis of modifiability. The complete set of scenarios comprises:

1. **The creation of a decision;**
2. **The deletion of a decision;**
3. **Updating a decision;**
4. **The creation of a business rule;**
5. **The deletion of a business rule;**
6. **The creation of a condition;**
7. **The deletion of a condition;**
8. **Updating a condition;**
9. **The creation of a fact value;**
10. **Updating a fact value;**
11. **The deletion of a fact value.**

As we have defined a set of scenarios we need to define the contents of each scenario as they need to be manually evaluated against the architectural descriptions in the next step. We set a minimum of five modifications per modification type for the small to moderate sized COPD case. An example is illustrated in Figure 22, whereas the full scenario set concerning the COPD case can be found in Appendix F.

ID	Modification type	Documents	Starting point	New situation
6	Create condition	V8 QOF -> V9 QOF	N.A.	MRC_COD
			N.A.	MRC_DAT
			N.A.	MRC1_COD
			N.A.	MRC1_DAT
			N.A.	OXYSAT_VAL
7	Delete condition	V8 QOF -> V9 QOF	COPD_COD	N.A.
			COPD_DAT	N.A.
			XFLU_COD	N.A.
			TXFLU_DAT	N.A.
			SPEX_COD	N.A.

Figure 22 - Example modifications in the modification scenarios create condition and delete condition for the COPD case

The second case, Diabetes Mellitus, has four times as many indicators with corresponding business rules. Therefore we set the minimum of twenty modifications per modification type for a large case. An example is illustrated in Figure 23, whereas the full scenario set concerning the Diabetes Mellitus case can be found in Appendix F.

The included modifications are semi-randomly created for each modification type based on historic modification data in combination with available documentation of the NHS containing the business rule documents. Concerning the process of modification selection for the creation of scenarios it is imperative that, while modifications are being selected semi-randomly, we slightly interfere with this process since we can estimate impact based on a few basic characteristics. Take for example the COPD case, indicator COPD8, which is the only indicator containing the concepts of TXFLU that relates specifically to that indicator. If any modification requires these business rules to be modified, little impact could be expected. This is different when we take a modification regarding a concept from which we know it is present in multiple indicators or even the whole COPD case. This implies that, relatively to the aforementioned modification, a high impact could be expected since all indicators should be modified. Due to this we included, next to randomly selected modifications, some modifications from which we know these will cause very little impact, and some that will cause a very large impact per modification.

ID	Modification type	Documents	Starting point	New situation
11	Delete Fact value	V8 QOF -> V9 QOF	1 - Patient disease status 73211009%	N.A.
			2- patient disease status 199223000%	N.A.
			3-patient disease status 49817004%)	N.A.
			4- DM_COD 6497000	N.A.
			5 - DM_COD 35425004	N.A.
			6- DM_COD 48499001	N.A.
			7- MALT_COD 19518008	N.A.
			8- MALT_COD 57378007,	N.A.
			9- PP_COD 268935007%	N.A.
			10- PP_COD 163111000)	N.A.
			11- PP_COD 268936008%	N.A.
			12- PP_COD 163123006	N.A.
			13- PP_COD 401221002	N.A.
			14-NPT_COD 390931008	N.A.
			15- NPT_COD 390932001	N.A.
			16- NPT_COD 394668001	N.A.
			17 NPT_COD 394667006	N.A.
			18- NPT_COD 394670005	N.A.
			19- NPT_COD 394669009	N.A.
			20- ACE_COD 318172002%	N.A.

Figure 23 - Example modifications in the modification scenario delete fact value for the Diabetes Mellitus case

4.2.4 Step 4 – Evaluation of scenarios

Now that the relevancy of the scenarios has been determined and realistically instantiated in the previous step the scenarios can be evaluated against the selected architectural candidates. The fourth step of the ALMA-method for application in the BRM domain involves 1) *the scenario size determination*, 2) *the normalized weight determination*, 3) *the productivity level per BRM modification type*, and 4) *the determination of the estimated maintenance effort*.

4.2.4.1 Instance-level analysis of impact per scenario (size)

The evaluation of the scenarios defined in the previous step involves the process of simulating the scenario while analysing and reporting on the impact of the scenario. This process is performed by manually searching for the modified element corresponding to the modification type in the architectural candidate. Then the impact is determined by counting the modified elements in the architectural candidate. An example of this is depicted in Figure 24. For the sake of this example we include two modification types. The first modification type and its corresponding impact is that of the Update Condition type, where the condition COPDEXC_DAT is updated to COPDEXEPTION_DAT. Analysis of the modified element shows that the modification affects multiple indicators, as the element is used in both the denominators COPD 8 and 10. The impact of this modification in the context of Figure 24 is therefore four rule families, which is represented by the letter A in the red boxes. For the second modification type we select the same modification type, but another instantiation with deviating impact from the first example. In the second modification type the condition TXFLU_COD is updated to TXFLUENZA_COD. However, since the condition is not used in multiple indicators, thus is indicator-specific, the impact is smaller than the first modification type included in this example, which is represented by the letter B in the blue boxes. Now that the modification type and its impact is determined the size of the modification needs to be calculated. As described in section 3.4.2 we utilize the measurement scale of function points instead of lines of code. We now analyse, based on the theoretical framework of Felfernig & Salbrechter (Felfernig & Salbrechter, 2004) as depicted in Table 18 and Table 19, the impact of the modification types per logical function. In our example context, instantiated with the TDM notation, each rule family represents one logical function. As we determine the impact for each scenario representing one single modification type we can utilize our adjusted function points per impacted logical function. For example, in the context of Figure 24 our scenario concerns the conditions highlighted by the red frames, being COPDEXC_DAT, to be updated. To calculate the amount of impact in function points we simply sum the amount of logical functions impacted and multiply this by the amount of function points per modification type as defined in Appendix H. For our example we count a total of four logical functions to be affected by the modification type, times the amount of function points per the modification type update condition (0.12907069) results into a total impact of 0.52 function point.

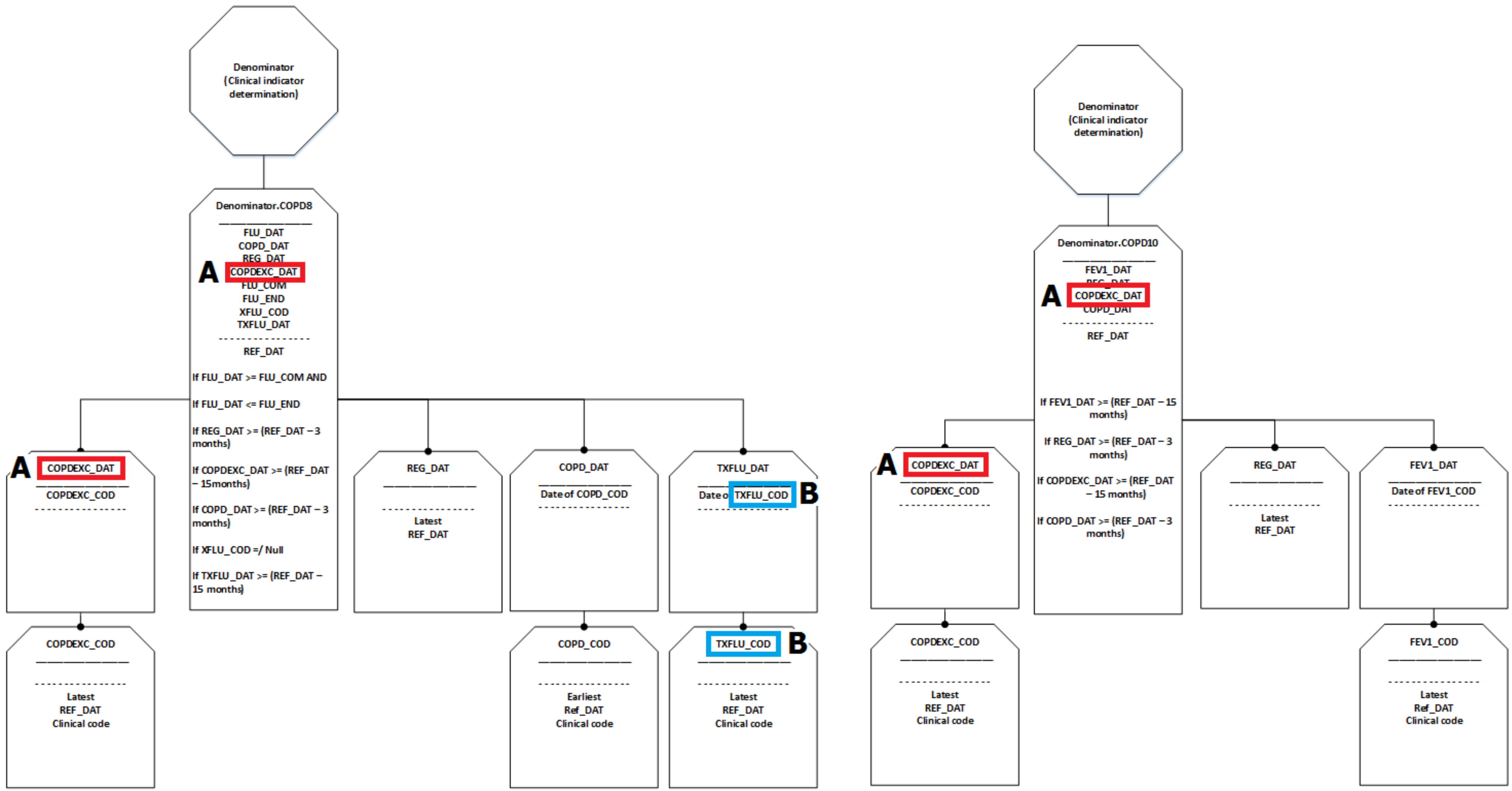


Figure 24 - Example of the size and impact determination process

Scenario size results

In this section we provide the results of the determination of scenario impact and size per architectural candidate. The results are presented per clinical indicator. The results of the first clinical condition, COPD, are presented in Table 18. The results of the second clinical condition, Diabetes Mellitus, are presented in Table 19. The base calculations for the determination of function points per logical function are presented in Appendix H.

Table 18 - Scenario impact and size analysis results for the clinical condition COPD

		Modification impact in LF			Size in FP		
Scenario ID	Scenario label	RFO	FO	DO	RFO	FO	DO
1	CD	5	10	10	0.02813	0.05626	0.05626
2	DD	8	17	25	0.17209	0.36570	0.53779
3	RD	8	10	54	0.09002	0.11252	0.60763
4	CBR	5	5	5	7.94281	7.94281	7.94281
5	DBR	5	5	5	0.16878	0.16878	0.16878
6	CC	11	20	25	1.58724	2.88589	3.60736
7	RC	24	52	55	4.76569	10.32566	10.92137
8	DC	39	39	45	5.03376	5.03376	5.80818
9	CFV	16	12	8	0.13238	0.09929	0.06619
10	UFV	18	13	8	0.14893	0.10756	0.06619
11	DFV	16	11	8	0.13238	0.09101	0.06619

Table 19 - Scenario impact and size analysis results for the clinical condition Diabetes Mellitus

		Modification impact in LF			Size in FP		
Scenario ID	Scenario label	RFO	FO	DO	RFO	FO	DO
1	CD	40	40	40	0.22505	0.22505	0.22505
2	DD	36	96	164	0.77442	2.06513	3.52793
3	RD	36	96	164	0.40508	1.08022	1.84538
4	CBR	20	20	20	31.77125	31.77125	31.77125
5	DBR	20	20	20	0.67514	0.67514	0.67514
6	CC	52	77	96	7.50331	11.11067	13.85226
7	RC	358	662	675	71.08817	131.45353	134.03495
8	DC	358	485	422	46.20731	62.59929	54.46783
9	CFV	125	74	62	1.03422	0.61226	0.51297
10	UFV	127	104	90	1.05077	0.86047	0.74464
11	DFV	130	72	62	1.07559	0.59571	0.51297

Based on our method of impact analysis as described in previous section we first determined the impact of the scenarios per architectural candidate as presented in the third, fourth and fifth columns. Furthermore, based on the size in FP that is required to modify each logical function we calculated the size in FP for each scenario, presented in the sixth, seventh, and eight column. Take for example the eight scenario labelled as DC (Delete Condition) as presented in Table 18. The modifications in that particular scenario result in the required modification of 39 logical functions in both the rule family-oriented and fact-oriented BRA's, while the third BRA, the decision-oriented BRA requires the modification of 35 logical functions. Based on these values we calculate the size in FP by multiplying the modification impact in LF by its base value of a rounded 0.2 FP/logical function. The results of this calculation shown in columns six, seven, and eight shows that the decision-oriented architecture scores worst compared to the other two architectural candidate.

4.2.4.2 Normalized weight determination

Based on the descriptive statistics derived from historic modification data presented in Table 10 and Table 11 in section 2.6 we calculate the normalized weights for each scenario by dividing the number of modifications of the scenario by the total number of modifications of all included scenarios. The results of the weight normalization are presented in Table 20 concerning the clinical indicator COPD and Table 21 for the clinical indicator Diabetes Mellitus. The normalized weights as presented in the tables are calculated utilizing Equation 1.

Table 20 - Normalized weight determination for the scenarios of the clinical case COPD

Clinical indicator: COPD										
CD	DD	UD	CBR	DBR	CC	DC	UC	CFV	UFV	DFV
6	0	17	10	1	4	3	69	73	67	49
Normalized weights										
0.020	0	0.057	0.033	0.003	0.013	0.010	0.231	0.244	0.224	0.164
Total MO's		299								

Table 21 - Normalized weight determination for the scenarios of the clinical case Diabetes Mellitus

Clinical indicator: Diabetes Mellitus										
CD	DD	UD	CBR	DBR	CC	DC	UC	CFV	UFV	DFV
51	58	3	221	34	2	169	100	175	103	21
Normalized weights										
0.054	0.062	0.003	0.236	0.036	0.002	0.180	0.107	0.187	0.110	0.022
Total MO's		937								

Based on the outcomes of the normalized weight calculations we can analyse, on a scale from zero to one, how important a given scenario is. The values presented act as a priority mechanism in ALMA's equation as they determine the weight of the size as elaborated upon in the previous section. For example, we compare from Table 21 the normalized weight for the scenario CBR (Create Business Rule) with the normalized weight for the scenario DVF (Delete Fact Value). The scenario CBR occurred 221 times of the total 937 modified coded in the past eight years. This results in a value of 0.236, or simply stated 23,6% of the modifications coded are that of the type CBR. The scenario DVF has a significantly lower amount of modifications coded; 21 occurrences, resulting in a share of 2,2% of the total modifications coded.

4.2.4.3 Productivity analysis results

Based on the qualitative interviews we identified the productivity scores for each modification type. The interview results are provided and elaborated upon in section 4.1. In this section we transform the derived interview results into variables (productivity levels) for utilization in Equation 3.

The first assumption that has to be made is that of the theoretical maximum capacity per employee per month for one Full Time Employee (FTE). We assumed 22 working days on average per month, excluding potential public holidays. Furthermore, we assumed a working day of nine hours, including one hour break time. Therefore we use the baseline of 176 operational hours which equals 10,560 operational minutes of theoretical maximum capacity per month. The conversion of hours to minutes scale is performed to calculate the amount of modification types that can be performed per month, which is expressed in minutes per single modification type. The average productivity scores are provided in Table 22.

Table 22 – Average productivity score per modification type

Modification type	Average time per MT	MT's/Month
CD	4.25	2484
DD	16.25	649
UD	8.5	1242
CBR	1200	8.8
DBR	25.5	414
CC	109	96
DC	150	70
UC	97.5	108
CFV	6.25	1689
UFV	6.25	1689
DFV	6.25	1689

Furthermore, the worst and best case productivity levels are presented in Table 23. The worst case productivity scores are derived from interviewee two, while the best case productivity score is derived from interviewee one, as described in section 4.1. Similarly as the average productivity score as presented in Table 22, the values in the second and fourth columns are expressed in minutes.

Table 23 - Worst and best case productivity levels per modification type

Modification type	Worst case average time per M	M's/Month	Best case average time per M	M's/Month
CD	7.5	1408	1	10560
DD	22.5	469	10	1056
UD	15	704	2	5280
CBR	1200	9	1200	9
DBR	60	176	15	704
CC	210	50	7.5	1408
DC	150	70	15	704
UC	90	117	15	704
CFV	7.5	1408	5	2112
UFV	7.5	1408	5	2112
DFV	7.5	1408	5	2112

4.2.4.4 ALMA effort calculation results

The previous sections all presented results which enable us to perform ALMA's equation as presented in Equation 3 to determine the total effort for each architectural candidate. At this stage the following dimensions are known: 1) *Impact and size of the scenarios*, 2) *normalized weights of the scenarios*, 3) *productivity scores per modification type*, and 4) *the modification estimation for a given amount of time based on historic modification data*. Based on the amount of scenarios in the scenario profile, the C(MP) dimension is determined. Since we include eleven modification types we use this number for the scenario profile. All results presented in this section are expressed in hours per QOF period. The QOF is republished two times a year, once per six months.

In Figure 25 we summarize the ALMA results for the three included architectural candidates based on the clinical condition COPD. The full sheet with underlying calculations are presented in Appendix I. Results shows that, concerning the COPD case, the RFO architecture scores best on effort prediction. Furthermore, the FO architecture shows a 2.8% of additional effort required, while the DO architecture shows an additional 16% of additional effort required to process the set of scenarios provided.

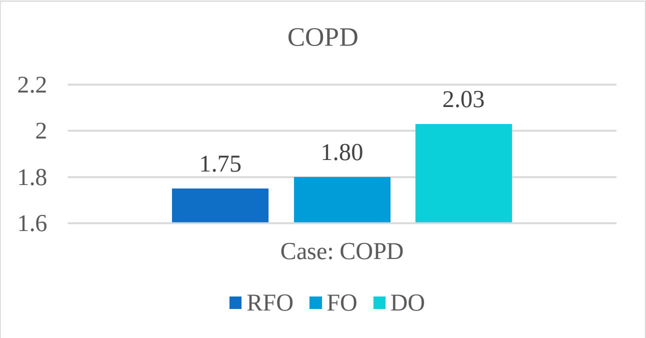


Figure 25 – ALMA effort calculation in hours concerning the COPD case

In Figure 26 we summarize the results from ALMA for the three selected architectural candidates based on the clinical condition Diabetes Mellitus. The full sheet with underlying calculations are presented in Appendix I. Results shows that, concerning the DM case, the RFO architecture scores best on effort prediction. Furthermore, the FO architecture shows a 49.14% of additional effort required, while the DO architecture shows an additional 47.81% of additional effort required to process the set of scenarios provided.

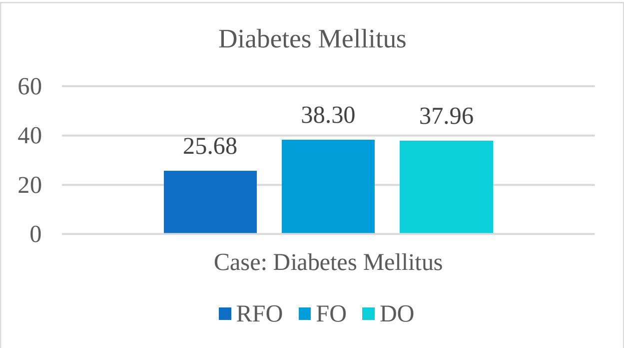


Figure 26 - ALMA effort calculation in hours concerning the Diabetes Mellitus case

Furthermore, we utilized the available data to discover the theoretical best and worst case outcomes regarding total effort. To be able to do so we stopped utilizing the averages from both interviewees concerning the effort in minutes per modification type, see section 4.2.4.3. Both interviewees differ in experience level, where interviewee one has four years of experience working with the QOF, while interviewee two has one year of experience working with the QOF. We report on the less experienced employee as the theoretical worst case effort calculation, while the more experienced employee is reported as theoretical best case effort calculation. The results for the COPD case are presented in Figure 27.

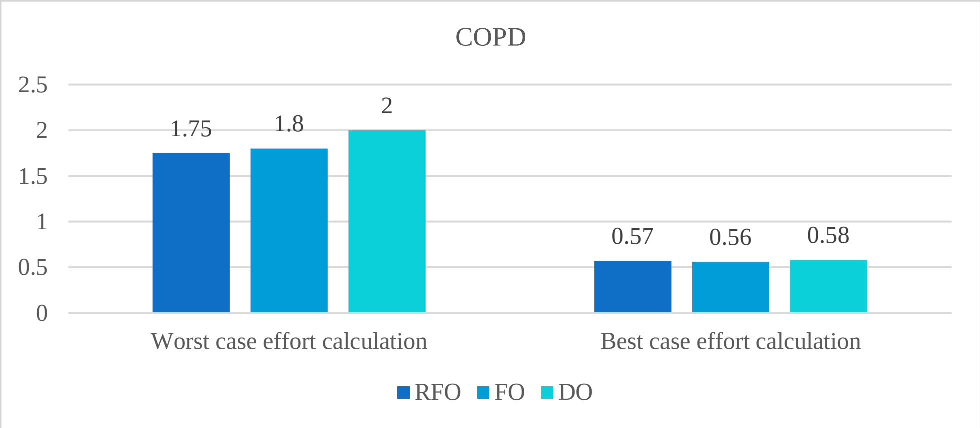


Figure 27 – ALMA worst case versus best case effort calculation concerning the COPD case

The results presented in Figure 27 shows that a worst case approach is very similar to the average effort calculation as presented earlier. The best case approach shows some difference, resulting just more than half an hour of effort predicted for each architectural candidate. The full sheet with underlying calculations are presented in Appendix I.

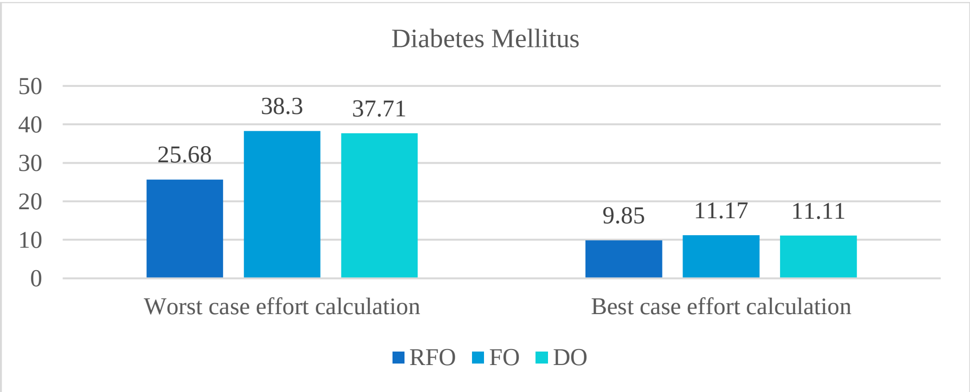


Figure 28 - ALMA worst case versus best case effort calculation concerning the Diabetes Mellitus case

The results for the second case, Diabetes Mellitus as presented in Figure 28 shows similar results. Results shows that the RFO architecture scores best on both worst and best case effort prediction. The worst case architecture shows similar percentage differences compared to the average effort calculation and is therefore not further elaborated upon. The best case effort prediction also resulted in the best performance by the RFO architecture, while the FO architecture shows a 13.38% of additional effort required and the DO architecture shows an additional 12.79% of additional effort required. The full sheet with underlying calculations are presented in Appendix I. Summarized, Table 24 presents the numeric results of the analysis outcomes with both the predicted effort as well as its corresponding percentage differences.

Table 24 - Summary of results of the effort prediction utilizing ALMA

COPD	**RFO**	**FO**	**DO**
Effort predicted (H)	1.75	1.80	2.03
Difference in effort (H)	N.A.	0.05	0.28
Difference in %	N.A.	2.8%	16%
DM	**RFO**	**FO**	**DO**
Effort predicted (H)	25.68	38.30	37.96
Difference in effort (H)	N.A.	12.62	12.28
Difference in %	N.A.	49.14%	47.81%
COPD – Worst case	**RFO**	**FO**	**DO**
Effort predicted (H)	1.75	1.80	2
Difference in effort (H)	N.A.	0.05	0.25
Difference in %	N.A.	2.85%	14.28%
COPD – Best case	**RFO**	**FO**	**DO**
Effort predicted (H)	0.57	0.56	0.58
Difference in effort (H)	N.A.	- 0.01	0.01
Difference in %	N.A.	- 1.78%	1.75%
DM – Worst case	**RFO**	**FO**	**DO**
Effort predicted (H)	25.68	38.30	37.71
Difference in effort (H)	N.A.	12.62	12.03
Difference in %	N.A.	49.14%	46.84%
DM – Best case	**RFO**	**FO**	**DO**
Effort predicted (H)	9.85	11.17	11.11
Difference in effort (H)	N.A.	1.32	1.26
Difference in %	N.A.	13.38%	12.79%

In the following chapters we interpret and conclude upon the outcomes of the evaluation of the included architectural candidates, covering the fourth step of the ALMA-method. Furthermore, we discuss the limitations and future research opportunities of this study.

5 Conclusion

Currently, many organizations employing large amounts of business rules as part of their products and/or services to deliver added value to their customers do not take into account the cohesion between those business rules, which makes it practically impossible to perform proper impact analysis before modifications are implemented. Furthermore, these products and/or services containing business rules are generally not designed to cope with modifiability. This seems very counterintuitive as previous research already proved that business logic, and therefore business rules, are characterized by the highest modification frequency in combination with the highest amount of effort required to perform the modifications when compared to other aspects of IS development and maintenance. To address this problem we aimed to answer the following main research question:

How can Business Rule Architectures be designed for modifiability, taking into account the concept of anticipated modifications?

In this research project the goal was to explore the concept of logical architectures for utilization in the BRM domain, with a focus on the modifiability aspect. In order to achieve this, we explored rule-oriented architectures to study the architectural modifiability based on the Architecture-Level Modifiability Analysis method adopted and adapted from the software architecture domain for the BRM domain. This was followed by the instantiation of the three selected rule-oriented architectures based on available case-specific data from the British National Health Service (NHS), concerning the clinical condition's COPD and Diabetes Mellitus. To be able to perform the analysis of modifiability more case-specific data is collected utilizing quantitative descriptive statistics based on large amounts of available historic data and qualitative interviews with BRM subject-matter experts that employed at the HSCIC, which provides the information, data and IT systems for the NHS. Furthermore, we created representative scenarios that serves as important input as well for the analysis of modifiability. The last step in this research project consisted of combining those results for further analysis by the equation utilized by the ALMA-method, evaluating the included architectural candidates.

To determine how Business Rule Architectures can be designed for modifiability we interpret the results of the analysis performed as reported upon in the previous chapter. In the analysis we examined three perspectives: 1) *the average effort prediction based on average productivity scores*, 2) *the worst case effort prediction based on the lower productivity scores*, and 3) *the best case effort prediction based on the higher productivity scores*. Concerning the first perspective we revealed that little difference in effort prediction exists between all three included architectural candidates for the smaller COPD case. When analysing the larger Diabetes Mellitus case, our findings suggest lower effort prediction for the rule family-oriented architecture compared to the other two included architectural candidates. Further analysis of the differences between the fact-oriented and decision-oriented architectures leads us to a difference in impact, between the COPD and Diabetes Mellitus cases. Results show that the impact of the fact-oriented architecture increases as the case size increases. However, results also shows the opposite for the decision-oriented architecture, which suggest that whenever the case size increases, less impact is measured. The difference between both architectural candidates is caused by redundancy of data which increases when the case grows, caused by elements that are reused for other decisions in the same case. This implies that the modifiability of the fact-oriented architecture decreases when the case size increases. This is caused by the star-schemed structures that do not relate to each other with relationships. This results in an increase in creation of redundant conditions and underlying fact values as the case size increases. On the other hand, the opposite for the decision-oriented architecture is true due to the utilization of links between decisions which creates the opportunity to reuse, for example, conditions and underlying fact values for other decisions. This results in less impact, thus higher modifiability of the architecture. A similar structure is adhered to by the rule family-oriented architecture, which relates architectural elements without creating redundant conditions and conclusions. Furthermore, the results suggest that the difference between the rule family-oriented architecture and the other two architectural candidates is caused by further separation of decision logic by both the fact-oriented and decision-oriented architectures, while this is not adhered to by the rule family-oriented architecture. This means less impact is calculated due to the impact on less individual logical functions in the architectural candidate. Lastly, results of the worst and best case perspectives reveal a similar outcome to that of the perspective which takes into account the average productivity scores. Although this research was carefully prepared and performed, we are aware of several research limitations and shortcomings which are elaborated upon in Section 5.1. Furthermore, the findings of this research project also provides several opportunities for further research, which are elaborated upon in section 5.2.

5.1 Discussion

This research aimed to 1) *explore possible rule-oriented architectures*, 2) *create architectures concerning the selected case organization*, the NHS, and 3) *adapt and perform the analysis of modifiability over these architectures utilizing ALMA*. We believe that the derived results and insights on the adaptation of existing theories and methods for neighbouring fields are a first step in conceptualizing the notion of a business rule architecture and laying a foundation for future research on the topic, as well as partly maturing the BRM research field in general. However, we believe certain aspects of this research project are susceptible to discussion.

One of the main limitations of this research project is ALMA's dependency on productivity levels of the case organization. Our selected case organization employs two experts that process business rules. Whether this amount is too low is debatable, but looking at the case organization we could not include more interviewees as this comprises the whole population. The interviewees stated that estimating the prompted productivity scores is a hard task and involves estimating averages which potentially suffer from inaccuracy. The derived figures from both experts are arithmetically calculated to an average and utilized in the prediction model as the productivity level of the selected case organization. Also, we saw the interviews as a good opportunity to derive qualitative factors which potentially or are known to affect productivity of the performing certain modification types. As we identified such factors we could argue that most but not all of those factors are taken into account when performing the analysis of modifiability. However, we believe that our efforts paid off in exploring possibilities to perform architectural analysis of modifiability for logical architectures utilized in the BRM domain.

Furthermore, the conversion rate of hours per function point is fully adopted from the software engineering domain. This particular decision potentially affects generalizability to the BRM domain. Also, literature suggest that such conversion rates are partly dependent on domain-specific characteristics like team productivity, experience, support, programming language, and project size. Such characteristics are not fully explored during this research, probably affecting accuracy of the conversion rate adopted. Again, we believe that exploratory studies such as this one not need to fully focus on deriving such accurate results first. Such concepts need to be researched domain-specific when the method itself has proven to function for the domain proposed as demonstrated in this research project.

Another limitation which affects precision of the results of this research project is that we assumed theoretical working months, counting 22 working days in a moth, each day counting for eight hours of productivity. It is unlikely that employees are fully productive for two periods of four hours a day in a row. However, it was beyond the scope of this research study to fully explore this as well for the case organization, as well as such information is hard to measure other than observing and measuring productivity.

The fact that this research project solely focuses on the modifiability aspect of logical architectures decreases generalizability to products and/or services as these are usually designed keeping in mind several quality attributes instead of only that of modifiability. Considering the large amounts of data processed by the QOF business rule sets we could expect that logical architectures also need to take into account quality attributes like performance and compatibility. We believe that focusing on the trade-off between those quality attributes before explorative research is conducted on the individual topics is rather unwise since much concepts are unexplored and potentially will affect research results.

Our last limitation regards the exploratory nature of our activities concerning the construction of the architecture descriptions and the manual analysis of impact on the included architectural candidates based on the defined scenarios. Such working methods were experienced sufficient in accuracy, but are inefficient in terms of effort. Although this research was performed by the master student in collaboration with two BRM experts, the method of working as described above is prone to errors.

5.2 Future research

This thesis presents a first exploration of the Business Rule Architecture (BRA) concept, with a focus on effort prediction and modifiability of the architectural structures utilized by the explored BRA's. The gathered data and insights as a result of this research project provides us with the means to derive several potential future research directions.

As is identified by the results of the literature analysis the current body of knowledge concerning the BRM domain, specifically on the concept of a logical business rule architecture is very limited. Development of solutions to support businesses working with large cohesive sets of business rules could be sparked more by further research into the concept of logical architectures for the BRM domain. We believe that our work is an initial addition to the current body of knowledge concerning BRA's and their application in practice and we argue that the results form a grounded basis for future research.

The focus of this particular research project lies entirely on the modifiability aspect of logical architectures. It is desirable to conduct more research on other, relevant, quality attributes that affect working with business rules and logical architectures to create products and/or services. Furthermore, similarly to research conducted in the software architecture domain, further research could focus on the trade-off relationship between different quality attributes, since it is known that these affect one another in various ways.

The method to assess to what extent an architectural candidate is modifiable that is utilized in this research project is one of many other methods to perform analysis of modifiability. Further research could focus on developing an adaptation of another modifiability analysis method known from research and practice as also identified in this research project. Also, focusing on the results of different methods of analysis available in literature, while discovering similarities and differences will lead to increased knowledge that could trigger the development of an even more mature method, further specified for utilization in the BRM domain. Furthermore, following studies could include other cases than the ones included in this particular research project, as increasing sample size will improve generalizability. Future research should focus on; 1) *increase the amount of cases* (clinical conditions), 2) *include multiple other organisations from the same* (healthcare) industry, and 3) *involve other industries*.

Similarly, the quest towards finding the most optimal modifiability could be expanded by performing the same method of analysis over architectural candidates other than included in this research project. We believe that there is a variety of architectural candidates left to analyse for modifiability in future studies. This will eventually create potential to construct a 'best of breed' architecture including extremely modifiable components and structures for application in the BRM domain.

As our results presented, depending on the size of the case analysed the decision whether to utilize a certain architectural candidate as included in our research can be questioned. Future research should focus on the identification of confounding factors that influence the design decision whether to adhere to a certain architectural standard. Possible confounding factors could be the size of the case, the case complexity, and/or the method of analysis utilized. When these are identified and made explicit the turning point whether to utilize architectural candidate A, B, or C can be made explicit.

Although our case organization and data specify the use of CRUD-related modification types further research could focus on organizations which are allowed to solely create new architectural elements due to compliancy. Such organizations are forced to accommodate and maintain multiple instantiations of architectures which contain relatively similar content, but are valid on different timeframes. Examples of such organizations are the Dutch Tax Administration, Immigration and Naturalization Service, and the Dutch social security agency. Experts argue that such contingency factors will greatly affect several variables utilized for analysis of modifiability as performed in this research project.

A factor which is deemed important in the world of impact analysis concerning the software domain is the cost of change. The method utilized and the results provided by this research project could fuel further research into cost prediction derived from effort calculation. Such figures could potentially support decision making concerning planned modifications, or as input to further calculate the most cost-effective way to modify products and/or services.

Furthermore, the artefacts created in this research project could benefit more from further validation by researchers and practitioners. Further research should focus on further external validation by experts on architectures in the IS domain, experts on BRM, BRM practitioners, etc. Input from the combined knowledge of such respondent groups are highly valuable at the current nascent maturity rate of the BRM domain.

As we learned in this study our exploratory approach is sufficient in accuracy but proved to be inefficient in terms of effort. Future studies focusing on involving more cases regarding the NHS or other industries should take into account tools to support researchers with creating architectural descriptions in such a way that 1) *it is less error-prone* (increasing accuracy) and 2) *it supports processing large and complex amounts of business logic as part of an architecture* (increasing efficiency).

Lastly, this research included a sample size of two to determine the productivity levels needed as input for the analysis of modifiability. As described in the limitations, this sample size is very low. The accuracy of the utilized productivity levels could benefit from further research which includes larger sample sizes concerning the measurement of productivity levels to generalize towards a more general view of productivity in the BRM domain.

6 References

Abran, A., Gil, B., & Lefebvre, É. (2004). Estimation models based on functional profiles. In *Proceedings of the International Workshop on Software Measurement–IWSM/MetriKon*. Kronisburg, Germany.

Abran, A., & Robillard, P. N. (1996). Function points analysis: An empirical study of its measurement processes. *IEEE Transactions on Software Engineering*, *22*(12), 895 – 910.

Albrecht, A. J., & Gaffney, J. E. (1983). Software function, source lines of code, and development effort prediction: a software science validation. *IEEE Transactions on Software Engineering*, *6*, 639 – 648.

Alexander, A. (2008). FunctionPoints. Retrieved January 26, 2015, from http://alvinalexander.com/FunctionPoints/node32.shtml

Algemene Rekenkamer. (2013). Complexe of oncontroleerbare regels werken fraude soms in de hand. Retrieved from http://verantwoordingsonderzoek.rekenkamer.nl/2013/rijksbreed/themas/fraude/complexe-oncontroleerbare-regels-werken-fraude-soms-de-hand

Allen, T., Mason, T., & Whittaker, W. (2014). Impacts of pay for performance on the quality of primary care. *Risk Management and Healthcare Policy*, *7*, 113–20. http://doi.org/10.2147/RMHP.S46423

Andersson, J. (2002). *Enterprise Information Systems Management - An Engineering Perspective Focusing on the Aspects of Time and Modifiability*. KTH, Royal Institute of Technology.

ANP. (2014). Belastingen doorgeschoten en te complex. Retrieved June 14, 2010, from http://www.parool.nl/parool/nl/30/ECONOMIE/article/detail/3633597/2014/04/11/Belastingen-doorgeschoten-en-te-complex.dhtml

Babar, M. a., & Gorton, I. (2004). Comparison of scenario-based software architecture evaluation methods. In *Proceedings of the 11th Asia-Pacific Software Engineering Conference* (pp. 600–607). IEEE. http://doi.org/10.1109/APSEC.2004.38

Bajec, M., & Krisper, M. (2005). A methodology and tool support for managing business rules in organisations. *Information Systems*, *30*(6), 423–443. http://doi.org/10.1016/j.is.2004.05.003

Barbacci, M., Klein, M. H., Longstaff, T. A., & Weinstock, C. B. (1995). Quality Attributes - CMU/SEI-95-TR-021 ESC-TR-95-021. Pittsburgh, Pennsylvania: Software Engineering Institute.

Barber, K. S., Graser, T., & Holt, J. (2002). Enabling iterative software architecture derivation using early non-functional property evaluation. In *Proceedings of the 17th IEEE International Conference on Automated Software Engineering* (pp. 172 – 182). IEEE.

Bass, L., Clements, P., & Kazman, R. (2012). *Software Architecture in Practice Third Edition*. (Software Engineering Institute, Ed.) (3rd ed.). Addison-Wesley.

Bass, L., Klein, M., & Bachmann, F. (2000). Quality attribute design primitives - (No. CMU/SEI-2000-TN-017). Pittsburgh, Pennsylvania: Carnegie-Mellon University Pittsburg.

Bengtsson, P. (2002). *Architecture-Level Modifiability Analysis*. Blekinge Institute of Technology.

Bengtsson, P., Lassing, N., Bosch, J., & Van Vliet, H. (2004). Architecture-level modifiability analysis (ALMA). *Journal of Systems and Software*, *69*(1-2), 129–147. http://doi.org/10.1016/S0164-1212(03)00080-3

Bennett, K. H., & Rajlich, V. T. (2000). Software maintenance and evolution: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering* (pp. 73 – 87). ACM.

Berander, P., Damm, L. O., Eriksson, J., Gorschek, T., Henningsson, K., Jönsson, P., & Tomaszewski, P. (2005). *Software quality attributes and trade-offs*.

Biletskiy, Y., & Ranganathan, G. R. (2008). An invertebrate semantic/software application development framework for knowledge-based systems. *Knowledge-Based Systems*, *21*(5), 371 – 376.

Blankena, F. (2012). Hoe je regels uitvoerbaar maakt. Retrieved from http://ibestuur.nl/academie/mc-succesvolle-ict-projecten-verslag-avond1-Hoe-je-regels-uitvoerbaar-maakt

Blenko, M. W., Mankins, M. C., & Rogers, P. (2010, June). The Decision-Driven Organization. *Harvard Business Review*. Retrieved from https://hbr.org/2010/06/the-decision-driven-organization

Blomgren, M., & Sundén, E. (2008). Constructing a European healthcare market: The private healthcare company Capio and the strategic aspect of the drive for transparency. *Social Science & Medicine*, *67*(10), 1512 – 1520.

Blueriq. (2015). Speel sneller in op veranderingen. Retrieved January 20, 2015, from https://www.blueriq.com/oplossingen/decision-management/

Boehm, B., & In, H. (1996). Identifying quality-requirement conflicts. *IEEE Software*, *13*(2), 25 – 35.

Boehm, B. W., Brown, J. R., Kaspar, H., Lipow, M., McLeod, G., & Merritt, M. (1978). *Characteristics of Software Quality*. Amsterdam: North-Holland Publishing Company.

Bosch, J. (2000). *Design and use of software architectures: adopting and evolving a product-line approach*. Pearson Education.

Boyer, J., & Mili, H. (2011). Agile business rule development. *Agile Business Rule Development*, 49–71. http://doi.org/10.1007/978-3-642-19041-4

Breivold, H. P., Crnkovic, I., & Eriksson, P. J. (2008). Analyzing software evolvability. *Computer Software and Applications*, *32*, 227 – 330.

Brinkkemper, S., Saeki, M., & Harmsen, F. (1999). Meta-modelling based assembly techniques for situational method engineering. *Information Systems*, *24*(3), 209 – 228.

British Medical Association. (2015). QOF payments. Retrieved January 16, 2015, from http://bma.org.uk/practical-support-at-work/contracts/gp-contracts-and-funding/independent-contractors/qof-guidance/qof-payments

Bundschuh, M., & Dekkers, C. (2008). *The IT measurement compendium: estimating and benchmarking success with functional size measurement*. Springer Science & Business Media.

Business Rules Group. (2003). *The Business Rules Manifesto*. Retrieved from http://www.businessrulesgroup.org/brmanifesto/BRManifesto.pdf

Carpers, J. (1996). *Applied Software Measurement: Assuring Productivity and Quality*. New York: McGraw-Hill, New York.

Chapin, N., Hale, J. E., Khan, K. M., Ramil, J. F., & Tan, W. G. (2001). Types of software evolution and software maintenance. *Journal of Software Maintenance and Evolution: Research and Practice*, *13*(1), 3 – 30.

Charfi, A., & Mezini, M. (2004). Hybrid web service composition: business processes meet business rules. In *Proceedings of the 2nd international conference on Service oriented computing ICSOC 04* (pp. 30–38). ACM. http://doi.org/10.1145/1035167.1035173

Chaudhuri, S., & Dayal, U. (1997). An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, *26*(1), 65 – 74. http://doi.org/10.1145/248603.248616

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, *13*(6), 377 – 387.

Curti, C., Ferrari, T., Gommans, L., Van Oudenaarde, S., Ronchieri, E., Giacomini, F., & Vistoli, C. (2005). On advance reservation of heterogeneous network paths. *Future Generation Computer Systems*, *21*(4), 525 – 538.

Darke, P., & Shanks, G. (1996). Stakeholder viewpoints in requirements definition: A framework for understanding viewpoint development approaches. *Requirements Engineering*, *1*(2), 88–105. http://doi.org/10.1007/BF01235904

Department of Health. (1998). *A First Class Service. Quality in the New NHS.* Retrieved from http://webarchive.nationalarchives.gov.uk/+/www.dh.gov.uk/en/publicationsandstatistics/publications/publicationspolicyandguidance/dh_4006902

Department of Health England. (2013). *Guide to the Healthcare System in England*. Retrieved from https://www.gov.uk/government/publications/guide-to-the-healthcare-system-in-england

Dienst Uitvoering Onderwijs. (2014). How does student finance work? Retrieved from http://www.duo.nl/particulieren/international-student/student-finance/how-does-it-work.asp

Dijkstra, E. W. (1974). On the role of scientific thought. In *Selected writings on computing: a personal perspective* (pp. 60 – 66). Springer New York.

Dobrica, L., & Niemela, E. (2002). A survey on software architecture analysis methods. *IEEE Transactions on Software Engineering*, *28*(7), 638 – 653.

Eakin, J. M., & Mykhalovskiy, E. (2003). Reframing the evaluation of qualitative health research: reflections on a review of appraisal guidelines in the health sciences. *Journal of Evaluation in Clinical Practice*, *9*(2), 187 – 194.

Edmondson, A. C., & Mcmanus, S. E. (2007). Methodological Fit in Management Field Research. *Proceedings of the Academy of Management*, *32*(4), 1155–1179.

E-overheid. (2014). Nederlandse Overheids Referentie Architectuur.

Felfernig, A., & Salbrechter, A. (2004). Applying function point analysis to effort estimation in configurator development. In *Proceedings of the International Conference on Economic, Technical and organisational aspects of Product Configuration Systems* (pp. 109 – 119).

Gall, M. D., Borg, W. R., & Gall, J. P. (1996). *Educational research: An introduction*. Longman Publishing.

General Medical Council. (2015). List of Registered Medical Practitioners - statistics. Retrieved January 16, 2015, from http://www.gmc-uk.org/doctors/register/search_stats.asp

Gillam, S., & Siriwardena, A. N. (2011). *The Quality and Outcomes Framework: QOF-transforming general practice*. Oxon: Radcliffe Publishing.

Glaser, B. G. (1978). *Theoretical sensitivity: Advances in the methodology of grounded theory*. Sociology Press.

Gold, N., & Mohan, A. (2003). A framework for understanding conceptual changes in evolving source code. In *Proceedings of the International Conference on Software Maintenance* (pp. 431 – 439). IEEE.

Golfarelli, M., & Rizzi, S. (1998). A methodological framework for data warehouse design. In *Proceedings of the 1st ACM international workshop on Data warehousing and OLAP* (pp. 3 – 9). ACM.

Graham, I. (2007). *Business rules management and service oriented architecture: a pattern language*. John wiley & sons.

Graves, T. (2013). CRUD, CRUDE and other action-acronyms. Retrieved January 25, 2015, from http://weblog.tetradian.com/2013/12/31/crud-crude-action-acronyms/

Henry, J. E., & Cain, J. P. (1997). A quantitative comparison of perfective and corrective software maintenance. *Journal of Software Maintenance: Research and Practice*, *9*(5), 281 – 297.

Henry, J. E. E., & Cain, J. P. P. (1997). A quantitative comparison of perfective and corrective software maintenance. *Journal of Software Maintenance-Research and Practice*, *9*(5), 281–297. http://doi.org/10.1002/(sici)1096-908x(199709/10)9:5<281::aid-smr154>3.3.co;2-g

Herron, D., & Garmus, D. (2000). Using Function Points Effectively. Retrieved January 28, 2015, from http://www.informit.com/articles/article.aspx?p=19795

Hevner, A., & Chatterjee, S. (2010). *Design research in information systems: theory and practice* (22nd ed.). Springer Science & Business Media.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105.

Hohwiller, J., Schlegel, D., Grieser, G., & Hoekstra, Y. (2011). Integration of BPM and BRM. *Lecture Notes in Business Information Processing*, *95 LNBIP*, 136–141. http://doi.org/10.1007/978-3-642-25160-3_12

Hohwiller, J., Schlegel, D., Grieser, G., & Hoekstra, Y. (2011). Integration of BPM and BRM. In *Business Process Model and Notation* (pp. 136 – 141). Springer Berlin Heidelberg.

Hong, S., van den Goor, G., & Brinkkemper, S. (1993). A formal approach to the comparison of object-oriented analysis and design methodologies. In *Proceeding of the Twenty-Sixth Hawaii International Conference on System Sciences* (Vol. 4, pp. 689 – 698). IEEE.

HSCIC. (2012). *Investment in General Practice - 2007-08 to 2011-12, England, Wales, Northern Ireland and Scotland*. Retrieved from http://www.hscic.gov.uk/catalogue/PUB07472

HSCIC. (2015). QOF business rules v31.0. Retrieved May 7, 2015, from http://www.hscic.gov.uk/qofbrv31

Inmon, W. H. (2005). *Building the Data Warehouse*. John wiley & sons.

ISO. (2011). ISO/IEC 25010:2011(en). Retrieved January 23, 2015, from ISO/IEC 25010:2011(en)

ISO. (2014). Iso/Iec 25010:2011. *Software Process: Improvement and Practice*, *2*(Resolution 937), 1 – 44. Retrieved from http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35733

JBoss Community. (2015). Drools homepage. Retrieved January 20, 2015, from http://www.drools.org/

Jonkers, H., Lankhorst, M. M., Ter Doest, H. W. L., Arbab, F., Bosma, H., & Wieringa, R. J. (2006). Enterprise architecture: Management tool and blueprint for the organisation. *Information Systems Frontiers*, *8*(2), 63–66. http://doi.org/10.1007/s10796-006-7970-2

Kan, S. H. (2002). *Metrics and models in software quality engineering*. Addison-Wesley Longman Publishing Co., Inc.

Kazman, R., Bass, L., Abowd, G., & Webb, M. (1994). SAAM: a method for analyzing the properties of software architectures. *Proceedings of 16th International Conference on Software Engineering*. http://doi.org/10.1109/ICSE.1994.296768

Kazman, R., Bass, L., Klein, M., Lattanze, T., & Northrop, L. (2005). A basis for analyzing software architecture analysis methods. *Software Quality Journal*, *13*(4), 329 – 355.

Kazman, R., Bass, L., Webb, M., & Abowd, G. (1994). SAAM: A method for analyzing the properties of software architectures. In *Proceedings of the 16th international conference on Software engineering* (pp. 81 – 90). IEEE Computer Society Press.

Kazman, R., Klein, M., & Clements, P. (2000). ATAM: Method for architecture evaluation - (No. CMU/SEI-2000-TR-004). Pittsburgh, Pennsylvania: Carnegie-Mellon University, Pittsburgh.

Kemerer, C. F., & Slaughter, S. (1999). An empirical approach to studying software evolution. *IEEE Transactions on Software Engineering*, *25*(4), 493 – 509.

Kim, Y., Lee, Z., & Gosain, S. (2005). Impediments to successful ERP implementation process. *Business Process Management Journal*, *11*(2), 158–170. http://doi.org/10.1108/14637150510591156

Kimball, R., & Ross, M. (2002). *The data warehouse toolkit*. (R. Elliott, Ed.) (2nd editio). New York: John wiley & sons.

Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit, The Definitive Guide to Dimensional Modeling*. http://doi.org/10.1145/945721.945741

King, R., & Green, P. (2012). Governance of primary healthcare practices: Australian insights. *Business Horizons*, *55*(6), 593 – 608.

Klein, M. H., Kazman, R., Bass, L., Carriere, J., Barbacci, M., & Lipson, H. (1999). Attribute-Based Architecture Styles. *Software Engineering Institute, Carnegie Mellon University*, 225 – 243. http://doi.org/CMU/SEI-99-TR-022

Kovacic, A. (2004). Business renovation: business rules (still) the missing link. *Business Process Management Journal*, *10*(2), 158 – 170.

Kvale, S. (2008). *Doing interviews*. SAGE Publications.

Lagerström, R. (2010). *Enterprise Systems Modifiability Analysis*. KTH, Royal Institute of Technology.

Lankhorst, M. (2009). *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Heidelberg: Springer.

Lester, H., & Campbell, S. (2010). Developing Quality and Outcomes Framework (QOF) indicators and the concept of'QOFability. *Quality in Primary Care*, *18*(2), 103 – 109.

Liao, S. (2004). Expert system methodologies and applications—a decade review from 1995 to 2004. *Expert Systems with Applications*, *28*(1), 93–103. http://doi.org/10.1016/j.eswa.2004.08.003

Lindvall, M., Tvedt, R. T., & Costa, P. (2003). An empirically-based process for software architecture evaluation. *Empirical Software Engineering*, *8*(1), 83 – 108.
Linstedt, D., Graziano, K., & Hultgren, H. (2009). *The New Business Supermodel, The Business of Data Vault modeling* (2nd editio). Lulu.com.
Longstreet, D. (2008). Function Points. www.softwaremetrics.com. Retrieved from www.softwaremetrics.com/files/OneHour.pdf
Lundberg, L., Bosch, J., Häggander, D., & Bengtsson, P. (1999). Quality Attributes in Software Architecture Design. *Database*, 1–10. Retrieved from http://www.janbosch.com/Articles/Guidelines.pdf
Mannaert, H., & Verelst, J. (2009). *Normalized systems: re-creating information technology based on laws for software evolvability*. Koppa.
Manuel, P. D., & AlGhamdi, J. (2003). A data-centric design for n-tier architecture. *Information Sciences*, *150*(3), 195 – 206.
Mario, G. (2010). *The art of enterprise information architecture: a systems-based approach for unlocking business insight*. Pearson Education India.
Martin, J., & Savant Institute. (1983). *Managing the data-base environment*. Englewood Cliffs (NJ): Prentice-Hall.
Mattsson, M., Grahn, H., & Mårtensson, F. (2006). Software Architecture Evaluation Methods for Performance, Maintainability, Testability, and Portability. In *Proceedings of the 2nd International Conference on the Quality of Software Architectures*. Retrieved from http://www.bth.se/fou/forskinfo.nsf/0/412d328ffb033edec125730e004b649d?OpenDocument&Click=\nhttp://www.researchgate.net/publication/200622064_Software_Architecture_Evaluation_Methods_for_Performance_Maintainability_Testability_and_Portability/file/72e7e52
Maxwell, K. D., Van Wassenhove, L., & Dutta, S. (1996). Software development productivity of European space, military, and industrial applications. *IEEE Transactions on Software Engineering*, *22*(10), 706 – 718.
McCall, J. a., Richards, P. K., & Walters, G. F. (1977). Factors in Software Quality. *Nat'l Tech. Information Servicel*, *1, 2 and 3*(ADA049055). Retrieved from http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA049055
Medvidovic, N., & Taylor, R. N. (2000). A classification and comparison framework for software architecture description languages. In *IEEE Transactions on Software Engineering* (pp. 70 – 93). IEEE.
Mohaghegh, S. D. (2005). Recent developments in application of artificial intelligence in petroleum engineering. *Journal of Petroleum Technology*, *57*(4), 86 – 91.
Morgan, T. (2002). *Business rules and information systems: aligning IT with business goals*. Addison-Wesley Professional.
Morris, P. (2002). Role of Measurement in Informed IT Decision Making. Canberra: Total Metrics.
Nammuni, K., Pickering, C., Modgil, S., Montgomery, A., Hammond, P., Wyatt, J. C., & Potts, H. W. (2004). Design-a-trial: a rule-based decision support system for clinical trial design. *Knowledge-Based Systems*, *17*(2), 121 – 129.
Nelson, M. L., Peterson, J., Rariden, R. L., & Sen, R. (2010). Transitioning to a business rule management service model: Case studies from the property and casualty insurance industry. *Information & Management*, *47*(1), 30–41. http://doi.org/10.1016/j.im.2009.09.007
NESMA. (2012). *FPA toegepast bij DATA WAREHOUSING*. Retrieved from http://www.ebookdb.org/reading/50261B162F7BG312183D2469/FPA-Toegepast-Bij-Data-Warehousing
NH Computing. (2008). The Normalisation Process. Retrieved January 22, 2015, from http://www.sqa.org.uk/e-learning/SoftDevRDS02CD/page_11.htm
NHS Employers. (2014). Quality and outcomes framework. Retrieved January 17, 2015, from http://www.nhsemployers.org/your-workforce/primary-care-contacts/general-medical-services/quality-and-outcomes-framework
NHS England. (2014). *2014/2015 General Medical Services (GMS) Contract Quality and Outcomes Framework (QOF)*. Retrieved from http://www.nhsemployers.org/~/media/Employers/Documents/Primary care contracts/QOF/2014-15/14-15 General Medical Services contract - Quality and Outcomes Framework.pdf
NHS England. (2015). About the National Health Service (NHS). Retrieved January 15, 2015, from http://www.nhs.uk/NHSEngland/thenhs/about/Pages/overview.aspx
NICTIZ. (2014). Nationaal ICT Instituut in de Zorg - NICTIZ.
Niessink, F., & van Vliet, H. (1997). Predicting maintenance effort with function points. In *Proceedings of the International Conference on Software Maintenance* (pp. 32 – 39). IEEE.
Nijssen, G. M., & Le Cat, A. (2010). *Kennis Gebaseerd Werken*. Amsterdam: PNA Publishing B.V.
NIVEL. (2015a). Wie heeft welke gezondheidsproblemen? Incidenties en prevalenties. Retrieved January 18, 2015, from http://www.nivel.nl/NZR/inci
NIVEL. (2015b). Zorggebruik per aandoening. Retrieved January 18, 2015, from http://www.nivel.nl/NZR/zorggebruik-per-aandoening
NVZ - Nederlandse Vereniging van Ziekenhuizen. (2013). Hoeveelheid aan regels maken ziekenhuiszorg complex. Retrieved June 14, 2010, from https://www.nvz-ziekenhuizen.nl/_page/12769/1137-nvz-voorzitter-van-rooy-hoeveelheid-aan-regels-maken-ziekenhuiszorg-complex

O'Brien, W. (2008). Avoiding semantic and temporal gaps in developing software intensive systems. *Journal of Systems and Software*, *81*(11), 1997 – 2013.

O'Brien, W. J., Hammer, J., Siddiqui, M., & Topsakal, O. (2008). Challenges, approaches and architecture for distributed process integration in heterogeneous environments. *Advanced Engineering Informatics*, *22*(1), 28 – 44.

Office for National Statistics. (2013). What are the top causes of death by age and gender? Retrieved January 18, 2015, from http://www.ons.gov.uk/ons/rel/vsob1/mortality-statistics--deaths-registered-in-england-and-wales--series-dr-/2012/sty-causes-of-death.html

OpenRules®. (2014). The Decision Model: Executable Live Primer. Retrieved January 20, 2015, from http://openrules.com/decision_model_primer.htm

Oskarsson, Ö. (1982). *Mechanisms of modifiability in large software systems*. VTT Grafiska.

Ossher, H., & Tarr, P. (2001). Using multidimensional separation of concerns to (re) shape evolving software. *Communications of the ACM*, *44*(10), 43 – 50.

Paré, G., Trudel, M.-C., Jaana, M., & Kitsiou, S. (2015). Synthesizing information systems knowledge: A typology of literature reviews. *Information & Management*, *52*(2), 183–199. http://doi.org/10.1016/j.im.2014.08.008

Paschke, A., & Bichler, M. (2008). Knowledge representation concepts for automated SLA management. *Decision Support Systems*, *46*(1), 187 – 205.

Petriu, D., Shousha, C., & Jalnapurkar, A. (2000). Architecture-based performance analysis applied to a telecommunication system. *IEEE Transactions on Software Engineering*, *26*(11), 1049 – 1065.

Pons, A. P. (2003). The furniture company: deductive databases and the scheduling problem. *International Journal of Information Management*, *23*(6), 523 – 536.

Port, D., & Liguo, H. (2003). Strategic architectural flexibility. In *Proceedings of the International Conference on Software Maintenance*. Amsterdam: IEEE.

Primatek. (2009). Business Rules: A classification. Primatek Consulting Inc. Retrieved from https://www.google.nl/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCcQFjAA&url=http://www.primatek.ca/blog/?dl_name=PrimatekConsulting-BusinessRulesClassification-WhitePaper.pdf&ei=-3xMVfSAIsmcsgHd5ICQBQ&usg=AFQjCNHvomMGtivF90oKOI7DiMcthMi

Radatz, J., Geraci, A., & Katki, F. (1990). *IEEE standard glossary of software engineering terminology*.

Randell, B. (1979). Software engineering in 1968. In *Proceedings of the 4th international conference on Software engineering* (pp. 1 – 10). IEEE Press.

Rijksinstituut voor Volksgezondheid en Milieu. (2013). Sterfte naar doodsoorzaak: Waaraan overlijden mensen in Nederland? Retrieved January 18, 2015, from http://www.nationaalkompas.nl/gezondheid-en-ziekte/sterfte-levensverwachting-en-daly-s/sterfte-naar-doodsoorzaak/waaraan-overlijden-mensen-in-nederland/

Ross, A. M., Rhodes, D. H., & Hastings, D. E. (2008). Defining changeability: Reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value. *Systems Engineering*, *11*(3), 246–262. http://doi.org/10.1002/sys.20098

Ross, R. G. (2003). *Principles of the business rule approach*. Addison-Wesley Longman Publishing Co.

Ross, J. W., Weill, P., & Robertson, D. C. (2007). Enterprise Architecture as Strategy. *Center for Infor Mation Systems Research, MIT …*, 1–10. Retrieved from http://semanticommunity.info/@api/deki/files/6830/=JeanneRoss01082007.pdf

Rowe, D., Leaney, J., & Lowe, D. (1998). Defining systems architecture evolvability - a taxonomy of change. *International Conference and Workshop: Engineering of Computer-Based Systems*, 45–52. http://doi.org/10.1109/ECBS.1998.10027

Schadd, M. P. D. (2013). Blueriq embraces the decision model. Blueriq. Retrieved from https://www.blueriq.com/kenniscenter/research-paper-blueriq-embraces-the-decision-model/

Schoen, C., Osborn, R., Huynh, P. T., Doty, M., Davis, K., Zapert, K., & Peugh, J. (2004). Primary care and health system performance: adults' experiences in five countries. *Health Affairs*, *23*, 283 – 283.

Seddon, M. E., Marshall, M. N., Campbell, S. M., & Roland, M. O. (2001). Systematic review of studies of quality of clinical care in general practice in the UK, Australia and New Zealand. *Quality in Health Care*, *10*(3), 152 – 158.

Shao, J., & Pound, C. J. (1999). Extracting business rules from information systems. *BT Technology Journal*, 179 – 186.

Shaw, D. R., Holland, C. P., Kawalek, P., Snowdon, B., & Warboys, B. (2007). Elements of a business process management system: theory and practice. *Business Process Management Journal*, *13*(1), 91–107. http://doi.org/10.1108/14637150710721140

Shaw, M., & Garlan, D. (1996). *Software architecture: perspectives on an emerging discipline* (1st ed.). Englewood Cliffs: Prentice Hall.

Shepperd, M., Mair, C., & Forselius, P. (2006). An Empirical Analysis of Software Productivity. In *Proceedings of the 3rd Software Measurement European Forum (SMEF'06)*. Rome: Southampton Solent University.

Simon, C., & Morton, A. (2010). Getting the most out of the QOF. In *The Quality and Outcomes Framework: QOF-Transforming General Practice* (pp. 111 – 127). Radcliffe Publishing.

Starfield, B., & Mangin, D. (2010). An international perspective on the basis for payment for performance. *Quality in Primary Care*, *18*(6), 399 – 404.
Stark, G. E. (1996). Measurements for managing software maintenance. In *Proceedings of the International Conference on Software Maintenance* (pp. 152 – 161). IEEE.
Strauss, A., & Corbin, J. M. (1990). *Basics of qualitative research: Grounded theory procedures and techniques*. Sage Publications, Inc.
Symons, C. R. (1988). Function point analysis: difficulties and improvements. *IEEE Transactions on Software Engineering*, *14*(1), 2 – 11.
Tarantino, A. (2006). *Manager's Guide to Compliance* (1st ed.). Hoboken, New Jersey: John wiley & sons.
Tarantino, A. (2008). *Governance, Risk, and Compliance Handbook: Technology, Finance, Environmental, and International Guidance and Best Practices*. John Wiley & Sons.
Tarr, P., Ossher, H., Harrison, W., & Sutton Jr, S. M. (1999). N degrees of separation: multi-dimensional separation of concerns. In *Proceedings of the 21st international conference on Software engineering* (Vol. 21, pp. 107 – 119). ACM.
TM forum. (2014). eTOM - Business process framework. Retrieved from http://www.tmforum.org/BusinessProcessFramework/1647/home.html
Turner, D. (2010). Qualitative Interview Design : A Practical Guide for Novice Investigators. *The Qualitative Report*, *15*(3), 754–760.
Van der Aalst, W. M. P. (1998). Three good reasons for using a Petri-net-based workflow management system. *Information and Process Integration in Enterprises: Rethinking Documents*, *08*, 161–182. http://doi.org/10.1.1.147.3781
Vanthienen, J. (2001). Ruling the business: about business rules and decision tables. In *New Directions in Software Engineering* (pp. 103 – 120).
Vanthienen, J., & Snoeck, M. (1993). Knowledge factoring using normalisation theory. *International Symposium on the Management of Industrial and Corporate Knowledge (ISMICK'93)*, 27 – 28.
Von Halle, B. (2007, June). The Rule Maturity Model: Five Steps to an Agile Enterprise. *Informationweek*. Retrieved from http://www.informationweek.com/software/information-management/the-rule-maturity-model-five-steps-to-an-agile-enterprise-/d/d-id/1056809?
Von Halle, B., & Goldberg, L. (2009). *The Decision Model: A Business Logic Framework Linking Business and Technology*. CRC Press.
Weber, B., Reichert, M., & Rinderle-Ma, S. (2008). Change patterns and change support features–enhancing flexibility in process-aware information systems. *Data & Knowledge Engineering*, *66*(3), 438 – 466.
Webster, J., & Watson, R. T. (2009). Analyzing the Past To Prepare for the Future : Writing a Review. *MIS Quarterly*, *26*(2), pp. xiii–xxiii/June 2002.
Weerd, I. Van De, & Brinkkemper, S. (2009). Meta-Modeling for Situational Analysis and Design Methods. In *Handbook of Research on Modern Systems Analysis and Design Technologies and Applications* (pp. 35–54). Hershey: Idea Group Publishing.
Weiderman, N. H., Bergey, J. K., Smith, D. B., & Tilley, S. R. (1997). Approaches to Legacy System Evolution - (No. CMU/SEI-97-TR-014). Pittsburgh, Pennsylvania: Carnegie-Mellon University of Pittsburgh.
Weissgerber, P., & Diehl, S. (2006). Identifying refactorings from source-code changes. In *21st IEEE/ACM International Conference on Automated Software Engineering* (pp. 231 – 240). IEEE.
Widom, J. (1995). Research problems in data warehousing. In *InProceedings of the fourth international conference on Information and knowledge management* (pp. 25 – 30). ACM. http://doi.org/10.1145/221270.221319
Williams, L. G., & Smith, C. U. (1998). Performance evaluation of software architectures. In *Proceedings of the 1st international workshop on Software and performance* (pp. 164 – 177). ACM.
Xiao, L., & Greer, D. (2009). Adaptive agent model: Software adaptivity using an agent-oriented model-driven architecture. *Information and Software Technology*, *51*(1), 109 – 137.
Zachman, J. A. (1997). Enterprise architecture: The issue of the century. *Database Programming and Design*, *10*(3), 44 – 53.
Zoet, M. (2010). *Aligning Governance, Risk Management and Compliance with Business Process Development through Business Rules*. Utrecht University.
Zoet, M. (2014). *Methods and Concepts for Business Rules Management* (1st ed.). Utrecht: Hogeschool Utrecht.
Zoet, M., de Haan, E., & Smit, K. (2014). *Van Wetsanalyse tot Producten en Diensten voor Burgers en Bedrijven*. Utrecht.
Zoet, M., Ravesteyn, P., & Versendaal, J. (2011). A structured analysis of business rules representation languages : Defining a normalisation form. *Proceedings of the 22nd Australian Conference on Information Systems*, Paper 20.
Zoet, M., Smit, K., & Leewis, S. (2015). A Classification of Modification Categories for Business Rules. In *Proceedings of the 28th Bled eConference*.

Appendix A – Interview design

Interviewee name:
Interviewee position:
Interview timestamp:
Interview duration:

Question 1: Please describe to what extend your monthly activities are related to the business rules of the Quality and Outcomes Framework which are part of the pay-for-performance scheme of the NHS (can be amount of hours, days or percentage):

Question 2: How many years of experience do you have working with business rules?

Question 3: How long does it take you to **create** one single decision?

Question 4: How long does it take you to **delete** one single decision?

Question 5: How long does it take to you **update** one single decision (decision label)?

Question 6: How long does it take you to **create** a single business rule?

Question 7: How long does it take you to **delete** a single business rule?

Question 8: How long does it take you to **create** a single condition?

Question 9: How long does it take you to **delete** a single condition?

Question 10: How long does it take you to **update** a single condition (condition label)?

Question 11: How long does it take you to **create** a single fact value?

Question 12: How long does it take you to **update** a single fact value?

Question 13: How long does it take you to **delete** a single fact value?

Question 14: What factors do you think or do you know influence the process of translating laws and regulations into business rules? (For example: Tooling, Education, Years of experience, etc.)

Question 15: Do you have any comments or tips for the interviewer or the interview itself?

Examples per change category

As part of the interview several examples are prepared and utilized during the interviews to improve comprehensibility of the modification types questioned, see Section 2.3.

COPD ruleset_v29.0 A, B, C Version Date:27/06/2014

3 Indicator COPD003: The percentage of patients with COPD who have had a review, undertaken by a healthcare professional, including an assessment of breathlessness using the Medical Research Council dyspnoea scale in the preceding 12 months.

a) Denominator ruleset

Rule number	Rule	Action if true	Action if false
1	If COPDRVW_DAT > (PAYMENTPERIODEND_DAT – 12 months) AND If MRC_DAT > (PAYMENTPERIODEND_DAT – 12 months)	Select	Next rule
2	If REG_DAT > (PAYMENTPERIODEND_DAT – 3 months)	Reject	Next rule
3	If COPDEXC_DAT > (PAYMENTPERIODEND_DAT – 12 months)	Reject	Next rule
4	If COPD_DAT > (PAYMENTPERIODEND_DAT – 3 months)	Reject	Select

D, E

b) Numerator ruleset: To be applied to the above denominator population

F, G, H

Rule number	Rule	Action if true	Action if false
1	If COPDRVW_DAT > (PAYMENTPERIODEND_DAT – 12 months) AND If MRC_DAT > (PAYMENTPERIODEND_DAT - 12 months)	Select	Reject

		Read codes v2	CTV3	
21	COPDRVW_COD	66YM. 66YB0 66YB1	XaIet XaXCa XaXCb	Latest <= ACHIEVEMENT_DAT
		(Codes for COPD review)		
22	COPDRVW_DAT	Date of COPDRVW_COD		Chosen record

I, J, K

Appendix B – Semi-structured interview transcripts

This appendix contains both interview transcripts. The original audio recordings can be found on the CD-ROM accompanying this thesis.

Interviewee name: Ross Ambler
Interviewee position: Senior Information Analyst
Interview timestamp: 18-03-2015 / 11:40 – 12:58
Interview duration: 01:18:36

- **Short introduction of the interviewer and the goals of the research and the interview**
- **Introductory questions regarding change management of the QOF**

You and Richard are the only ones translating the policies and law and regulations that are changed two times a year and translates them in to business rules. Is that correct?

RA: yes that is part of the QOF framework that is basically extracted from primary care systems and the way that is currently done they specified the indicators in the business rule formats and then we hand them of another team. The team that puts it in the GP software and is allowed the data is extracted. That's how it all hangs together.

KS: so if I'm correct you also update the BR you manage the documents and if changes occur then you change the BR format and can be implemented later by other teams.

RA: Yes, twice a year there is a read code release where all the read codes within their clusters that specify if the patient got asthma or COPD there is a new batch of those release twice a year. Basically if a new COPD code comes out in April or October we are responsible for working with clinics and other stakeholders to decide of that code comes in the clusters and if does the documents have to be updated. And we pass that to the system suppliers and that will ensure that the GP practice uses the new code they will be rewarded as part of the framework.

KS: Yes ok, that's what I've read before. So that's great to hear. So my first question there for is can you describe at what extent your monthly activity's comprise working with BR? Is it only working with BR? There is a difference because twice a year there is a release and I think it is very busy for you after the release, right?

RA: Yes, besides the QOF BR we are also involved in hand services as well. We are busy primarily with QOF BR. As part of the BR it's not just writing the BR, its discussing with a lot of stakeholders, NHS England, Nice and NHS employers as well and GP committee. If Nice decide that a new indicator needs to been in the QOF than it needs to be negotiated with all the other stakeholders. The actual writing of the business rules takes quite a bit of our time but it's also the underlying discussion in the background which we get involved in and also we have a contact centre where inquiry's coming trough we need to answer. And yes it very do the cause of the year and we know every April and October we need to update the read codes and a period for updating the BR. At the same time there are discussions going on how we write BR for services and things like that. It's just a cycle of constant work. It's the case at which kind of work at which kind of time.

KS: good to hear. But if you look at the QOF. That's not really needed. Because the other frameworks ore services that you build BR for are also applicable. If we are talking about really changing, creating

or deleting BR can you say an average on a monthly bases how much time it take of your month. A percentage of the month? And estimate?
RA: Are we talking about the actually document itself and all the underlying work?
KS: No not the discussions because the discussions. Changing the documents and working with the BR. I don't know do you document them into word or you use tooling for it? Or anything else? That's relevant to know because it can be an influential factor that influence working with BR. I am talking about updating and creating of the BR and not the underlying discussions.
RA: Once we know what we are doing it depends on how many BR we have on at the time. The cost of some demands have more indicators or a couple that's a lot to work through. It's difficult to quantify in a time period it's just so variable between releases.
KS: Ok, I'll write it down. Maybe it is interesting to know how many years of experience you have working with BR. Because the department you are working with is founded in 2013 if I'm correct?
RA: I've been working for the Health & Social Care Information Centre for 7 years. But in term of this area of BR we only got the responsibility for writing them and maintaining them for 3,5 years. They used to be managed by an organization called Connection for Health and then when Nice took it over and the NHS became responsible for them.

- **Questions regarding measuring productivity scores**

If you look at the example in the example figure we see A, B and C. We call them decisions. The decision itself is the percentage of patients with COPD who have had a review. How many of those can you create on an hourly basis?
RA: It's not a standalone type once all of that is been agreed it's just that indicator decision there?
KS: Yes
RA: Once we get told what it is. The time it takes to just change the decision title.
KS: Yes it is the title and the description. So that's all
RA: Well I think most of the time we get given certainly for QOF without the time it is negotiated. We do not have too much involvement in terms of the QOF side of things of deciding what look like and how they read to the next step.
KS: So it's only copy paste if I am correct then? And us it as a label for the indicator, if I am correct?
RA: Yes that is correct. That defines what we are looking for in the denominator and the numerator. We tend to get even notes in most cases. In free hand services we have a part in deciding that. It tends to take a while because it has so much involvement in stakeholders so it needs to be agreed by everyone. But we try to make them consistent. Some indicators are quiet complicated and you might have to try to encompass lots of things in the working and this takes a longer time period to finalize. Where some are finalize in a minute.
KS: I can imagine it is hard to quantify then. I have a similar question about the deletion of the decision. Is it easy to delete the decision? If that is required by the update.
RA: It's is easy enough for us to do. But we need to get instructions from the other stakeholders. If we need to delete something out of QOF then it need go through a negotiation panel with Nice and NHS employers. So the actual removal of the text from the BR is a relatively swift job for us we just need to been told.
KS: ok just searching the document and updating by removing the text. That's it if I'm correct?

RA: Yes if we removing the indicator the decision will go along with the denominator and the numerator. We keep up changelogs on our website if you noticed. The actual process of removing things is relatively is quick and easy. Just receiving the instructions that's all.
KS: Is it hard to search like those indicators or is it just open the document and remove and it's finished. Does it take a long time to find or query?
RA: We know a COPD indicator is in a COPD rule sample so it is relatively simple to find. The difficulty comes with the clusters. So for example COPD 3 needs to be removed as ordered by Nice then al the clusters within the indicator. Some of those clusters maybe used by other indicators within the set. So if we removed the COPD 3 indicator we take the denominator and the numerator out and if any of those clusters used within other indicators we leave those clusters in. To manage that we have a database with every cluster with all the linkage between them. We can utilize the database to make it easier for us to see what is affected by the removal.
KS: Because part of the change is also of course the searching for the dependencies what you call cluster. It is relatively easy to find out to query. You have a system for it you just told. Can it be a hard job to find all of the dependencies?
RA: Yes, we have problems at the start as you can imagine. Because the way it is cross setup I think it is 25 different documents. There are a few clusters I think blood pressure, BP_COD, I think that is used in 4 different documents. So if you can imagine the only way we could find where blood pressure was used we have to go through all 25 different documents. As now we have a database so we can interrogate our database and type in BP_COD and it bring back every ruleset where that cluster appears in it makes our lives a lot easier. It was a bit of a job to set up in the first place but now we have it up and running. It's now the case to update it. It takes **10 days to update it at each release.**
KS: that's very important to know. So deleting a decision also involves querying the other database for its dependencies. But can we say that such an action is performed in 5 – 10 minutes. How long does it take?
RA: To actually search the database find the dependencies and delete it. It can be **a 10 minute job I suppose.**
KS: Those estimation are not a problem because this experiment is also theoretical and right now we are performing other interviews in our organization to find out what is the average productivity measure we can use in our formula. It's not only your figures but this measures are very case specific because it's for the NHS of course. That's why I am asking through on the productivity levels. So if you update a decision that does not involve any dependencies or does it?
RA: If it just a renaming it does involve keeping it up to date in our database. If the renaming is agreed and it does not involve changing the denominator or others it's just quick and easy task as well. Just receiving the instruction.
KS: Searching for it is like **one minute or something?** It is very easy to perform the instruction is clear and modify the label and its ok. Am I right?
RA: Yes providing the fact it has no effect on the numerator and denominator. It is **a 2 minute job** and it has to go through a quality assurance before it goes out to the big wide world.
KS: And the quality assurance, is that Richard? Not all people know how to read BR right?
RA: Yes that's part of the issue we have, not many people are familiar with reading them. We have an internal quality assurance that's Richard as well and Dave and Louise.
KS: You have a line manager right?

RA: Yes Dave is line manager and Louise is a college we have and is currently on maternity leave. We have an internal quality insurance and it tend to be whoever is in the team at the time. And it's also when a proposal for new indicators comes in we send them out to a panel of experts and interested GP's and system suppliers just to get their opinions on it. We will draft up some BR and specify what the changes going to look like one we find out. And they go through to a review panel and that's just for the new indicators. We do not get a lot of feedback from them.
KS: So there is a mechanism in place that reviews the quality of the changes. That's good to know I didn't know that yet. On to the next question. The BR itself. I've marked them in the example with D and E. so we have rule numbers and each row is a BR, right?
RA: We called the term BR the logic of the indicator. Yes they are rules.
KS: You call the full rule number set indicator logic.
RA: Yes.
KS: one of those numbers as marked number 2. How much of those rules you can create on an hourly basis. That's one of your jobs right? To write the rules of the agreed changes.
RA: I would depend on the complexity of the indicator and de size of the change.
KS: this is really on the creating of the BR. Creating a new rule. Creating from scratch takes a while longer if am correct?
RA: Yes it would take a lot of work. One of us write it and then pas it through. Some things are more complicated than others.
KS: can you estimate just an average? How much on an average can you do in an hour?
RA: Yes, it's difficult. We split of work on it and then come back to them. It all depend on the complexity. If it is a simple one we could do it in **5 minutes.**
KS: And if it is a complex one how much is it then?
RA: it can anything from **5 minutes to several days to weeks.** But those are the complicated stuff.
KS: But if you talking about days you are talking about the full package and not just the rule.
RA: Sometimes the rule itself takes days. Depending on that one rule and where that one rule is.
KS: How many days is that in average for a complex rule.
RA: A general rule. 3 to 5 days.
KS: That's how I can really quantify it. I can image you also delete BR out of the logic. Out of the indicator logic. Does that take a lot of time? Because you also have to check the dependencies or does it not applies on this because the data in the BR also is in the clusters.
RA: The dependencies on the clusters and how the rule links to the clusters it also removing the cluster as well. And there is also sometimes if a rule needs to be deleted from COPD sometimes it have a knock-on effect. If you looking at it just in isolation just the process of removing the rule, checking if it have any dependencies in the database that is done in **15 minutes.**
KS: And if we are talking about. You said it has a knock-on effect because it really can relate to the rules in the whole set. To find out such problems how long does that take? From your answer it takes longer then?
RA: Just depend on what the issue is. If we are remove a set we know there are certain rules like Rule 2 (example). We know that is in every denominator. If there is a knock-on effect it takes longer. Checking the database.
KS: So the time is then doubled from 15 to 30 minutes or 45 minutes?
RA: Yes probably 30 minutes.

KS: Yes the other dependencies should be checked as well. Ok that's clear. We go to the next category. We see conditions and we call the one in the F, G and H conditions. Is that the same for you?
RA: We refer to them as clusters.
KS: How many of those clusters (conditions) can you create on an hourly basis? We are not talking about and full BR but in an existing BR with new clusters.
Because for the next questions if you are really changing the cluster we call them fact value. If you really change the value of the fact value itself then you are not influencing the cluster name if I am correct? The cluster is merely a label for the content of the fact value.
RA: Yes correct. The fact values are defined in the cluster.
KS: is it like a 5 minute job or just to create or is it longer or shorter?
RA: Depending on the complexity if it is just the case of adding that label in to the rule. Once we've got that label sorted it's just a **5 – 10 minute job**.
KS: the deletion what's the productivity of that? If you delete such things than you have the dependencies again.
RA: That is the same issue. Checking in the database for dependencies. That will take **15 minutes.**
KS: there are also dependencies when you update such conditions, right?
RA: Yes specially if we got a cluster that are in more than 1 rule set we keep them consistent between documents but if we update in more than 1. If we change BP_COD in bloodpres_COD it changes in more than 1 document.
KS: on your experience does that happens a lot? Like, checking other documents for the effect of the condition.
RA: Yes we have to do it for a couple.
KS: If you make an estimation like 10 minutes again or 15 minutes for searching for the dependencies and really change the other documents as well.
RA: Searching and changes it in 1 document it's like **15 minutes**. It increases by 5 minutes per time per document.
KS: So it builds up.
RA: Yes it's related with the content.
KS: the last part is about the fact values. In the I, J and K section. Those of the historical data we have analysed. The fact values changes the most often as a category so we see fact values are often created and deleted for example.
RA: Yes.
KS: Because it is a simple operation. Because you only have to insert a code for a creation. Right?
RA: Yes. The main volume of work lead in to deciding what comes in. The actual removing and creating is relatively a straight forward job. They have dependencies as well and diabetes code is used over several clusters. For that we utilize the database to find out the dependencies. The actual work is **a 5 minute job.**
KS: That's both for creating and deleting?
RA: Yes, that's just the physical addition to the document itself. Updating the database takes a bit longer.
KS: Can you estimate the updating in the database?
RA: We tend to it every 6 months at terms of QOF incorporate the fact values and the decision it takes **about 10 days**. A full roll over as we call it. That's depend on how many changes happens in the year.

KS: so my last question about the fact value then is do you think there is a change between creation/updating/deletion of fact values. Is there a difference between them? Because literature on LoC for example in BR we work with FP. In the software development industry we see they also use LoC as a measurement. In BR that is different.
RA: We are not involved in the final decision. In terms of deleting/updating ore creating it's all the same not much of a difference.

- **Closing questions regarding factors that influence productivity**

The last question: what factors do you think that influence the process? For example, Tooling, experience and/or education?
RA: Yes, tooling and experience are definitely a factor. Primary care experience it's not just working with codes you have to understand how the primary care works and depend on other stakeholders.
KS: And how did you combine that knowledge because you have some IT knowledge and primary care knowledge? Your education of following courses at you organization?
RA: I use to work with the QOF outputs and of the BR. And some work for Nice for commercial database data. And a lot of self-education.
KS: I don't want to be rude but is it possible to gain access to those documents for future research?
RA: Are you talking about the training documents?
KS: I think we could learn a lot. The NHS is a big and complex case and we use that information for research purpose.
RA: I think I have to contact my manager for that.
KS: Do you have any additional comments or tips?
RA: Yes, some things are hard to define, quantifying takes a lot of estimating.

Interviewee name: Richard Taylforth
Interviewee position: Senior information Analyst
Interview timestamp: 18-03-2015 / 13:00 – 14:01
Interview duration: 01:01:54

- **Short introduction of the interviewer and the goals of the research and the interview.**
- **Introductory questions regarding change management of the QOF**

KS: To what extend do your daily activities comprise changing BR's?
RT: Same story as RA. Depends on time of year.
KS: I seek for a general percentage of how much time from your work time do you work on BR? Like, for example, 40 percent.
RT: Same story about depending how big the changes and size is, see the answer of Ross.
KS: Not a problem, Ross also found it hard to quantify an answer. We are trying to refine the measurement instrument and use at other Dutch companies also.
KS: How much years of experience do you got working with BR?
RT: 11 months of experience working with BR. The whole team got more experience than me.

- **Questions regarding measuring productivity scores**

KS: If you create a decision how long does it take?
RT: is this like any changes in the indicators?
KS: for example: they have agreed upon the change of version 29 to 30 with a new indicator COPD 25.
RT: it's just the case of how long does it take.
KS: Yes correct.
RT: Same story about how long it does till everything is agreed. **5 – 10 minutes**
KS: Do you think there is a difference between Renaming, deleting or creating of a decision? Then can really have dependencies. Do you think it influences the time it takes?
RT: Examples of dependencies of Indicator dependencies.
KS: So if we take of deletion of such an indicator with no dependencies.
RT: It requires a lot of checking for cluster dependencies.
KS: so are we talking minutes, hours or days?
RT: Same answer as Ross explained concerning the dependency-database that is used.
KS: basically if you delete is without any dependencies you always have to check it. How much time takes the checking?
RT: If we want to delete a COPD 3 indicator. See the elaboration of Ross on clusters. Query the database and make sure what is need to be done. **15 – 30 minutes.** The changelogs are find on our website and are available for everyone. And those changelogs we have to keep up to date.
KS: ok that's clear. Do you have the same problem with dependencies if you update a decision?
RT: Just the word in change then. We also have to update our database so. I think **15 minutes.**

KS: That's clear. On to the next section D and E are about the BR or the logic as you call it.
How many BR can you create on an hourly basis? How long does it take to create such a rule?
RT: See the elaboration of Ross on Rule complexity**.** In QOF we have an exclusion and exception. Patients have a change to be selected first then they get accepted after**. Simple rules takes 5 – 10 minutes** and for some more complex rules it takes hours or days and if scenario testing it takes weeks. Scenario testing takes the whole teams with different kind scenarios. This takes a lot more time to produce.
KS: You mentioned like a complex rule takes a week.
RT: Yes some maybe longer.
KS: Deleting such a rule. Also you have dependencies then. How much of those rules can you delete in an hour?
RT: Rule number 2 you have highlighted in the example. That's a straight forward deletion with not many dependencies so that will take **10 – 15 minutes. More complex deletions takes up to an hour several days.** Checking if the deletion you have done I done correct.
KS: And you also mentioned different kinds of scenarios in testing?
RT: It's not a program to do it in. We used a lot of time scale drawings.
KS: That's more manual labour right?
RT: Yes but I don't know any kind of easy way to get that done.
KS: ok that's clear. Then we have done the BR part. Now we can move on to the Condition's part or the cluster part. That's F, G and H in the example.
If you have to create conditions in existing BR how long does that take?
RT: That can take quite some time. If you have an advance clinical knowledge then it get faster. But we do not take any clinical decisions. So that can takes quite a while.
KS: so if we say. We add an existing one. Really straight forward then?
RT: That's relatively a straight forward task with things to considering. See the same example Ross proposed
KS: ok so are we talking like **an hour?**
RT: yes potential.
KS: And if we want a full new one?
RT: In most cases that can takes longer because we need to know more of our stakeholders.
KS: So when that is agreed upon?
RT: maybe a similar time frame I think slightly longer.
KS: So then we are talking about. You said 60 minutes depending on the complexity.
RT: This one can be double that. **So 120 minutes**.
KS: If we say, deletion of conditions
RT: This slightly covered in the first one. Checking dependencies and the flow of the logic of the ruleset. It could be quite time consuming. **Potentially a day**. Depending on the complexity.
KS: So significantly longer for a deletion then a creation?
RT: It is not necessary longer. **But almost the same.**
KS: It's is an estimation so it don't have to be right on.
The last for the conditions.
Renaming the condition.
RT: It could be quite time consuming.

KS: And if we say you have to change only for the source?
RT: Again a straight forward task. Checking the documents and updating the changelogs. A couple of hours I think. 1 or 2 hours
KS: and with more dependencies at other documents we can add time?
RT: Yes.
KS: The last category is about the fact values. How many of those fact values can you create in an hour?
RT: Same answer as Ross provided, see his elaboration.
RT: 5 – 10 minutes time per fact value. And for each kind of dependency add up the same amount.
KS: and with update and deletion are we talking about the same time ranges?
RT: we have to make sure we captured every single rule set and the change log itself.
KS: so it is not a significantly difference between those?
RT: At single codes no. But complex changes longer so 5 – 10 minutes.

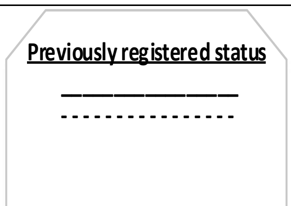
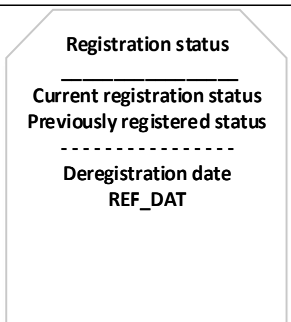
- **Closing questions regarding factors that influence productivity**

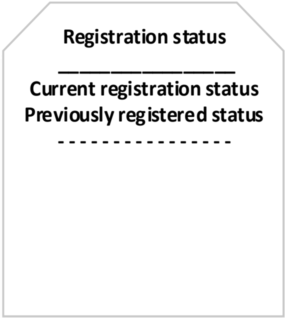
KS: So my last question is about the factors you think influence the process performing the changes in the documents?
RT: Stakeholders takes a lot more time. So discussion with the stakeholders takes a lot of time.
KS: so if the changes of the stakeholders are less ambiguous that takes less time.
RT: Yes indeed. And **primary care knowledge.**
KS: and regarding your training?
RT: yes I worked on BR before on coming to the team. I came from another team in the organization which is an analytical team. Working on diabetes audits. When I first joined the team I was tested by the other team for setting up a BR. Sort of scenario testing. It is quite an analytical role so an analytical background is good.

Appendix C – Architecture description languages

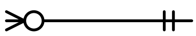
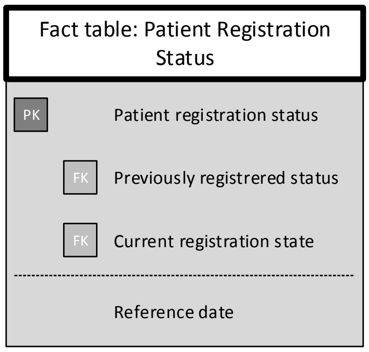
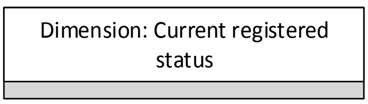
This appendix includes the three ADL's utilized in this research project. For each ADL we include and elaborate upon each symbol. For TDM we only elaborate the rule-oriented approach. For the remaining two ADL's we elaborate on both the data-orientation and rule-orientation.

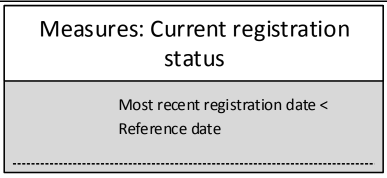
TDM

Symbol:	**Rule-orientation** (Von Halle & Goldberg, 2009)**:**
	Relates derived conditions.
	The decision model notation starts with a decision. The top rule family is derived from the decision.
	TDM requires the business logic to be normalized, meaning that each decision is supported by a maximum of one conclusion. The conclusion is expressed above the solid line.
	Two types of conditions are recognized. The first type of conditions are derived facts, expressed between the solid and dotted line. These conditions are used as conditions to answer a conclusion in their current rule family, but are conclusions underlying decisions. The second type of conditions are base facts and are expressed below the dotted lines in a decision. Base facts are atomic facts which do not have to be derived by any underlying decision.
	If a condition is placed under the solid line the condition is derived and is related to the next rule family.
	If a condition is placed under the dotted line the condition is called a base fact and is not derived.

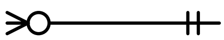
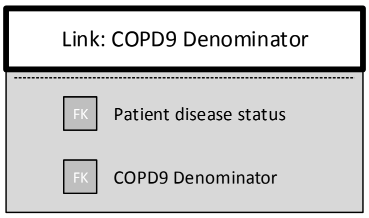
	The complete set of the conclusion and conditions in one square are defined as a rule family.

Kimball's dimensional modeling

Symbol:	**Data-orientation** (Kimball & Ross, 2013)	**Rule-orientation:**
	The relationship between tables, complemented by the cardinality of the relationship.	The relationship between a fact table and a derived fact in the form of a dimension. It also explains the flow and hierarchy.
	A Fact table consists of the measurements, metrics or facts of a given information request. It is located at the centre of the dimensional model and is surrounded by dimension tables. Fact tables contain the content of the data warehouse and store different types of measures like additive, non-additive, and semi additive measures.	Fact tables are the core of the dimensional model and is surrounded by dimensions containing other facts. The fact table contains a primary key and one or multiple foreign keys that derive other facts.
	Dimensions are the foundation of the fact table, and is where the data for the fact table is collected. Dimension tables contain descriptive textual information concerning the data it contains.	Dimensions are the foundation of the fact table, and is where the data for the fact table is collected. Dimension tables contain descriptive textual information concerning the derived fact.
	N.A.	A measures table contains the decision logic used to determine its parent fact. Decision logic comprises

		formulas, variables, and operators.
FK Patient disease status	A foreign key is the key that is allowed to migrate to other entities to define the relationships	A foreign key is an indicator that illustrates that there is a relationship between tables.
PK HUB INHT_DAT	A primary key is the driver of the table and only one key within the entity is selected to be the primary key.	The primary key is the driver of the table. Only one conclusion within the table is selected to be the primary key.
....	N.A.	Within the fact table a dotted line is included. This line represents the boundary between derived and non-derived facts. All facts placed above the dotted line are derived facts, all facts placed below the dotted line are non-derived facts.

Linstedt's data vault modeling

Notation:	**Data-orientation** (Linstedt et al., 2009)**:**	**Rule-orientation:**
	The relationship between tables, sometimes complemented by the cardinality of the relationship.	The relationship between the hubs, links, and satellites.
	Associations or transactions between primary business keys (Hubs) are modelled using Link tables	Associations or transactions between decisions (Hubs) are modelled using Link tables. The link contains de primary keys of the connected Hubs.

HUB COPD9 Denominator	The Hub contains all elements needed to answer a business request, and at least contains the following elements: a business (primary) key, a foreign (surrogate) key, used to connect the other structures to this table, and metadata fields.	The Hub represents a single decision. The Hub only contains descriptive textual information concerning the decision.
SAT COPD9 Denominator IF COPDSPIR_DAT >= (COPD_DAT – 3 months) AND IF COPDSPIR_DAT <= (COPD_DAT + 12 months)	Temporal attributes and descriptive attributes are stored in separate tables called Satellites. These consist of metadata linking them to their parent Hub or Link. The metadata describes the origin of the association and attributes, as well as a timeline with start and end dates for the attribute.	A satellite table contains the decision logic used to determine its parent table. Decision logic comprises formulas, variables, and operators.
FK Patient disease status	A foreign key is the key that is allowed to migrate to other entities to define the relationships	A foreign key is an indicator that illustrates that there is a relationship between tables.
PK HUB INHT_DAT	A primary key is the driver of the table and only one key within the entity is selected to be the primary key.	The primary key is the driver of the table. Only one conclusion within the table is selected to be the primary key.

Appendix D – Case-specific factors that influence productivity

The productivity scores are very dependent on how the modification type affect other elements with it relates to in the same or other business rules documents. To support the business rules team of the HSCIC a database is utilized on which all dependencies are managed. In almost all instances the database is consulted before a change is processed to see what effect it potentially has on other business rules documents. As can be observed from the productivity scores provided by the interviewees, most modification types are given an extra timeframe which is needed to check and process any dependencies the original change affects. Furthermore, according to the interviewees the concept of element complexity is also an important factor for each modification type. For example: the creation of a business rule differs significantly ranging from five minutes for a simple business rule to five days for a complex business rule which uses more conditions and have to be tested against scenarios manually. Again, whether the logic is existing in other business rules documents and can be reused or has to be created from scratch to successfully implement the change influences the productivity. The same phenomena can be observed when analysing the creation of conditions. Both interviewees state that these scores are heavily dependent on whether totally new logic has to be designed and implemented (creating a new condition label, updating the database, and creating a new cluster with underlying fact values) or existing logic can be applied in the change's context.

Furthermore, both interviewees were very consistent in identifying influential factors for the productivity of processing changes to the business rules documents. These factors were: 1) tooling to support processing changes to the business rules documents, 2) experience with working with business rules in general, 3) experience with the subject matter (medical domain), 4) available (internal) training, 5) having an analytical background, and 6) the completeness and unambiguousness of change requirements by stakeholders. While the first five are relatively straightforward factors, the sixth factor is elaborated further upon due to its potential effect on processing changes. The interviewees state that it occurs that stakeholders do not specify enough information or specify it in such a way that it is experienced as ambiguous by the BRM experts. When this occurs the BRM experts have to check back with the relevant stakeholder(s) to acquire enough information for them to perform the change as business rules are always processed to have a binary or predefined outcome, which does not allow for unambiguousness. Also, the BRM experts are not authorized to interpret unambiguous requirements. Decisions regarding interpretation of the requirements are at all times made by the originating stakeholder itself rather than by the BRM experts. The process of communicating with stakeholders regarding the stated requirements can take considerable time and will potentially hamper productivity.

Appendix E – Architecture descriptions of the QOF

Due to size issues this appendix can be found at the DVD that is enclosed with this thesis.

Appendix F – Scenario descriptions of the QOF

Due to size issues this appendix can be found at the DVD that is enclosed with this thesis.

Appendix G – Case selection criteria statistics

Clinical condition	Pages V8	Pages V19	Pages V29	BR's V8	BR's V19	BR's V29	Total modifications/Type
Blood pressure	X	X	3	X	X	16	22
Cervical Screening	X	X	5	X	X	16	39
Contraception	X	X	15	X	X	17	107
Cytology	4	5	X	16	16	X	73
PAD	X	X	15	X	X	36	124
Osteoprosis	X	X	15	X	X	37	171
Obesity	4	4	4	13	13	14	42
CVD	X	15	17	X	33	21	336
Cancer	6	6	9	18	19	18	154
Dementia	6	13	17	16	31	34	180
Atrial Fibrillation	10	10	17	38	38	41	207
Asthma	9	9	14	48	48	50	286
CKD	10	12	12	50	64	51	235
COPD	12	13	17	59	60	74	292
CHD	20	28	25	153	111	67	807
Diabetes	29	39	36	222	207	112	766

Appendix H – FP conversion per modification type

Modification Type	Time per modification type	Amount of minutes per function point	Amount of function points per modification type
CD	4.25	755	0.00562616
DD	16.25	755	0.02151178
UD	8.5	755	0.01125232
CBR	1200	755	1.58856235
DBR	25.5	755	0.03375695
CC	109	755	0.14429441
DC	150	755	0.19857029
UC	97.5	755	0.12907069
CFV	6.25	755	0.00827376
UFV	6.25	755	0.00827376
DFV	6.25	755	0.00827376

Appendix I – Base calculations for ALMA

In this appendix we present all calculations performed to estimate the total effort for each architectural candidate, utilizing the ALMA-method. First, we present the data sheets as described in the results chapter. The data sheets are followed by the base calculations as part of the prediction model. This set-up is repeated three times, once for the average effort estimation, once for the best case effort estimation, and once for the worst case effort estimation.

Average effort estimation

COPD data sheet:

MO	Changes COPD 8Y	Time per MO	MO's/M	MO's/FP	FP/MO	Normalized scenario weight COPD
CD	6	4.25	2484	177.6	0.01	0.02
DD	0	16.25	649	46.5	0.02	0.00
UD	17	8.5	1242	88.8	0.01	0.06
CBR	10	1200	8.8	0.6	1.59	0.03
DBR	1	25.5	414	29.6	0.03	0.00
CC	4	109	96	6.9	0.14	0.01
DC	3	150	70	5.0	0.20	0.01
UC	69	97.5	108	7.7	0.13	0.23
CFV	73	6.25	1689	120.8	0.01	0.24
UFV	67	6.25	1689	120.8	0.01	0.22
DFV	49	6.25	1689	120.8	0.01	0.16
Total	299					1

Diabetes Mellitus data sheet:

MO	Changes DM 8Y	Time per MO	MO's/M	MO's/FP	FP/MO	Normalized scenario weight DM
CD	51	4.25	2484	177.6	0.01	0.05
DD	58	16.25	649	46.5	0.02	0.06
UD	3	8.5	1242	88.8	0.01	0.00
CBR	221	1200	8.8	0.6	1.59	0.24
DBR	34	25.5	414	29.6	0.03	0.04
CC	2	109	96	6.9	0.14	0.00
DC	169	150	70	5.0	0.20	0.18
UC	100	97.5	108	7.7	0.13	0.11
CFV	175	6.25	1689	120.8	0.01	0.19
UFV	103	6.25	1689	120.8	0.01	0.11
DFV	21	6.25	1689	120.8	0.01	0.02
Total	937					1

Prediction model for the rule family-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	5	0.02813	0.02006689	0.000564497	0.007891307	22.22668698	10	2.222668698	1
DD	8	0.17209	0	0	0				
UD	8	0.09002	0.056856187	0.005118111	0.071547852				
CBR	5	7.94281	0.033444816	0.265645878	3.713556358				
DBR	5	0.16878	0.003344482	0.000564497	0.007891307		**Analysis results**		
CC	11	1.58724	0.013377926	0.021233961	0.296836938		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
DC	24	4.76569	0.010033445	0.047816258	0.668440144		2.22	0.28	0.02
UC	39	5.03376	0.230769231	1.161636219	16.23891776		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
CFV	16	0.13238	0.244147157	0.032320249	0.451816024		27.98	3.50	0.29
UFV	18	0.14893	0.224080268	0.033371763	0.466515517		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
DFV	16	0.13238	0.163879599	0.021694413	0.303273769		1679.00	209.88	17.49
							Total impact estimate H/period	**Total impact estimate M/period**	
							1.75	104.94	

	DM								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	40	0.22505	0.054429029	0.012249053	0.171233788	359.0089473	11	32.63717702	1
DD	36	0.77442	0.06189968	0.047936607	0.67012254				
UD	36	0.40508	0.003201708	0.001296959	0.018130636				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		**Analysis results**		
DBR	20	0.67514	0.036286019	0.024498107	0.342467576		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
CC	52	7.50331	0.002134472	0.016015602	0.223887684		32.64	4.08	0.34
DC	358	71.08817	0.18036286	12.8216648	179.2385231		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
UC	358	46.20731	0.106723586	4.93140954	68.93789349		410.90	51.36	4.28
CFV	125	1.03422	0.186766275	0.19315747	2.700215621		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
UFV	127	1.05077	0.109925293	0.115505959	1.614698082		24654.12	3081.77	256.81
DFV	130	1.07559	0.022411953	0.024106052	0.336986909		**Total impact estimate H/period**	**Total impact estimate M/period**	
							25.68	1540.88	

Prediction model for the fact-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	10	0.05626	0.02006689	0.001128995	0.015782615	22.93786397	10	2.293786397	1
DD	17	0.36570	0	0	0				
UD	10	0.11252	0.056856187	0.006397638	0.089434816				
CBR	5	7.94281	0.033444816	0.265645878	3.713556358				
DBR	5	0.16878	0.003344482	0.000564497	0.007891307		**Analysis results**		
CC	20	2.88589	0.013377926	0.038607201	0.539703524		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
DC	52	10.32566	0.010033445	0.103601892	1.44828698		2.29	0.29	0.02
UC	39	5.03376	0.230769231	1.161636219	16.23891776		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
CFV	12	0.09929	0.244147157	0.024240186	0.338862018		28.88	3.61	0.30
UFV	13	0.10756	0.224080268	0.024101829	0.336927874		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
DFV	11	0.09101	0.163879599	0.014914909	0.208500716		1732.73	216.59	18.05
							Total impact estimate H/period	**Total impact estimate M/period**	
							1.80	108.30	

	DM								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	40	0.22505	0.054429029	0.012249053	0.171233788	535.3773787	11	48.67067079	1
DD	96	2.06513	0.06189968	0.127830951	1.78699344				
UD	96	1.08022	0.003201708	0.003458556	0.048348364				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		Analysis results		
DBR	20	0.67514	0.036286019	0.024498107	0.342467576		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
CC	77	11.11067	0.002134472	0.023715411	0.331525993		48.67	6.08	0.51
DC	662	131.45353	0.18036286	23.70933547	331.4410678		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
UC	485	62.59929	0.106723586	6.680820186	93.39351492		612.76	76.60	6.38
CFV	74	0.61226	0.186766275	0.114349222	1.598527648		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
UFV	104	0.86047	0.109925293	0.094587557	1.322272445		36765.82	4595.73	382.98
DFV	72	0.59571	0.022411953	0.013351044	0.186638904		**Total impact estimate H/period**	**Total impact estimate M/period**	
							38.30	2297.86	

Prediction model for the decision-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	10	0.056261583	0.02006689	0.001128995	0.015782615	25.74874761	10	2.574874761	1
DD	25	0.537794546	0	0	0				
UD	54	0.607625099	0.056856187	0.034547246	0.482948004				
CBR	5	7.942811755	0.033444816	0.265645878	3.713556358				
DBR	5	0.16878475	0.003344482	0.000564497	0.007891307		Analysis results		
CC	25	3.607360339	0.013377926	0.048259001	0.674629405		Total impact estimate FP/8Y	Total impact estimate FP/Y	Total impact estimate FP/M
DC	55	10.92136616	0.010033445	0.109578925	1.531841998		2.57	0.32	0.03
UC	45	5.808181096	0.230769231	1.340349484	18.7372128		Total impact estimate H/8Y	Total impact estimate H/Y	Total impact estimate H/M
CFV	8	0.066190098	0.244147157	0.016160124	0.225908012		32.42	4.05	0.34
UFV	8	0.066190098	0.224080268	0.014831895	0.20734023		Total impact estimate M/8Y	Total impact estimate M/Y	Total impact estimate M/M
DFV	8	0.066190098	0.163879599	0.010847207	0.151636885		1945.06	243.13	20.26
							Total impact estimate H/period	Total impact estimate M/period	
							2.03	121.57	

	DM								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	40	0.22505	0.054429029	0.012249053	0.171233788	530.6732093	11	48.24301903	1
DD	164	3.52793	0.06189968	0.218377875	3.05278046				
UD	164	1.84538	0.003201708	0.005908367	0.082595121				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		Analysis results		
DBR	20	0.67514	0.036286019	0.024498107	0.342467576		Total impact estimate FP/8Y	Total impact estimate FP/Y	Total impact estimate FP/M
CC	96	13.85226	0.002134472	0.029567265	0.413331109		48.24	6.03	0.50
DC	675	134.03495	0.18036286	24.17492665	337.9497292		Total impact estimate H/8Y	Total impact estimate H/Y	Total impact estimate H/M
UC	422	54.46783	0.106723586	5.813002307	81.26198618		607.38	75.92	6.33
CFV	62	0.51297	0.186766275	0.095806105	1.339306948		Total impact estimate M/8Y	Total impact estimate M/Y	Total impact estimate M/M
UFV	90	0.74464	0.109925293	0.081854617	1.144274231		36442.78	4555.35	379.61
DFV	62	0.51297	0.022411953	0.011496733	0.160716834		Total impact estimate H/period	Total impact estimate M/period	
							37.96	2277.67	

Worst case effort estimation

COPD data sheet:

MO	Changes COPD 8Y	Time per MO	MO's/M	MO's/FP	FP/MO	Normalized scenario weight COPD
CD	6	7.5	2484	177.6	0.01	0.02
DD	0	22.5	649	46.5	0.03	0.00
UD	17	15	1242	88.8	0.02	0.06
CBR	10	1200	8.8	0.6	1.59	0.03
DBR	1	60	414	29.6	0.08	0.00
CC	4	210	96	6.9	0.28	0.01
DC	3	150	70	5.0	0.20	0.01
UC	69	90	108	7.7	0.12	0.23
CFV	73	7.5	1689	120.8	0.01	0.24
UFV	67	7.5	1689	120.8	0.01	0.22
DFV	49	7.5	1689	120.8	0.01	0.16
Total	299					1

Diabetes Mellitus data sheet:

MO	Changes DM 8Y	Time per MO	MO's/M	MO's/FP	FP/MO	Normalized scenario weight DM
CD	51	7.5	2484	177.6	0.01	0.05
DD	58	22.5	649	46.5	0.03	0.06
UD	3	15	1242	88.8	0.02	0.00
CBR	221	1200	8.8	0.6	1.59	0.24
DBR	34	60	414	29.6	0.08	0.04
CC	2	210	96	6.9	0.28	0.00
DC	169	150	70	5.0	0.20	0.18
UC	100	90	108	7.7	0.12	0.11
CFV	175	7.5	1689	120.8	0.01	0.19
UFV	103	7.5	1689	120.8	0.01	0.11
DFV	21	7.5	1689	120.8	0.01	0.02
Total	937					1

Prediction model for the rule family-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	5	0.02813	0.02006689	0.000564497	0.007891307	22.22668698	10	2.222668698	1
DD	8	0.17209	0	0	0				
UD	8	0.09002	0.056856187	0.005118111	0.071547852				
CBR	5	7.94281	0.033444816	0.265645878	3.713556358				
DBR	5	0.16878	0.003344482	0.000564497	0.007891307		**Analysis results**		
CC	11	1.58724	0.013377926	0.021233961	0.296836938		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
DC	24	4.76569	0.010033445	0.047816258	0.668440144		2.22	0.28	0.02
UC	39	5.03376	0.230769231	1.161636219	16.23891776		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
CFV	16	0.13238	0.244147157	0.032320249	0.451816024		27.98	3.50	0.29
UFV	18	0.14893	0.224080268	0.033371763	0.466515517		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
DFV	16	0.13238	0.163879599	0.021694413	0.303273769		1679.00	209.88	17.49
							Total impact estimate H/period	**Total impact estimate M/period**	
							1.75	104.94	

	DM								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	40	0.22505	0.054429029	0.012249053	0.171233788	359.0089473	11	32.63717702	1
DD	36	0.77442	0.06189968	0.047936607	0.67012254				
UD	36	0.40508	0.003201708	0.001296959	0.018130636				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		**Analysis results**		
DBR	20	0.67514	0.036286019	0.024498107	0.342467576		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
CC	52	7.50331	0.002134472	0.016015602	0.223887684		32.64	4.08	0.34
DC	358	71.08817	0.18036286	12.8216648	179.2385231		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
UC	358	46.20731	0.106723586	4.93140954	68.93789349		410.90	51.36	4.28
CFV	125	1.03422	0.186766275	0.19315747	2.700215621		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
UFV	127	1.05077	0.109925293	0.115505959	1.614698082		24654.12	3081.77	256.81
DFV	130	1.07559	0.022411953	0.024106052	0.336986909		**Total impact estimate H/period**	**Total impact estimate M/period**	
							25.68	1540.88	

Prediction model for the fact-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	10	0.05626	0.02006689	0.001128995	0.015782615	22.93786397	10	2.293786397	1
DD	17	0.36570	0	0	0				
UD	10	0.11252	0.056856187	0.006397638	0.089434816				
CBR	5	7.94281	0.033444816	0.265645878	3.713556358				
DBR	5	0.16878	0.003344482	0.000564497	0.007891307		**Analysis results**		
CC	20	2.88589	0.013377926	0.038607201	0.539703524		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
DC	52	10.32566	0.010033445	0.103601892	1.44828698		2.29	0.29	0.02
UC	39	5.03376	0.230769231	1.161636219	16.23891776		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
CFV	12	0.09929	0.244147157	0.024240186	0.338862018		28.88	3.61	0.30
UFV	13	0.10756	0.224080268	0.024101829	0.336927874		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
DFV	11	0.09101	0.163879599	0.014914909	0.208500716		1732.73	216.59	18.05
							Total impact estimate H/period	**Total impact estimate M/period**	
							1.80	108.30	
	DM								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	40	0.22505	0.054429029	0.012249053	0.171233788	535.3773787	11	48.67067079	1
DD	96	2.06513	0.06189968	0.127830951	1.78699344				
UD	96	1.08022	0.003201708	0.003458556	0.048348364				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		Analysis results		
DBR	20	0.67514	0.036286019	0.024498107	0.342467576		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
CC	77	11.11067	0.002134472	0.023715411	0.331525993		48.67	6.08	0.51
DC	662	131.45353	0.18036286	23.70933547	331.4410678		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
UC	485	62.59929	0.106723586	6.680820186	93.39351492		612.76	76.60	6.38
CFV	74	0.61226	0.186766275	0.114349222	1.598527648		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
UFV	104	0.86047	0.109925293	0.094587557	1.322272445		36765.82	4595.73	382.98
DFV	72	0.59571	0.022411953	0.013351044	0.186638904		**Total impact estimate H/period**	**Total impact estimate M/period**	
							38.30	2297.86	

Prediction model for the decision-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	10	0.099285147	0.02006689	0.001992344	0.027851673	25.44157461	10	2.544157461	1
DD	25	0.744638602	0	0	0				
UD	54	1.072279587	0.056856187	0.060965729	0.852261184				
CBR	5	7.942811755	0.033444816	0.265645878	3.713556358				
DBR	5	0.397140588	0.003344482	0.001328229	0.018567782		Analysis results		
CC	25	6.949960286	0.013377926	0.092976057	1.299744725		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
DC	55	10.92136616	0.010033445	0.109578925	1.531841998		2.54	0.32	0.03
UC	45	5.361397935	0.230769231	1.237245677	17.29588874		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
CFV	8	0.079428118	0.244147157	0.019392149	0.271089614		32.03	4.00	0.33
UFV	8	0.079428118	0.224080268	0.017798274	0.248808276		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
DFV	8	0.079428118	0.163879599	0.013016648	0.181964262		1921.86	240.23	20.02
							Total impact estimate H/period	**Total impact estimate M/period**	
							2.00	120.12	
	DM								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	40	0.39714	0.054429029	0.021615976	0.302177273	527.1657311	11	47.92415737	1
DD	164	4.88483	0.06189968	0.302369365	4.226926791				
UD	164	3.25655	0.003201708	0.01042653	0.145756096				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		Analysis results		
DBR	20	1.58856	0.036286019	0.057642604	0.805806061		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
CC	96	26.68785	0.002134472	0.056964456	0.796325989		47.92	5.99	0.50
DC	675	134.03495	0.18036286	24.17492665	337.9497292		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
UC	422	50.27800	0.106723586	5.365848283	75.01106416		603.37	75.42	6.29
CFV	62	0.61557	0.186766275	0.114967326	1.607168338		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
UFV	90	0.89357	0.109925293	0.09822554	1.373129077		36201.91	4525.24	377.10
DFV	62	0.61557	0.022411953	0.013796079	0.192860201		**Total impact estimate H/period**	**Total impact estimate M/period**	
							37.71	2262.62	

Best case effort estimation

COPD data sheet:

MO	Changes COPD 8Y	Time per MO	MO's/M	MO's/FP	FP/MO	Normalized scenario weight COPD
CD	6	1	2484	177.6	0.001323802	0.02006689
DD	0	10	649	46.5	0.01323802	0
UD	17	2	1242	88.8	0.002647604	0.056856187
CBR	10	1200	8.8	0.6	1.588562351	0.033444816
DBR	1	15	414	29.6	0.019857029	0.003344482
CC	4	7.5	96	6.9	0.009928515	0.013377926
DC	3	15	70	5.0	0.019857029	0.010033445
UC	69	15	108	7.7	0.019857029	0.230769231
CFV	73	5	1689	120.8	0.00661901	0.244147157
UFV	67	5	1689	120.8	0.00661901	0.224080268
DFV	49	5	1689	120.8	0.00661901	0.163879599
Total	299					1

Diabetes Mellitus data sheet:

MO	Changes DM 8Y	Time per MO	MO's/M	MO's/FP	FP/MO	Normalized scenario weight DM
CD	51	1	2484	177.6	0.001323802	0.054429029
DD	58	10	649	46.5	0.01323802	0.06189968
UD	3	2	1242	88.8	0.002647604	0.003201708
CBR	221	1200	8.8	0.6	1.588562351	0.235859125
DBR	34	15	414	29.6	0.019857029	0.036286019
CC	2	7.5	96	6.9	0.009928515	0.002134472
DC	169	15	70	5.0	0.019857029	0.18036286
UC	100	15	108	7.7	0.019857029	0.106723586
CFV	175	5	1689	120.8	0.00661901	0.186766275
UFV	103	5	1689	120.8	0.00661901	0.109925293
DFV	21	5	1689	120.8	0.00661901	0.022411953
Total	937					1

Prediction model for the rule family-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	5	0.00662	0.02006689	0.000132823	0.001856778	7.299737733	10	0.729973773	1
DD	8	0.10590	0	0	0				
UD	8	0.02118	0.056856187	0.001204261	0.016834789				
CBR	5	7.94281	0.033444816	0.265645878	3.713556358				
DBR	5	0.09929	0.003344482	0.000332057	0.004641945		Analysis results		
CC	11	0.10921	0.013377926	0.001461052	0.02042456		Total impact estimate FP/8Y	Total impact estimate FP/Y	Total impact estimate FP/M
DC	24	0.47657	0.010033445	0.004781626	0.066844014		0.73	0.09	0.01
UC	39	0.77442	0.230769231	0.178713264	2.49829504		Total impact estimate H/8Y	Total impact estimate H/Y	Total impact estimate H/M
CFV	16	0.10590	0.244147157	0.025856199	0.361452819		9.19	1.15	0.10
UFV	18	0.11914	0.224080268	0.026697411	0.373212414		Total impact estimate M/8Y	Total impact estimate M/Y	Total impact estimate M/M
DFV	16	0.10590	0.163879599	0.017355531	0.242619015		551.42	68.93	5.74
							Total impact estimate H/period	Total impact estimate M/period	
							0.57	34.46	

	DM								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	40	0.05295	0.054429029	0.00288213	0.040290303	137.6797865	11	12.51634423	1
DD	36	0.47657	0.06189968	0.02949945	0.412383102				
UD	36	0.09531	0.003201708	0.000305167	0.004266032				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		Analysis results		
DBR	20	0.39714	0.036286019	0.014410651	0.201451515		Total impact estimate FP/8Y	Total impact estimate FP/Y	Total impact estimate FP/M
CC	52	0.51628	0.002134472	0.001101991	0.015405116		12.52	1.56	0.13
DC	358	7.10882	0.18036286	1.28216648	17.92385231		Total impact estimate H/8Y	Total impact estimate H/Y	Total impact estimate H/M
UC	358	7.10882	0.106723586	0.758678391	10.60582977		157.58	19.70	1.64
CFV	125	0.82738	0.186766275	0.154525976	2.160172497		Total impact estimate M/8Y	Total impact estimate M/Y	Total impact estimate M/M
UFV	127	0.84061	0.109925293	0.092404767	1.291758465		9454.85	1181.86	98.49
DFV	130	0.86047	0.022411953	0.019284842	0.269589528		Total impact estimate H/period	Total impact estimate M/period	
							9.85	590.93	

Prediction model for the fact-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	10	0.01324	0.02006689	0.000265646	0.003713556	7.130647133	10	0.713064713	1
DD	17	0.22505	0	0	0				
UD	10	0.02648	0.056856187	0.001505327	0.021043486				
CBR	5	7.94281	0.033444816	0.265645878	3.713556358				
DBR	5	0.09929	0.003344482	0.000332057	0.004641945		**Analysis results**		
CC	20	0.19857	0.013377926	0.002656459	0.037135564		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
DC	52	1.03257	0.010033445	0.010360189	0.144828698		0.71	0.09	0.01
UC	39	0.77442	0.230769231	0.178713264	2.49829504		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
CFV	12	0.07943	0.244147157	0.019392149	0.271089614		8.98	1.12	0.09
UFV	13	0.08605	0.224080268	0.019281463	0.269542299		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
DFV	11	0.07281	0.163879599	0.011931927	0.166800573		538.65	67.33	5.61
							Total impact estimate H/period	**Total impact estimate M/period**	
							0.56	33.67	
	DM								
		Impact*FP/MO		**Size * weight**	**S+W * Productivity level**	**Sum of all effort**		**Total effort / amount of scenarios included**	
	Impact	**Size in FP**	**Normalized Weight**	**Normalized size**	**Effort/scenario**	**Total effort**	**Amount of scenarios included**	**Average effort**	**Amount of modifications expected**
CD	40	0.05295	0.054429029	0.00288213	0.040290303	156.1286965	11	14.19351786	1
DD	96	1.27085	0.06189968	0.078665201	1.099688271				
UD	96	0.25417	0.003201708	0.000813778	0.011376086				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		Analysis results		
DBR	20	0.39714	0.036286019	0.014410651	0.201451515		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
CC	77	0.76450	0.002134472	0.001631794	0.022811422		14.19	1.77	0.15
DC	662	13.14535	0.18036286	2.370933547	33.14410678		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
UC	485	9.63066	0.106723586	1.02781849	14.36823306		178.70	22.34	1.86
CFV	74	0.48981	0.186766275	0.091479378	1.278822118		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
UFV	104	0.68838	0.109925293	0.075670046	1.057817956		10721.78	1340.22	111.69
DFV	72	0.47657	0.022411953	0.010680835	0.149311123		**Total impact estimate H/period**	**Total impact estimate M/period**	
							11.17	670.11	

Prediction model for the decision-oriented architecture containing the cases COPD & Diabetes Mellitus

	COPD								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	10	0.01323802	0.02006689	0.000265646	0.003713556	7.385706563	10	0.738570656	1
DD	25	0.33095049	0	0	0				
UD	54	0.142970612	0.056856187	0.008128764	0.113634825				
CBR	5	7.942811755	0.033444816	0.265645878	3.713556358				
DBR	5	0.099285147	0.003344482	0.000332057	0.004641945		Analysis results		
CC	25	0.248212867	0.013377926	0.003320573	0.046419454		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
DC	55	1.092136616	0.010033445	0.010957892	0.1531842		0.74	0.09	0.01
UC	45	0.893566322	0.230769231	0.206207613	2.882648123		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
CFV	8	0.052952078	0.244147157	0.012928099	0.180726409		9.30	1.16	0.10
UFV	8	0.052952078	0.224080268	0.011865516	0.165872184		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
DFV	8	0.052952078	0.163879599	0.008677765	0.121309508		557.92	69.74	5.81
							Total impact estimate H/period	**Total impact estimate M/period**	
							0.58	34.87	

	DM								
		Impact*FP/MO		Size * weight	S+W * Productivity level	Sum of all effort		Total effort / amount of scenarios included	
	Impact	Size in FP	Normalized Weight	Normalized size	Effort/scenario	Total effort	Amount of scenarios included	Average effort	Amount of modifications expected
CD	40	0.05295	0.054429029	0.00288213	0.040290303	155.3352935	11	14.12139032	1
DD	164	2.17104	0.06189968	0.134386385	1.878634129				
UD	164	0.43421	0.003201708	0.001390204	0.019434146				
CBR	20	31.77125	0.235859125	7.493538518	104.7547879		Analysis results		
DBR	20	0.39714	0.036286019	0.014410651	0.201451515		**Total impact estimate FP/8Y**	**Total impact estimate FP/Y**	**Total impact estimate FP/M**
CC	96	0.95314	0.002134472	0.002034445	0.028440214		14.12	1.77	0.15
DC	675	13.40349	0.18036286	2.417492665	33.79497292		**Total impact estimate H/8Y**	**Total impact estimate H/Y**	**Total impact estimate H/M**
UC	422	8.37967	0.106723586	0.894308047	12.50184403		177.79	22.22	1.85
CFV	62	0.41038	0.186766275	0.076644884	1.071445558		**Total impact estimate M/8Y**	**Total impact estimate M/Y**	**Total impact estimate M/M**
UFV	90	0.59571	0.109925293	0.065483694	0.915419385		10667.30	1333.41	111.12
DFV	62	0.41038	0.022411953	0.009197386	0.128573467		**Total impact estimate H/period**	**Total impact estimate M/period**	
							11.11	666.71	

Appendix J – Paper: A Classification of Modification Categories for Business Rules

Paper: A Classification of Modification Categories for Business Rules – Submitted and accepted for the 28th bled eConference (June 7 – 10, 2015; Bled, Slovenia)

28th Bled eConference

#WellBeing

June 7 - 10, 2015; Bled, Slovenia

A Classification of Modification Categories for Business Rules

Martijn Zoet

HU University of Applied Sciences Utrecht, Nijenoord 1, 3552 AS Utrecht, Netherlands, martijn.zoet@hu.nl

Koen Smit

HU University of Applied Sciences Utrecht, Nijenoord 1, 3552 AS Utrecht, Netherlands, koen.smit@hu.nl;
Utrecht University Graduate School of Natural Sciences, Princetonplein 5, 3584 CC Utrecht, The Netherlands, k.smit@students.uu.nl

Sam Leewis

HU University of Applied Sciences Utrecht, Nijenoord 1, 3552 AS Utrecht, Netherlands, sam.leeuwis@hu.nl

Abstract

Business rules play a critical role in an organization's daily activities. With the increased use of business rules (solutions) the interest in modelling guidelines that address the manageability of business rules has increased as well. However, current research on modelling guidelines is mainly based on a theoretical view of modifications that can occur to a business rule set. Research on actual modifications that occur in practice is limited. The goal of this study is to identify modifications that can occur to a business rule set and underlying business rules. To accomplish this goal we conducted a grounded theory study on 229 rules set, as applied from March 2006 till June 2014, by the National Health Service. In total 3495 modifications have been analysed from which we defined eleven modification categories that can occur to a business rule set. The classification provides a framework for the analysis and design of business rules management architectures.

Keywords: Business Rules Management, Business Rules Modifications, Business Rule Architectures, Change Management.

Introduction

Laws, regulation, protocols, standards, are each example of rules that organizations are forced to act in accordance with (Shao and Pound 1999; Bajec and Krisper 2005; Tarantino, 2008). Each of the previous mentioned form of rules is applied to guide/constrain entities, such as individuals, teams and organizations to act in accordance with internal or external provided criteria. Take, for example, a general practice. From a regulatory and legislative point of view, business rules are used to restrict access to patient information, force general practitioners to be more transparent in their decision-making and constrain the incentive system general practices can apply (Blomgren and Sunden, 2008; King and Green, 2012). In addition to externally provided criteria, organizations themselves also create additional rules, which they want teams and individuals to comply to. For example a general practitioner states rules on how a specific decision must be made.

To prevent individuals and teams in an organization deviating from desired behaviour, laws regulation, protocols and standards are translated to business rules. A business rule is (Morgan 2002): "*a statement that defines or constrains some aspect of the business intending to assert business structure or to control the behaviour of the business.*" In addition to faster changing and increased amounts of laws, regulation, protocols and standards implemented, trends like higher demanding customers and, faster changing customer's demands give rise to an increase in the amount of business rules as well as an increased pace of modifications to these business rules. Thereby increasing the need to decompose and structure business rules to accommodate for expected or unexpected modifications and making it possible to rapidly modify them when necessary.

Scientific research with respect to business rules decomposition and structuring to address modifiability in terms of anomalies such as insertion, updates and deletion is scarce (Vanthienen and Snoeck 1993; Von Halle and Goldberg, 2010; Anonymous et al., 2012). Current research that is conducted mostly applies experimental research methods and applies theoretical modifications that can occur to a business rule set. This paper extends understanding of business rules modification by addressing the type of modifications that can occur to a business rule (set). Dissimilar to previous research we do not approach this from a theoretical point of view, but analyse eight years of actual modifications to a business rule set. Within this scope, the research question addressed is: *"Which modifications can impact a business rule set?"* Answering this question will help practitioners better manage business rules that support analytical activities in business processes.

The remainder of this paper proceeds as follows. The next section provides a context by describing business rules, separation of concerns, and theory on modification that can occur to business rules. The third section describes the data collection and data analysis. Section four presents the analysis and results of the grounded theory study. The final section summarizes the study's core findings, contributions as well as its limitations.

Literature

Evolution of information systems is characterized by functional or non-functional modifications that occur to the information system. Modifications are necessary because of changes in 1) the operating environment, 2) the implementation technology, and/or 3) in stakeholder needs. In this work, we adopt the concept of modifiability as defined by Bass et al, (2012): "*The ability to incorporate anomalies to an information system made possible by the minimal number of changes*." An information system cannot be engineered to adept to every possible modification. Qumer and Henderson (2006, p3) state that a system must be able to accommodate *"changes rapidly, following the shortest time span, using economical, simple and quality instruments in a dynamic environment and applying updated prior knowledge and experience to learn from the internal and external environment*." From a technical and economic perspective it is impossible to build a system that can cope with every modification possible.
To increase the number of modifications an information system can cope with, multiple design principles have been proposed and validated. One important principle in information systems and computer sciences which enables organizations to manage change is separation of concerns (Versendaal, 1991,

Van der Aalst, 1996, Weske, 2007). The advantages of applying the separation of concerns principle are simplified development and simplified maintenance. Development and maintenance are simplified because concerns are separated and therefore can be modified independently of each other without having to know the other concern's details. Although several variants of separation of concerns have been proposed, various authors agree on a general evolution of information technology architecture which is depicted in Figure 1. This general evolution follows the decoupling of operating systems from applications, database from applications, the user interface from the application and in the 90's the workflow from the application. With each of the concerns separated, research streams started to focus on modifications within the individual concerns answering questions like: *"which modifications can occur to a database?"*, *"how to cope with change to databases?"*, *"which modifications can occur to user interface?"*, and *"which modifications can occur to workflows?"* In the workflow (Business Process Management) community this research has led to the classification of different type of business processes, e.g. workflow processes, adaptive case management and, straight through processes. Based on the change behaviour of the process a different design paradigm is applied to design and execute the business process. For example a process which is highly structured applies workflow management while a process which is late-structured applies adaptive case management (Van der Aalst, 1996). This example illustrates that organizations need to make a decision on what set of anticipated modifications should be defined to cope with to be able to utilize a stable product and/or service (Mannaert and Verelst, 2009).

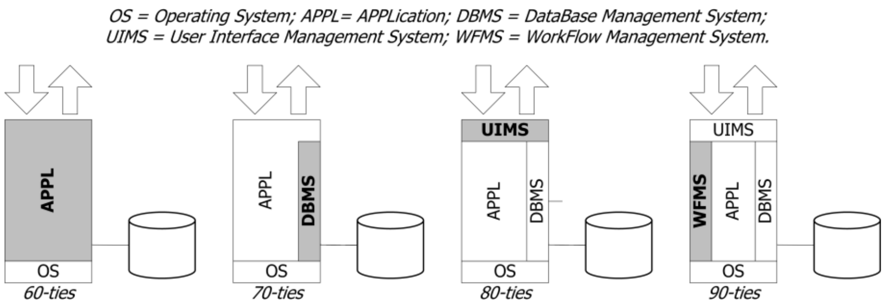


Figure 1 - Evolution of Information Technology Architecture (Van der Aalst, 1996)

The next wave of separation followed around the 2000's where research and practice started to propose the separation of business rules from the application and create a separate layer (Chapin et al, 2001; Boyer and Mili, 2011, Graham, 2006). Chapin et al. (2001) states that among the other concerns (application, databases, user interface and workflow) business rule modifications are the most frequent and have the highest impact on software and business processes. Additionally, the authors identified that the other concerns rely extensively on the support of business rules and that modifications to business rules are commonly the most significant in terms of effort required, thereby indicating the need to properly manage modifications to business rules.

Scientific research with respect to business rules modeling guidelines that address manageability in terms of anomalies such as insertion, updates and deletion is scarce (Vanthienen and Snoeck 1993; Zoet et al. 2011). Some research regarding this subject can be identified in the knowledge management community (e.g. Vanthienen and Snoeck 1993), the business rules management community (e.g. Zoet et al, 2011) and the software engineering community (Chapin et al, 2001). Chapin et al. (2001) proposes that modifications to business rules are either 1) Reductive, 2) Corrective, or 3) Enhancive of nature. The first modification archetype, Reductive, comprises reducing the business logic implemented. The second modification archetype, Corrective, comprises refinement and making more specific of implemented business rules. The third modification archetype, Enhancive, comprises changing and adding upon the repertoire of software implemented business rules to enlarge or extend their scope. Although Chapin et al. (2001) proposes a theoretical set of modification archetypes they do not elaborate in detail how they affect business rules and how to manage / design business rules in such a way that one can cope with change. Vanthienen and Snoeck (1993) propose in their study, based on relational theory and database normalization, guidelines to factor knowledge thereby improving maintainability. VanThienen and Snoeck's (1993) research showed that normalization has a positive effect on the average number of business rules affected when anomalies occur. Thus, when anomalies such as updates, inserts and deletes occur, the number of business rules affected in third normal form is less than the number of rules affected in first normal form. However, their research is based on decision tables instead of business rules in general. Building on the work of VanThienen and Snoeck (1993), Zoet et al. (2011) developed a normalization procedure based on representational difference analysis of existing business rules modelling languages, relational theory and database normalization. The procedure consists of three steps: 1) apply first normalization form, 2) apply second normalization form and 3) apply third normalization form. This research strengthens the conclusions drawn by VanThienen and Snoeck (1993) that normalization has a positive effect on the average number of business rules affected when anomalies occur. A contribution from practice which has the same focus is The Decision Model (Goldberg, 2010). Von Halle and Goldberg's (2010) normalization procedure also is based on the ideas proposed by VanThienen and Snoeck (1993), showing similarities with the solution proposed by Zoet et al. (2011). An important difference between the method proposed by Von Halle and Goldberg (2010) and Zoet et al. (2011) is that the latter supports multiple business rules formalism like decision tables, event condition action languages while Von Halle and Goldborg (2010) focus only on decision tables.

Previous research provides conceptual and theoretical understanding of modifications that can occur to business rules. However, these studies applied controlled experiments based on small case studies and/or theorized modifications that can occur to business rules. Thereby focusing on generalization from construct or theory to collected data and generalization from theory to theory (Lee and Baskerville, 2003). We feel that this represents a notable gap, and we argue that there is a need to generalize from collected data to constructs and theory. Differently stated, collecting modifications which occurred to business rule (sets) and generalize this to a theoretical framework. A research method to generalize from data to constructs and theory is grounded theory (Glaser, 1978), which therefore will be adapted for this research.

Data collection and analysis

The goal of this research is to identify and define the most common set of anticipated modifications (Manneart and Verelst, 2009) that impact the design of a business rule set. To accomplish this goal a research approach is needed that can: 1) identify modifications applied to the business rule and 2) identify similarities and dissimilarities between types of modifications. An additional criterion is that the set of anticipated modifications is grounded in practice. Each of these goals are realized when applying grounded theory. The purpose of grounded theory is to (Glaser, 1978): "*explain with the fewest possible concepts, and with the greatest possible scope, as much variation as possible in the behaviour and problem under study*." Theory states that the first selection of respondents and documentation is based on the phenomenon studied at a group of individuals, organization, information technology, or community that best represents this phenomenon (Glaser, 1978). Our choice for a case was based on theoretical and pragmatic criteria. Our theoretical criterion was: "the case site should deal with business rules, regulation, laws or policies that change frequently." Our pragmatic criterion was: "the case site should have kept different versions of the business rules, regulation, laws or policies." Based on these criteria the British National Health Service (NHS) was selected.

Data Collection

The NHS is built up from four different health care systems, England, Northern Ireland, Scotland, and Wales. These regions combined provide healthcare services for over 64.1 million UK residents. The NHS employs more than 1.6 million people, which makes it one of the top five workforces in the world in terms of scale. Over one million patients every 36 hours make use of NHS services. A significant part of healthcare management in the UK by the NHS focuses on the management of chronic diseases. In April 2004 the NHS introduced the Quality and Outcomes Framework (QOF) as part of the new General Medical Services (GMS) contract. The QOF is a Pay-for-Performance-scheme covering a range of clinical, organizational, and patient areas in primary care. It is established to reward practices for the provision of high quality care and helps fund further improvements in the delivery of clinical care. The QOF includes the measurement of different domains, however, due to the scope of this study only the clinical and public health domains are considered. The NHS manages the QOF which is a Pay-for-Performance-scheme in that comprises to 25 clinical conditions. For each individual condition they create business rules to select when a clinic must be paid for the treatment of the patient (Gilliam and Siriwardena, 2011).

The business rule sets are updated twice a year to accommodate the introduction of new insights revealed by empirical research and/or changes in law and regulations. At the time of writing, the combination of these domains contain 25 clinical conditions, with a large amount of underlying indicators, which make up for 80 percent of the commonly encountered health issues in primary care (Gilliam and Siriwardena, 2011). Examples of clinical conditions as part of the QOF are: Heart Failure (HF), Diabetes Mellitus (DM), and Chronic Obstructive Pulmonary Disease (COPD).

Of the 25 clinical conditions, 16 have been analysed. The selection of the 16 clinical diseases has been done semi-randomly. First we selected the two clinical conditions with the largest set of business rules: Coronary heart disease and Diabetes Mellitus. After which fourteen additional diseases have been randomly selected: Chronic Obstructive Pulmonary Disease (COPD), Cancer, Asthma, Obesity, Atrial Fibrillation, Chronic Kidney Disease (CKD), Cardiovascular Disease (CVD), Blood Pressure, Contraception, Osteoporosis, Peripheral Arterial Disease (PAD), Cervical Screening, Cytology, and Dementia. For each disease the different versions of the business rules have been collected. At the time of writing the QOF is at version 29. However, version 1 till 8 and 20 cannot be retrieved, not even by the NHS itself. Therefore our analysis included versions 9 till 19 and 21 till 29. In total, the data collected comprises 229 versions (documents) of clinical conditions, from which the publication ranges from March 2006 until June 2014. In total, 16 out of 25 clinical conditions have been fully coded.

Data Analysis

The goal of the first phase of coding (open coding) was to establish a coding scheme. To develop the coding scheme, first, each individual researcher read and coded two consecutive versions of a randomly selected clinical condition. In open coding the unit of analysis are business rule sets and individual business rules (Boyatizs, 1998). For examples of open coding in our study see Table 1. After both researchers finished, the coded parts were discussed and compared to understand the process and agree on the elements that had to be coded. The result of this first cycle was a coding scheme. The goal of the second cycle of coding was to refine the coding scheme. Therefore two researchers, one researcher from the first cycle and one new researcher, coded multiple consecutive versions of multiple clinical conditions. The clinical conditions were randomly selected from the pool of clinical conditions. After both researchers finished, the coded parts were discussed among the three researchers, including the researchers from the first round. In these sessions coding was compared to understand the process and agree on the elements that had to be coded. The result of this second cycle was an improved coding scheme. The goal of the third cycle was to code the remainder of the 229 versions of clinical conditions and identify the modifications. This cycle was performed by two researchers. The third researcher acted as reliability coder which randomly selected modifications and compared his coding to those of the other two researchers. An extract of the coding scheme is shown in first row of Table 1. Open coding resulted in 3495 references classified to eleven modification categories: A) create decision, B) delete decision, C) update decision, D) create business rule, E) delete business rule, F) create condition, G) delete condition, H) update condition, I) create fact value, J) delete fact value, and K) update fact value. An overview of all modifications per modification category is provided in Figure 2.

Table 1 - Examples of open coding: clinical condition COPD (Health and Social Care Information Centre, 2007)

Text Fragments Version A	Text Fragments Version B	Open Coding
Clinical indicator COPD8	Clinical indicator COPD13	**Update decision**
If COPDSPIR_DAT >= (COPD_DAT – 3 months) AND If COPDSPIR_DAT <= (COPD_DAT + 12 months)	If COPDSPIR_DAT >= (COPD_DAT – 3 months)	**Delete business rule**

Read codes v2: (8I2M., 8I3b., 8I6L.) **SNOMED-CT:** (415571003, 415572005, 415570002) **CTV3:** (XaJz4, XaK27, XaK2A)	**Read codes v2:** (8I2M., 8I3b., 8I6L., 8I6d.) **SNOMED-CT:** (415571003, 415572005, 415570002, 279261000000103) **CTV3:** (XaJz4, XaK27, XaK2A, XaMh9)	**Create fact value**

The second phase of coding is axial coding. To support this process Glasser (1978) formulated 18 coding families. Glaser (1992) stresses that researchers should not blindly apply each individual coding family to data at hand. The application for a specific coding family must emerge first from the research question and secondly from the data. The purpose of applying coding families in our research was to determine mutual exclusivity between and completeness of the modifications that can be applied to business rules (sets). To test for mutual exclusivity and completeness we therefore applied coding families that searched for end stages, clusters, conceptual ordering, conformity, and structural ordering: the ordering and elaboration family and means-goal family (Glaser, 1978). Applying the mentioned coding families served as a basis for the business rule modifiability framework, which is depicted in Table 2.

	A - Create decision	B - Delete decision	C - Update decision	D - Create business rule	E - Delete business rule	F - Create condition	G - Delete condition	H - Update condition	I - Create fact value	J - Delete fact value	K - Update fact value
Version 9				1		1	3		72	49	13
Version 10									15	12	10
Version 11									21	40	12
Version 12			8	3		2			9	8	14
Version 13									3		10
Version 14	12	4	1	7	4	2		9	16	4	10
Version 15									13	150	10
Version 16							157		39	310	10
Version 17									2		12
Version 18									19		10
Version 19	16	20	9	26	64	81		5	4	7	11
Version 21			1	4					106	16	12
Version 22	4	2	7	25	86	83	12	2	59	28	20
Version 23				1					99	12	12
Version 24									107	2	12
Version 25	33	32	52	19	28	10	21	18	29	33	93
Version 26				4					77	4	25
Version 27				8					113	7	44
Version 28	2	16	10	13	70	27	2	534	30	15	26
Version 29								8	16		14
Total	67	74	88	111	252	206	195	576	849	697	380
Grand total	3495										

Figure 2 - Amount of modifications per modification category

Furthermore, it is interesting to report on what caused the large amount of modifications for some versions of the business rule sets. For example, we know that the large amount of modifications concerning the modification type Delete fact value in version 16 are caused by the phase out of a medical information system containing those fact values. However, it is beyond the scope of this study to fully elaborate on these causes. More research on the causes of the large amount of modifications for some versions can be found in the work of Gilliam and Siriwardena (2011).

Results

In this section the identified modification categories are presented elaborated upon. To ground the modification categories, our research includes an example of a business rule set within the context of the QOF which is provided in Figure 3 and Figure 4.

Table 2 - Business rules modification framework

	Decision	Business Rule	Condition	Fact value
Create	CD	CBR	CC	CFV
Update	UD		UC	UFV
Delete	DD	DBR	DC	DFV

A. The first modification is identified as: "*create decision.*" This modification adds an additional decision or sub-decision to the already existing set of business rules. This includes all underlying variables such as new business rules and new fact values. This particular modification category is observed 67 times out of 3495 observations.

B. The second modification is identified as "*delete decision.*" This modification deletes a decision that, for example became obsolete. This includes all underlying variables such as new business rules and new fact values. This particular modification category is observed 74 times out of 3495 observations.

C. The third modification is identified as "*Update decision.*" This modification solely updates the name (label) of a specific concept without changing underlying logic. An example regarding the QOF is a decision currently labelled as: *Amount of achievement points obtained*, which is modified into: *Amount of achievement percentage obtained*. This particular modification category is observed 88 times out of 3495 observations.

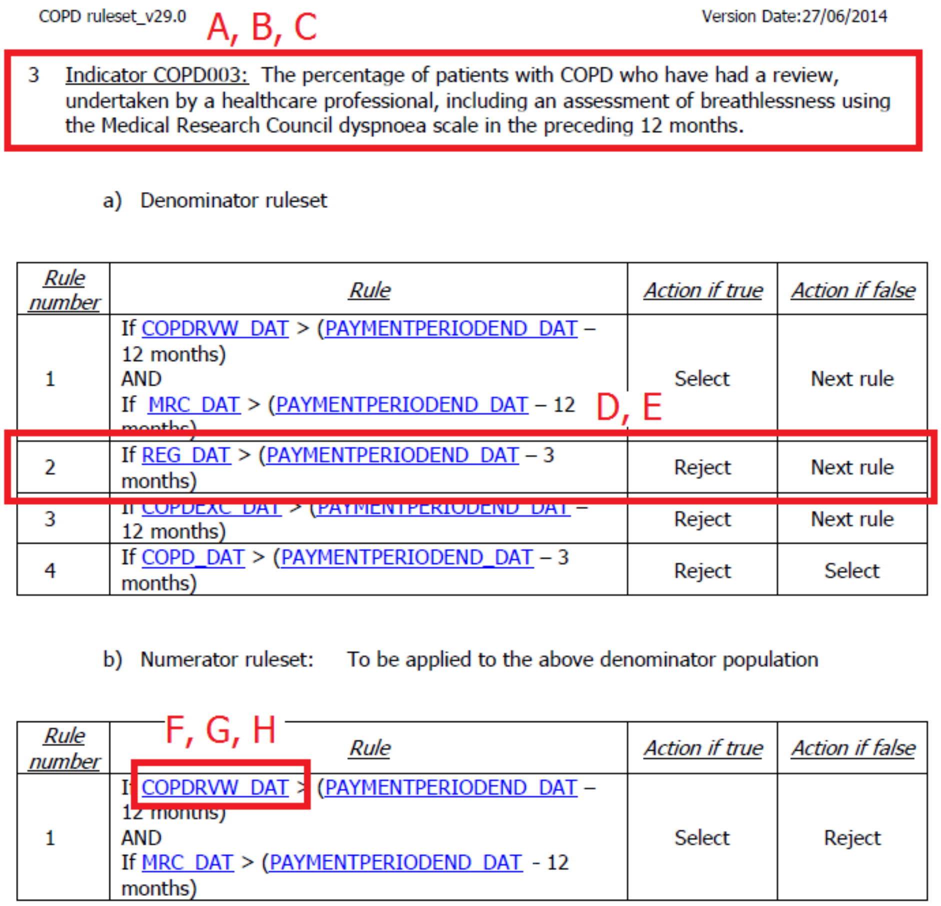
D. The fourth modification is identified as "*create business rule.*" This modification creates a new business rule within the business rule set of a given decision, including one or more conditions and one conclusion. This particular modification category is observed 111 times out of 3495 observations.

E. The fifth modification is identified as "*delete business rule.*" This modification deletes an existing business rule within the business rule set of a given decision, including one or more conditions and one conclusion. This particular modification category is observed 252 times out of 3495 observations.

F. The sixth modification is identified as "*create condition.*" This modification creates a new condition to be used by existing or new conclusions. An example regarding the QOF is the addition of a ratio to calculate the conclusion *final points achieved*. The condition *relative achievement ratio* is added in the calculation to balance inequalities of register list sizes of general practices. This particular modification category is observed 206 times out of 3495 observations.

G. The seventh modification is identified as "*delete condition.*" This modification deletes an existing condition from a given ruleset. An example regarding the QOF is the deletion of the condition *higher threshold*. In the new situation, GP's will or will not achieve the minimum threshold and will not be able to attain bonus achievement over a certain achievement percentage anymore. This particular modification category is observed 195 times out of 3495 observations.

H. The eight modification is identified as "*Update condition.*" This modification solely updates the name (label) of a condition. An example regarding the QOF is a condition currently labelled as: *REF DAT*, which is modified into: *ACHIEVEMENT DAT.* This particular modification category is observed 576 times out of 3495 observations.



COPD ruleset_v29.0 Version Date:27/06/2014

3 Indicator COPD003: The percentage of patients with COPD who have had a review, undertaken by a healthcare professional, including an assessment of breathlessness using the Medical Research Council dyspnoea scale in the preceding 12 months.

a) Denominator ruleset

Rule number	Rule	Action if true	Action if false
1	If COPDRVW_DAT > (PAYMENTPERIODEND_DAT – 12 months) AND If MRC_DAT > (PAYMENTPERIODEND_DAT – 12 months)	Select	Next rule
2	If REG_DAT > (PAYMENTPERIODEND_DAT – 3 months)	Reject	Next rule
3	If COPDEXC_DAT > (PAYMENTPERIODEND_DAT – 12 months)	Reject	Next rule
4	If COPD_DAT > (PAYMENTPERIODEND_DAT – 3 months)	Reject	Select

b) Numerator ruleset: To be applied to the above denominator population

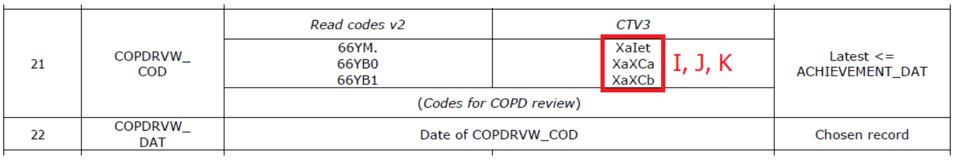
Rule number	Rule	Action if true	Action if false
1	If COPDRVW_DAT > (PAYMENTPERIODEND_DAT – 12 months) AND If MRC_DAT > (PAYMENTPERIODEND_DAT - 12 months)	Select	Reject

Figure 3 - Example business rule document of the QOF 1/2 (Health and Social Care Information Centre, 2014)

I. The ninth modification is identified as "*create fact value.*" This modification creates a new fact value for its parent condition or conclusion. An example regarding the QOF is the addition of a fact value under a new condition labelled as *maximum raw points achieved*. The fact value added operates as an upper threshold and is set to *550*. This particular modification category is observed 849 times out of 3495 observations.

J. The tenth modification is identified as "*delete fact value.*" This modification deletes an existing fact value from its parent condition or conclusion. An example regarding the QOF is deleting a fact value from the conclusion *patient registration status*. From the four available conclusions this ruleset can generate, the fact value *previously registered* is deleted, leaving the possibility to generate three conclusions. This particular modification category is observed 697 times out of 3495 observations.

K. The eleventh modification is identified as "*Update fact value.*" A fact value is a possible value or fixed value of its parent condition. An example regarding the QOF is renaming the fact values of the condition *Upper threshold* from 70 achievement percentage to 80 achievement percentage. This particular modification category is observed 380 times out of 3495 observations.



		Read codes v2	CTV3	
21	COPDRVW_COD	66YM. 66YB0 66YB1	XaIet XaXCa XaXCb I, J, K	Latest <= ACHIEVEMENT_DAT
		(Codes for COPD review)		
22	COPDRVW_DAT	Date of COPDRVW_COD		Chosen record

Figure 4 - Example business rule document of the QOF 2/2 (Health and Social Care Information Centre, 2014)

The eleven identified modifications have a hierarchical structure. In this structure the highest level of is a decision followed by business rules, conditions and fact values. The existence of a hierarchy indicates a cause and effect relationship between the different elements. For example, when a new decision is created the possibility exist that also new business rules, conditions and fact values must be created. The data shows this is not always the case since underlying hierarchical elements are reused. Due to size constraints we decided to omit a full overview of this phenomenon.

Conclusion & discussion

Business rules are widely applied, standalone and embedded in smart objects. Therefore they have become a separate concern in information system design. As a result they also have to be managed separately. From a technical and economic perspective it is impossible to build an information system that can cope with every modification possible. Therefore a choice has to be made which defined set of anticipated modifications the system must be stable to cope with (Mannaert and Verelst, 2009). The purpose of this research is to define the set of anticipated modifications a business rule set must be able to cope with. To be able to this we addressed the following research question: "*Which modifications can impact a business rule set.*" In order to answer this question, we conducted a grounded theory study on modifications occurring in the business rules applied for payment to primary care organizations in the United Kingdom by the NHS, the QOF payment schemes. In total we analysed 3495 modifications that occurred during the last eight years resulting in a set of modification types that can occur to business rules (sets).

From the data, we identified eleven types of modifications: A) create decision, B) delete decision, C) update decision, D) create business rule, E) delete business rule, F) create condition, G) delete condition, H) update condition, I) create fact value, J), delete fact value, and K) update fact value. From a research perspective, our study provides a generalization from collected data to constructs and theory (Lee and Baskerville, 2003). Thereby it provides a fundament for further research which can focus on building business rule architectures that can optimally cope with the identified modifications. From a practical perspective, our study provides an overview of the modifications that can occur to business rules which can help organizations to construct test scenarios that help information systems to cope with future modifications.

Several limitations may affect our results. The first limitation is the related to sample size. While the sample size of business rules modifications (3495) is representative, the modification types are all derived from one organization, which may limit generalization. The second limitation is related to the first, our sampling strategy. Our research was applied to business rule sets from the medical industry. And while the medical industry is known for the relatively high amount of utilization of business rules, several other industries are interesting to include as well; for example the financial or governmental industries. The omission of modifications to business rules from other industries may also limit generalization. Adding business rule sets from other industries will be a part of further research.

References

Appleton, D. (1984). Business Rules: The Missing Link. *Datamation*, 145-150.

Arnott, D., & Pervan, G. (2005). A critical analysis of decision support systems research. *Journal of Information Technology, 20*(2), 67-87.

Bajec, M., & Krisper, M. (2005). A methodology and tool support for managing business rules in organisations. *Information Systems, 30*, 423-443.

Bass, L., Clements, P., & Kazman, R. (2012). *Software Architecture in Practice (3rd Edition)* New York: Addison-Wesley Professional.

Blomgren, M., & Sunden, E. (2008). Constructing a European Healthcare Market: The private healthcare company Capio and the strategic aspect of the drive for transparency. . *Social Science & Medicine, 67*(10), 1512-1520.

Boyatzis, R. (1998). *Transforming qualitative information: Thematic analysis and code development*. Thousand Oaks, CA: Sage.

Boyer, J., & Mili, H. (2011). *Agile Business Rules Development: Process, Architecture and JRules Examples*. Heidelberg: Springer.

Gillam, S., & Siriwardena, N. (2011). *The Quality and Outcomes Framework: QOF-transforming general practice*: Radcliffe Publishing.

Glaser, B. (1978). *Theoretical Sensitivity: Advances in the Methodology of Grounded Theory*. Mill Valley, CA: Sociology Press.

Glaser, B., & Strauss, A. (1967). *Discovery of Grounded Theory. Strategies for Qualitative Research*. Mill Valley: Sociology Press

Graham, I. (2006). *Business Rules Management and Service Oriented Architecture*. New York: Wiley.

Health and Social Care Information Centre. (2007). Chronic Obstructive Pulmonary Disease Rule Set Version 10. http://webarchive.nationalarchives.gov.uk/20110426140805/http://pcc.nhs.uk/business-rules-v10-0: Health and Social Care Information Centre.

Health and Social Care Information Centre. (2014). Chronic Obstructive Pulmonary Disease Rule Set Version 29 http://www.hscic.gov.uk/qofbrv29: Health and Social Care Information Centre.

King, R., & Green, P. (2012). Governance of primary healthcare practices: Australian insights. *Business Horizons, 55*(6), 593-608.

Kovacic, A. (2004). Business renovation: business rules (still) the missing link. *Business Process Management Journal, 10*(2), 158-170.

Lee, A., & Baskerville, R. (2003). Generalizing Generalizability in Information Systems Research. *Information Systems Research, 14*(3), 221-223.

Liao, S.-H. (2004). Expert System Methodologies and Applications - A Decade Review from 1995 to 2004. *Expert Systems with Applications, 28*(1), 93-103.

Mannaert, H., & Verelst, J. (2009). *Normalized Systems: Re-creating Information Technology Based on Laws for Software Evolvability*. Antwerpen: Koppa Digitale Media.

Morgan, T. (2002). *Business rules and information systems: aligning IT with business goals*. London: Addision-Wesley.

Nelson, M. L., Peterson, J., Rariden, R. L., & Sen, R. (2010). Transitioning to a business rule management service model: Case studies from the property and casualty insurance industry. *Information & Management, 47*(1), 30-41.

Nelson, M. L., Rariden, R. L., & Sen, R. (2008). A Lifecycle Approach towards Business Rules Management. *Proceedings of the 41st Hawaii International Conference on System Sciences*, Hawaii.

Qumer, A., & Henderson-Seller, B. (2006). Measuring agility and adoptability of agile methods: a 4-dimensional analytical tool. *Proceedings of the IADIS International Conference Applied Computing*, Barcelona.

Shao, J., & Pound, C. J. (1999). Extracting business rules from information systems. *BT Technology Journal*, 179-186.

Tarantino, A. (2008). *Governance, Risk, and Compliance Handbook*. New Jersey: Wiley.

Taylor, J. (2013). The Decision Management Manifesto: An Introduction. In Decision Management Solutions (Ed.). decisionmanagementsolutions.com/decision-management-manifesto

Van der Aalst, W. (1996). Three Good Reasons for Using a Petri-net-based Workflow Management System*. Proceedings of the International Working Conference on Information and Process Integration in Enterprises*, Cambridge.

Van Thienen, J., & Snoeck, M. (1993). *Knowledge factoring using normalization theory*. Paper presented at the International Symposium on the Management of Industrial and Corporate Knowledge, Compiegne.

Vanthienen, J., & Wets, G. (1994). Restructuring and optimizing knowledge representations*. Proceedings of the Sixth International Conference on Tools with Artificial Intelligence, 1994.*

Versendaal, J. (1991). *Separation of the User Interface and Application.* PhD, Technische Universiteit Delft, Rotterdam.

Von Halle, B., & Goldberg, L. (2010). *The Decision Model: A Business Logic Framework Linking Business and Technology*. Boca Raton, FL: Auerbach Publications.

Weske, M. (2007). *Business Process Management - Concepts, Languages, Architectures*. New York: Springer.

Zoet, M., Versendaal, J., Ravesteyn, P., & Welke, R. (2011). Alignment of Business Process Management and Business Rules*. Proceedings of the European Conference on Information Systems*.

Appendix K – Paper B: Utilizing change effort prediction to analyse modifiability of business rules architectures

Paper: Utilizing change effort prediction to analyse modifiability of business rules architectures

<Conference>

<Conference>

<Conference track / theme>

<Conference date month, dates, year; location, country>

Utilizing change effort prediction to analyse modifiability of business rules architectures

Koen Smit

Utrecht University Graduate School of Natural Sciences,
Princetonplein 5, 3584 CC Utrecht, The Netherlands, k.smit@students.uu.nl

Martijn Zoet

HU University of Applied Sciences Utrecht, Nijenoord 1,
3552 AS Utrecht, The Netherlands, martijn.zoet@hu.nl

Sjaak Brinkkemper

Utrecht University Graduate School of Natural Sciences,
Princetonplein 5, 3584 CC Utrecht, The Netherlands, s.brinkkemper@uu.nl

Ronald Batenburg

Utrecht University Graduate School of Natural Sciences,
Princetonplein 5, 3584 CC Utrecht, The Netherlands, r.s.batenburg@uu.nl

Abstract

Business rules (BR's) play a critical role in an organization's daily activities. With the increased use of BR (solutions) the interest in modifiability guidelines that address the manageability of BR's has increased as well. Increasing pressure from external and internal requirements on compliancy, amongst others, leads to the utilization of increasingly larger amounts of BR's as part of products and/or services. The current literature gap on the utilization of logical architectures applied in the BRM domain triggered us to explore and propose the concept of a Business Rule Architecture (BRA). As this is a very broad spectrum to elaborate upon we specifically focus on modifiability of BRA's. To be able to explore modifiability of BRA's we conducted a mixed method approach. We constructed three architectural candidates for application in the BR's Management (BRM) domain: 1) Rule family-oriented architecture, 2) Fact-oriented architecture, and 3) Decision-oriented architecture and validated them by means of the Architecture-Level Modifiability Analysis (ALMA)-method. Analysis results show that the rule family-oriented architecture scores best on modifiability, followed by the fact oriented architecture, and lastly the decision-oriented architecture.

Keywords: Business Rules, Business Rule Architecture (BRA), Modifiability

Introduction

Laws, regulation, protocols, standards, are each example of rules that organizations are forced to act in accordance with (Bajec & Krisper, 2005; Shao & Pound, 1999; Tarantino, 2008). Each of the previous mentioned form of rules is applied to guide/constrain entities, such as individuals, teams and organizations to act in accordance with internal or external provided criteria. Take, for example, a general practice. From a regulatory and legislative point of view, rules are used to restrict access to patient information, force general practitioners to be more transparent in their decision- making and constrain the incentive system general practices can apply (Blomgren & Sundén, 2008; King & Green, 2012). In addition to externally provided criteria, organizations themselves also create additional rules, which they want teams and individuals to comply to. For example a general practitioner states rules on how a specific decision must be made. To prevent individuals and teams in an organization deviating from desired behaviour, laws regulation, protocols and standards are translated to business rules. According to Morgan (2002) a business rule is defined as: "*a statement that defines or constrains some aspect of the business intending to assert business structure or to control the behaviour of the business.*" In addition to faster changing and increased amounts of laws, regulation, protocols and standards implemented, trends like higher demanding customers, and faster changing customer's demands give rise to an increase in the amount of BR's as well as an increased pace of modifications to these business rules. Thereby increasing the need to decompose and structure BR's to accommodate for expected or unexpected modifications and making it possible to rapidly modify them when necessary. As the amount of BR's utilized in products and/or services increases, the need for efficient maintenance increases as well. It is common for practitioners to scope the context of the product and/or service which is being designed before the BR's themselves are added as part of the business logic (Zoet et al., 2014). Currently these scopes or context, also referred to as business rule architectures, are often built while not adhering to well-known and proven design principles which should be taken into account to ensure, amongst other quality attributes, modifiability to support future evolution of the product and/or service (Zoet et al., 2015).

This study features the evaluation of modifiability of three business rule architectures. As no uniform definition of a business rule architecture exists in the current body of knowledge we propose the following definition: "*A Business Rule Architecture is a formal description of a cohesive set of business logic and related relationships with the goal to provide knowledge about the structure, dependencies, and design principles."* All three architectural candidates are adopted and adapted for applicability in the BRM domain by utilizing the design science approach, featuring eight validation cycles with BRM-experts. This study comprises the inclusion of the rule family-oriented architecture, fact-oriented architecture, and decision-oriented architecture. To ground our research we selected the NHS as our case organization. The NHS published business rules documents since 2004, which served as input on which the architectural candidates are based. To be able to evaluate the included architectural candidates regarding modifiability we address the following research question: "*How can Business Rule architectures be designed for modifiability, taking into account the concept of anticipated modifications?"*

The remainder of this paper proceeds as follows. The next section provides this research's context by describing business rules, separation of concerns, and theory on modification that can occur to business rules. The third section describes the research methods utilized for this study. This is followed by the data collection in section four. The fifth section presents the data analysis and results. Lastly, section six comprises a summarization of the study's core findings, contributions as well as its limitations.

Literature

The primary goals of software engineering are to improve software quality, reduce software production costs, and facilitate maintenance and evolution. To achieve these goals, organizations constantly seek for development technologies and methodologies that add value in terms of complexity reduction, increase comprehensibility, promote reuse, and facilitate evolution of software systems (Tarr et al., 1999). As these goals are all contributing to the overall perceived quality of software systems, mechanisms utilized for these goals are often conflicting of nature. This creates problems that complicate software engineering further. In their work, Ossher & Tarr (2001) indicate that these problems are related to separation of concerns, as coined by Dijkstra (1974). The ability to achieve the goals that represent different concerns depends on the ability to keep and manage separate all concerns of importance of software engineering. Based on the further advance of separation of concerns in software engineering van der Aalst presented a literature overview, as further visualized by Zoet (2014), depicted in Figure 1. This formed ground for further separation of a new concern: 'workflows' also referred to as the 'flow' within applications.

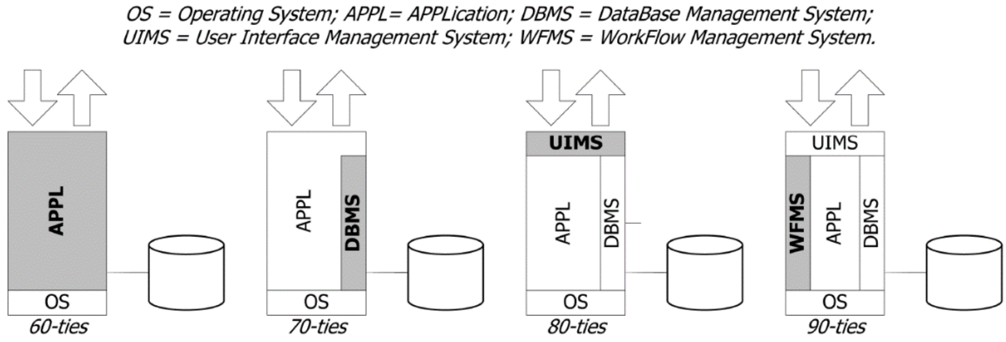


Figure 1: Evolution of the separation of concerns (Zoet, 2014)

According to Zoet (2014), separating BR's is the next step in the evolution of separating concerns from application development, which is in line with earlier work of Boyer & Mili (2011) and Graham (2007). The independent treatment of BR's by organizations also implies a different approach than process and data management (J. Hohwiller, Schlegel, Grieser, & Hoekstra, 2011). As much research effort is currently spend on implementation-dependent development of BR's we specifically analyse the design process of implementation-independent business logic.

Our goal is to analyse modifiability of different architectural candidates utilizing an Architecture Evaluation Method (AEM). To be able to do so we need to 1) select multiple architectural candidates, 2) select case material to ground our architectural candidates and, 3) select an appropriate AEM. As the current body of knowledge lacks any architectural standard for the BRM domain which takes into account design principles regarding modifiability, we reviewed literature of neighbouring fields like enterprise architectures, software architectures, and data(warehouse) architectures. According to Van Thienen (1993) and Zoet (2011) the theoretical underpinnings of the database and datawarehouse methods are most similar to that of the rule-oriented approach. Literature regarding the datawarehouse domain reveals many standards, but three and widely-used schools exist: 1) the utilization of a relational model adhering to the normalization principles proposed by Inmon (2005), 2) the utilization of star-and snowflake-schemes, focusing on dimensions of data proposed by Kimball (2002), and 3) the utilization of the relational data vault principle, based on hubs, satellites, and links proposed by Linstedt (2009).

The data warehouse architecture as defined by Inmon (2005) is similar to the architectural structure utilized by The Decision Model (TDM), as both architectures are based on relational theory (Zoet et al., 2011). Therefore we selected TDM, which is a rule family-oriented architecture as an architectural candidate in this research. Furthermore, the work of Kimball (2002) is identical to the theoretical underpinnings of the Cognition enhanced Natural language Information Analysis Method (CogNIAM) approach (Nijssen & Le Cat, 2010). Both architectures are different from the first mentioned architectural candidate as it is focused on performance and centers around facts instead of rule-families. Therefore we selected the fact-oriented architecture to be included in this research. Lastly, the Data Vaults architecture has no theoretical equivalent in the BRM domain. As Linstedt (2009) argues that the Data Vault architecture is a hybrid of the latter two architectures we selected the widely utilized concepts used in business logic described by Ross (2003) and Zoet (2014) for the construction of the decision-oriented architecture based on the theoretical underpinnings as presented in the work of Linstedt et al., (2009).

Authors agree on one phenomena constantly happening in the world of IS/IT: '*change*'. Change as part of evolution is necessary and inevitable, as products and/or services are influenced by 1) the ever-changing operating environment, 2) changes in implementation technology, and 3) stakeholder needs which are either functional or quality requirements (Rowe et al., 1998). Business logic, amongst the other concerns, is characterized by the highest change frequency (Chapin et al., 2001). This demands an approach that takes into account quality attributes like evolvability, and as an important sub-aspect of that, modifiability of business logic. According to Bass, Klein, and Bachman (2000) Modifiability is defined as: "*the ability of a system to be changed after it has been deployed*". To be able to control evolution, thus modifiability of software systems, Mannaert and Verelst (2009) propose the concept of *stability*. Stability with regards to the evolution of software systems can be achieved by utilizing a set of anticipated changes. Based on this, a recent study on a classification of modifications applicable on business logic was conducted to make a first step to make explicit a set of modifications which can be utilized to aid the evaluation of business logic (Zoet et al., 2015). As these modifications are known to stakeholders, stability can be achieved by standardizing activities to process these modifications more easily. Additionally, anticipated modifications provide room for more detailed impact assessment regarding existing products and/or services which embed business logic. Furthermore, it is important for organizations to have insights regarding the modifiability of their business logic. Therefore it is important to evaluate and quantify to what extend a given architectural candidate is. The current body of knowledge features several methods to evaluate modifiability of software architectures (Babar & Gorton, 2004), also referred to as an AEM.

Multiple AEM's are available to evaluate architectural candidates (Mattsson et al., 2006). To ground the analysis of modifiability we select an appropriate AEM of the current body of knowledge. In literature we identified six appropriate AEM's which either focus on modifiability or a trade-off analysis between multiple quality attributes of architectural candidates: EBAE (Lindvall et al., 2003), SAAM (Kazman, Bass, Abowd, et al., 1994), ATAM (Kazman et al., 2005), ALMA (Bengtsson et al., 2004), QUASAR (Bosch, 2000), and ABAS (Klein et al., 1999). From these results we needed to select the most appropriate AEM. To be able to do so we adhered to three criteria, 1) the AEM has a focus on modifiability, 2) the AEM has at least 100 citations to ensure scientific rigor, and 3) the AEM is utilized in follow-up research studies which evaluated its usefulness in practice. The second round of selection resulted in the exclusion of five AEM's, whereas only ALMA remained that complied with all three criteria.

ALMA comprises five main steps, 1) *Goal formulation*, 2) *Creation of architecture descriptions*, 3) *Elicitation of scenarios*, 4) *Evaluation of scenarios*, and 5) *Interpretation of the results* (Bengtsson et al., 2004). First, the goal of the analysis is to compare three architectural candidates on modifiability utilizing effort prediction. Furthermore, the architectural candidates are created based on the case at hand, while the same holds for the elicitation of scenarios. In their work, Bengtsson et al. (2004) propose an equation to calculate the effort required to process the included scenarios, depicted in Equation 1. The equation includes a summation of the products per type of change in the numerator, which is divided by the total amount of scenarios included (maintenance profile), as represented by C(MP). The products are determined by multiplying the size of the scenario, as represented by *S*, by the weight of the scenario, as represented by j. Furthermore, the product is multiplied by its corresponding productivity level, as represented by P_N. Lastly, the result of dividing the numerator by the denominator is multiplied with the total amount of modifications expected.

$$E_M = \frac{\sum_{AI}\left(\left(\sum_{CC_i} s_j\right)\cdot P_{cc} + \left(\sum_{NP_i} s_j\right)\cdot P_p + \left(\sum_{NC_i} s_j\right)\cdot P_{nc}\right)}{C(MP)} \cdot CT$$

Equation 1: Total effort determination

The formula as proposed in the work of Bengtsson et al (2004) is adopted and slightly adapted to accommodate the eleven modification types as found to be applicable for BRM in earlier research (Zoet et al., 2015). The results of the conversion of the equation are presented in Equation 2.

$$E_M = \frac{\sum_{AI}\left(\left(\sum_{CD_i} s_j\right)\cdot P_{cd} + \left(\sum_{DD_i} s_j\right)\cdot P_{dd} + \left(\sum_{UD_i} s_j\right)\cdot P_{ud} + \left(\sum_{CBR_i} s_j\right)\cdot P_{cbr} + \left(\sum_{DBR_i} s_j\right)\cdot P_{dbr} + \left(\sum_{CC_i} s_j\right)\cdot P_{cc} + \left(\sum_{DC_i} s_j\right)\cdot P_{dc} + \left(\sum_{UC_i} s_j\right)\cdot P_{uc} + \left(\sum_{CFV_i} s_j\right)\cdot P_{cfv} + \left(\sum_{UFV_i} s_j\right)\cdot P_{ufv} + \left(\sum_{DFV_i} s_j\right)\cdot P_{dfv}\right)}{C(MP)} \cdot CT$$

Equation 2: Total effort determination as adapted for this study

For example, in Equation the modification types included are Change Code (Pcc), New Parameter (Pp), and New Code (Pnc). We adapted the formula in such a way that the modification types concerning business logic are included, for example Create Decision (Pcd). This process is depicted in Figure 2.

A: $$\left(\sum_{CC_i} s_j\right) \cdot P_{cc} +$$ → B: $$\left(\sum_{CD_i} s_j\right) \cdot P_{cd} +$$

Figure 2: Adaptation of (a part of) the equation of Berngtsson et al (2004)

Research method

The goal of this research is to evaluate the rule family-oriented, fact-oriented, and decision-oriented architectures regarding modifiability by utilizing effort prediction. In addition to the goal of the research, also the maturity of the research field is a factor to determine the appropriate research methods and techniques. The maturity of the BRM research field, with regard to non-technological research, is nascent (Kovacic, 2004; Nelson et al., 2010; Zoet, 2014). Focus of research in nascent research fields should lie on "identifying new constructs and establishing relationships between identified constructs" (Edmondson & Mcmanus, 2007). Summarized, to accomplish our research goal, a research approach is needed in which 1) different architectural candidates have to be adopted and/or created, 2) an appropriate method to analyse modifiability of architectural candidates is selected, and 3) the analysis is performed and its results interpreted and reported in order to contribute to the incomplete body of knowledge. The analysis of modifiability is conducted utilizing the ALMA method in a quantitative manner. Furthermore, to ground the construction of the architectural candidates we utilized the design science framework of Hevner et al. (2004) which comprised of eight validation cycles. Lastly, we conducted two semi-structured interviews to reveal the productivity metrics of the NHS.

Quantitative evaluation of modifiability

For the quantitative analysis of modifiability we selected and utilized ALMA (Bengtsson et al., 2004) as the most appropriate method to ground our analysis of business rule architectures concerning modifiability. To be able to do so several input variables need to be collected. First, the scenario size will be determined utilizing function points as the unit of analysis. Second, the weight of the modification scenario will be determined utilizing a normalized weight for each scenario based on historic data as reported upon in the work of Zoet, Smit & Leewis (2015). Furthermore, the third variable comprises the amount of modifications estimated for a given period. Again, based on the work of Zoet, Smit & Leewis (2015), the total amount of modifications are included to estimate the amount of modifications for the following eight years. Lastly, the productivity scores concerning the scenarios included determine how much time is needed to process the modifications as part of the scenarios. Literature concerning productivity of processing modifications offer some examples which we could utilize as input for the analysis of business rule architectures (Henry & Cain, 1997; Maxwell, Van Wassenhove, & Dutta, 1996). However, it is proposed that productivity should be measured at the case organization due to the many context-specific influences that should be taken into account (Bengtsson et al., 2004).

Semi-structured interviews

As described in section 3.1 it is desirable that the productivity metrics included to evaluate architectural candidates on modifiability are context-specific. To be able to derive these metrics we created an interview protocol. To ground our interviews we based our research protocol on earlier research on a classification of modification types regarding the same case organization (Zoet et al., 2015). The interview protocol consists of a set of fifteen questions from which the main part, questions 3 – 13, focus on quantifying how much time is needed to perform a single modification of a modification type, for example: Create Decision and Delete Condition. For a detailed description of the modifications see section two. Furthermore, a case specific example was created to guide the interviewees with identifying the elements on which modifications could occur.

Data Collection

To ground our choice for a case organization we selected a case organization based on theoretical and pragmatic criteria. Our theoretical criterion was: "the case site should deal with business rules, regulation, laws or policies that change frequently." Our pragmatic criterion was: "the case site should have kept documentation of the business rules, regulation, laws or policies." Based on these criteria the British National Health Service (NHS) was selected. The NHS is built up from four different health care systems, England, Northern Ireland, Scotland, and Wales. These regions combined provide healthcare services for over 64.1 million UK residents. The NHS employs more than 1.6 million people, which makes it one of the top five workforces in the world in terms of scale. Over one million patients every 36 hours make use of NHS services. A significant part of healthcare management in the UK by the NHS focuses on the management of chronic diseases. In April 2004 the NHS introduced the Quality and Outcomes Framework (QOF) as part of the new General Medical Services (GMS) contract. The QOF is a Pay-for-Performance-scheme covering a range of clinical, organizational, and patient areas in primary care. It is established to reward practices for the provision of high quality care and helps fund further improvements in the delivery of clinical care. The QOF includes the measurement of different domains, however, due to the scope of this study only the clinical and public health domains are considered. The NHS manages the QOF which is a Pay-for-Performance-scheme in that comprises to 25 clinical conditions. For each individual condition they create BR's to select when a clinic must be paid for the treatment of the patient (Gillam & Siriwardena, 2011).

The business rule sets are updated twice a year to accommodate the introduction of new insights revealed by empirical research and/or changes in law and regulations. At the time of writing, the combination of these domains contain 25 clinical conditions, with a large amount of underlying indicators, which make up for 80 percent of the commonly encountered health issues in primary care (Lester & Campbell, 2010). Examples of clinical conditions as part of the QOF are: Heart Failure (HF), Diabetes Mellitus (DM), and Chronic Obstructive Pulmonary Disease (COPD). Furthermore, other documents, for example the guidance documents concerning the QOF specifically designed for NHS employees were extracted to support creation of the architectural candidates.

Lastly, the interviews will be conducted with two subject-matter experts working at the Health and Social Case Information Centre (HSCIC, as a department of the NHS) which are responsible for the translation of requirements, resulting from new laws and regulations, policies, changing healthcare need, and/or research outcomes, into business logic. Due to practical reasons, the interviews are conducted utilizing telephone interviews, and are recorded by an audio recorder via a secondary device.

Data Analysis

Data analysis consisted of five main activities, 1) determination of the modification scenario's, 2) analysis of the size of each included modification scenario, 3) analysis of the standardized weights of each modification scenario, 4) analysis of the collected data concerning the interviews to derive productivity levels, and 5) the analysis of the modification occurrences to be able to calculate the total amount of modifications. First, we determined the structure for each scenario. Instead of including multiple modification types in one scenario we adhered to the eleven modification types as presented in the work of Zoet, Smit & Leewis (2015). For the COPD case we included five modifications per modification type, whereas for the DM case we included twenty modifications per modification type, as this case is four times as large in size compared to the COPD case. The size for each modification scenario is manually analysed utilizing function points as the unit of analysis, based on the work of Felfernig & Salbrechter (2004). To be able to do so we identified each element in the architectural candidates as one logical function.

For each logical function the size in function points is calculated by multiplying the amount of logical functions affected by the modification in the scenario, by the amount of function points per modification type. The amount of function points per modification type is calculated by dividing the total amount of minutes needed to process one modification by the total amount of minutes per function point, which we derived from earlier research on project delivery rates in software engineering, averagely 755 minutes per function point (Bundschuh & Dekkers, 2008; Shepperd et al., 2006). The standardized weights are analysed by utilizing the results on modifications per modification types as presented in the work of Zoet, Smit, and Leewis (2015). The calculation of the normalized weights is performed by dividing the amount of modifications observed of a given modification type by the total amount of modifications observed.

The interview data was analysed using arithmetic calculations. For this initial study regarding modifiability we think it is a sufficient means to express productivity levels. The productivity levels are expressed as an average of the amount of minutes it takes to process a single modification type, taken from the averages as derived from both interviews.

Results

As described earlier we conducted two interviews with subject-matter experts as well as the analysis of modifiability of three architectural candidates utilizing an adapted variant of ALMA which is applicable for business rule architectures. First, we present the interview results regarding productivity levels in Figure 3. In this figure the average time in minutes per modification is presented. From this we derived the number of modifications per month assuming a theoretical maximum productivity as elaborated upon in chapter five. Furthermore, the best and worst case productivity averages per modification type are presented accompanied with their corresponding maximum capacity per month.

Modification type	Average time per M	M's/Month	Worst case average time per M	M's/Month	Best case average time per M	M's/Month
CD	4.25	2484	7.5	1408	1	10560
DD	16.25	649	22.5	469	10	1056
UD	8.5	1242	15	704	2	5280
CBR	1200	8.8	1200	9	1200	9
DBR	25.5	414	60	176	15	704
CC	109	96	210	50	7.5	1408
DC	150	70	150	70	15	704
UC	97.5	108	90	117	15	704
CFV	6.25	1689	7.5	1408	5	2112
UFV	6.25	1689	7.5	1408	5	2112
DFV	6.25	1689	7.5	1408	5	2112

Figure 3 - Interview results regarding productivity of modifications (in minutes)

Based on our interview results we were able to predict the effort required to process the included scenarios, for each architectural candidate. Results shows that, concerning the COPD case, the RFO architecture scores best on effort prediction. Furthermore, the FO architecture shows a 2.8% of additional effort required, while the DO architecture shows an additional 16% of additional effort required to process the set of scenarios provided, see Figure 4. Additionally, results shows that, concerning the DM case, the RFO architecture scores best on effort prediction. Furthermore, the FO architecture shows a 49.14% of additional effort required, while the DO architecture shows an additional 47.81% of additional effort required to process the set of scenarios provided.

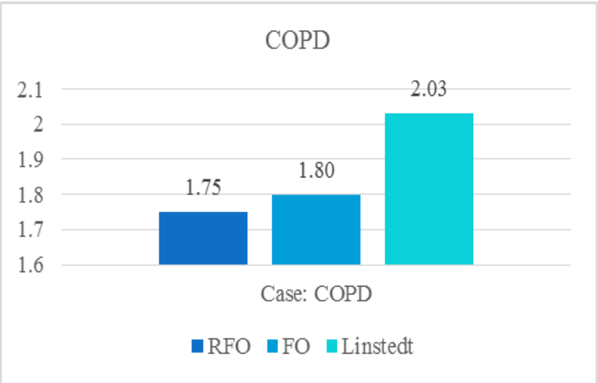


Figure 4 - Effort prediction in hours (COPD)

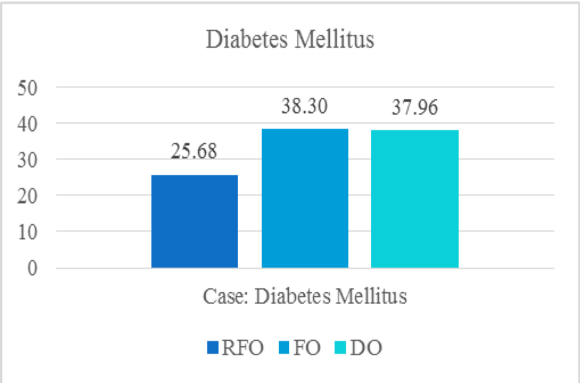


Figure 5 - Effort prediction in hours (DM)

Additionally, we utilized the available data to discover the theoretical best and worst case outcomes regarding total effort. To be able to do so we stopped utilizing the averages from both interviewees concerning the effort required in minutes per modification type. Both interviewees differ in experience level, where interviewee one has four years of experience working with the QOF, while interviewee two has one year of experience working with the QOF. We report on the less experienced employee as the theoretical worst case effort prediction, while the more experienced employee is reported as theoretical best case effort prediction. The results for the COPD case are presented in Figure 6.

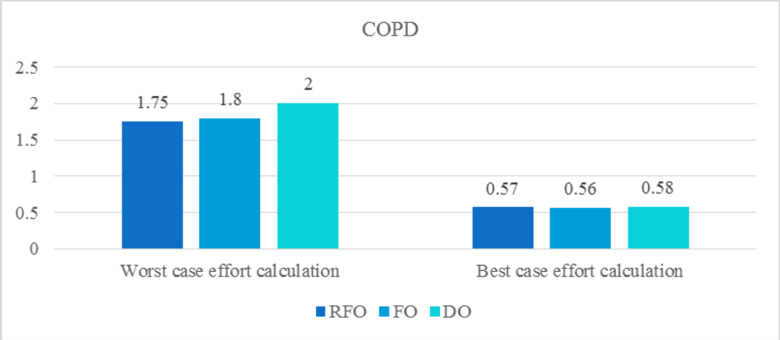


Figure 6 - Worst versus best case effort prediction in hours (COPD)

The results presented in Figure 6 shows that a worst case approach is similar to the average effort calculation as presented earlier. The best case approach shows some difference, resulting just more than half an hour of effort predicted for each architectural candidate.

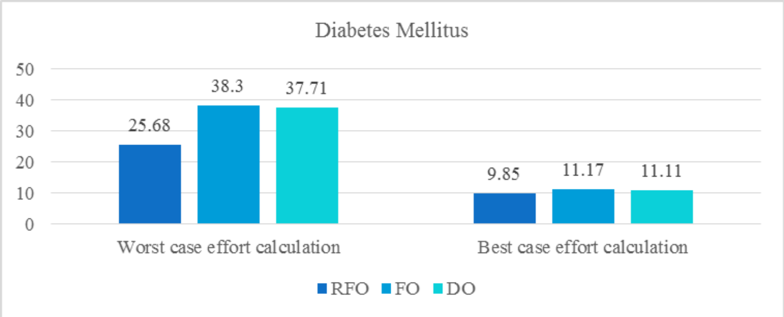


Figure 7 - Worst versus best case effort prediction in hours (DM)

The results regarding the second case, Diabetes Mellitus, are presented in Figure 7. Again, the RFO architecture scores best on both worst and best case effort prediction. The worst case architecture shows similar percentage differences compared to the average effort calculation and is therefore not further elaborated upon. The best case effort prediction also resulted in the best performance by the RFO architecture, while the FO architecture shows a 13.38% of additional effort required and the DO architecture shows an additional 12.79% of additional effort required. Summarized, Table 1 presents the numeric results of the analysis outcomes with both the predicted effort as well as its corresponding percentage differences.

Table 1 - Summary of results of the effort prediction utilizing ALMA

COPD	**RFO**	**FO**	**DO**
Effort predicted (H)	1.75	1.80	2.03
Difference in effort (H)	N.A.	0.05	0.28
Difference in %	N.A.	2.8%	16%
DM	**RFO**	**FO**	**DO**
Effort predicted (H)	25.68	38.30	37.96
Difference in effort (H)	N.A.	12.62	12.28
Difference in %	N.A.	49.14%	47.81%
COPD – Worst case	**RFO**	**FO**	**DO**
Effort predicted (H)	1.75	1.80	2
Difference in effort (H)	N.A.	0.05	0.25
Difference in %	N.A.	2.85%	14.28%

COPD – Best case	**RFO**	**FO**	**DO**
Effort predicted (H)	0.57	0.56	0.58
Difference in effort (H)	N.A.	- 0.01	0.01
Difference in %	N.A.	- 1.78%	1.75%
DM – Worst case	**RFO**	**FO**	**DO**
Effort predicted (H)	25.68	38.30	37.71
Difference in effort (H)	N.A.	12.62	12.03
Difference in %	N.A.	49.14%	46.84%
DM – Best case	**RFO**	**FO**	**DO**
Effort predicted (H)	9.85	11.17	11.11
Difference in effort (H)	N.A.	1.32	1.26
Difference in %	N.A.	13.38%	12.79%

Analysis revealed that an average difference of 9.4% in effort is predicted between all three included architectural candidates for the smaller COPD case. When analysing the larger Diabetes Mellitus case, our findings suggest lower effort prediction for the rule family-oriented architecture compared to the other two included architectural candidates. Further analysis of the differences between the fact-oriented and decision-oriented architectures reveals a difference in impact between both cases. Results show that the impact of the fact-oriented architecture increases as the case size increases. However, the opposite holds for the decision-oriented architecture, which suggest that whenever the case size increases, less impact is measured. The difference between both architectural candidates is caused by redundancy of data which increases when the case grows, caused by elements that are reused for other decisions in the same case. This implies that the modifiability of the fact-oriented architecture decreases when the case size increases. This is caused by the star-schemed structures that do not relate to each other with relationships. This results in an increase in creation of redundant conditions and underlying fact values as the case size increases. On the other hand, the opposite for the decision-oriented architecture is true due to the utilization of links between decisions which creates the opportunity to reuse, for example, conditions and underlying fact values for other decisions. This results in less impact, thus higher modifiability of the architecture. A similar structure is adhered to by the rule family-oriented architecture, which relates architectural elements without creating redundant conditions and conclusions. Furthermore, the results suggest that the difference between the rule family-oriented architecture and the other two architectural candidates is caused by further separation of decision logic by both the fact-oriented and decision-oriented architectures, while this is not adhered to by the rule family-oriented architecture. This means less impact is calculated due to the impact on less individual logical functions in the architectural candidate. Lastly, results of the worst and best case perspectives reveal a similar outcome to that of the perspective which takes into account the average productivity scores.

Conclusions & discussion

BR's are widely applied, standalone and embedded in smart objects. Therefore they have become a separate concern in information system design. As a result they also have to be managed separately. The purpose of this research is to explore the concept of business rule architectures and their aspect of modifiability as this is an important aspect in the evolution of products and/or services that utilize large amounts of business logic. Furthermore, the purpose was to evaluate a selection of architectural candidates to evaluate what makes these architectures modifiable or not. To be able to this we addressed the following research question: "*How can Business Rule architectures be designed for modifiability, taking into account the concept of anticipated modifications?*" In order to answer this question, we conducted interviews with subject-matter experts and utilized the design science approach to build our artefacts needed to evaluate architectural candidates on modifiability, based on case data containing large amounts of business logic published by the NHS. The analysis of modifiability concerning the three included architectural candidates focused on the estimation of the average effort, the worst case effort, and the best case effort to process the predefined set of scenarios.

Based on the analysis conducted we can conclude that the rule family-oriented architecture scores best concerning predicted effort, also revealing that this architectural structure is best modifiable of all three included architectural candidates. Furthermore, our findings suggest that the modifiability of smaller cases benefits from the utilization of the fact-oriented architecture as it is less sensitive to impact when a low amount of relationships are needed.

However, a larger case with more business logic benefits from the decision-oriented architecture as it re-uses existing elements within the context of business logic it is utilized. Lastly, the analysis of worst versus best case effort prediction further validated our findings as it presented similar results in the comparison of all three included architectural candidates.

Several limitations may affect our results. The first limitation is related to the amount of interviewees included in this research. Whether this amount is too low is debatable, but looking at the case organization we could not include more interviewees as this comprises the whole population within the NHS. Moreover, most limitations are related to sample size, which is applicable to multiple key points in this research. First, the amount and differentiation of sizes could be improved by including more clinical conditions as CHD, which is very large, or Asthma, which is small. Furthermore, other architectural candidates, featuring other architectural structures could be added to improve generalizability as well. Likewise, this study only featured an adapted version of ALMA as a method of analysis. Generalizability could be improved when more methods of analysis are utilized for the analysis of modifiability of business rule architectures. Therefore, future research could focus on the inclusion of more cases and architectural candidates while also utilizing and comparing the results of different methods of analysis to reveal if a method of analysis could be tailored for utilization in the BRM domain.

From a theoretical perspective, our study provides new knowledge on the concept of a business rule architecture, as this is clearly a knowledge gap in the current body of research. From a practical perspective, our study provides input relevant for the design of products and/or services where business logic is utilized to support decision making. Business logic, amongst other concerns in software engineering, is characterized by the highest change frequency, implying that modifiability is an important aspect to take into account.

References

Babar, M. a., & Gorton, I. (2004). Comparison of scenario-based software architecture evaluation methods. In Proceedings of the 11th Asia-Pacific Software Engineering Conference (pp. 600–607). IEEE. http://doi.org/10.1109/APSEC.2004.38

Bajec, M., & Krisper, M. (2005). A methodology and tool support for managing business rules in organisations. Information Systems, 30(6), 423–443. http://doi.org/10.1016/j.is.2004.05.003

Bass, L., Klein, M., & Bachmann, F. (2000). Quality attribute design primitives - (No. CMU/SEI-2000-TN-017). Pittsburgh, Pennsylvania: Carnegie-Mellon University Pittsburg.

Bengtsson, P., Lassing, N., Bosch, J., & Van Vliet, H. (2004). Architecture-level modifiability analysis (ALMA). Journal of Systems and Software, 69(1-2), 129–147. http://doi.org/10.1016/S0164-1212(03)00080-3

Blomgren, M., & Sundén, E. (2008). Constructing a European healthcare market: The private healthcare company Capio and the strategic aspect of the drive for transparency. Social Science & Medicine, 67(10), 1512 – 1520.

Bosch, J. (2000). Design and use of software architectures: adopting and evolving a product-line approach. Pearson Education.

Boyer, J., & Mili, H. (2011). Agile business rule development. Agile Business Rule Development, 49–71. http://doi.org/10.1007/978-3-642-19041-4

Bundschuh, M., & Dekkers, C. (2008). The IT measurement compendium: estimating and benchmarking success with functional size measurement. Springer Science & Business Media.

Chapin, N., Hale, J. E., Khan, K. M., Ramil, J. F., & Tan, W. G. (2001). Types of software evolution and software maintenance. Journal of Software Maintenance and Evolution: Research and Practice, 13(1), 3 – 30.

Dijkstra, E. W. (1974). On the role of scientific thought. In Selected writings on computing: a personal perspective (pp. 60 – 66). Springer New York.

Edmondson, A. C., & Mcmanus, S. E. (2007). Methodological Fit in Management Field Research. Proceedings of the Academy of Management, 32(4), 1155–1179.

Felfernig, A., & Salbrechter, A. (2004). Applying function point analysis to effort estimation in configurator development. In Proceedings of the International Conference on Economic, Technical and organisational aspects of Product Configuration Systems (pp. 109 – 119).

Gillam, S., & Siriwardena, A. N. (2011). The Quality and Outcomes Framework: QOF-transforming general practice. Oxon: Radcliffe Publishing.

Graham, I. (2007). Business rules management and service oriented architecture: a pattern language. John wiley & sons.

Henry, J. E. E., & Cain, J. P. P. (1997). A quantitative comparison of perfective and corrective software maintenance. Journal of Software Maintenance-Research and Practice, 9(5), 281–297. http://doi.org/10.1002/(sici)1096-908x(199709/10)9:5<281::aid-smr154>3.3.co;2-g

Hohwiller, J., Schlegel, D., Grieser, G., & Hoekstra, Y. (2011). Integration of BPM and BRM. In Business Process Model and Notation (pp. 136 – 141). Springer Berlin Heidelberg.

Inmon, W. H. (2005). Building the Data Warehouse. John wiley & sons.

Kazman, R., Bass, L., Abowd, G., & Webb, M. (1994). SAAM: a method for analyzing the properties of software architectures. Proceedings of 16th International Conference on Software Engineering. http://doi.org/10.1109/ICSE.1994.296768

Kazman, R., Bass, L., Klein, M., Lattanze, T., & Northrop, L. (2005). A basis for analyzing software architecture analysis methods. Software Quality Journal, 13(4), 329 – 355.

Kimball, R., & Ross, M. (2002). The data warehouse toolkit. (R. Elliott, Ed.) (2nd edition). New York: John wiley & sons.

King, R., & Green, P. (2012). Governance of primary healthcare practices: Australian insights. Business Horizons, 55(6), 593 – 608.

Klein, M. H., Kazman, R., Bass, L., Carriere, J., Barbacci, M., & Lipson, H. (1999). Attribute-Based Architecture Styles. Software Engineering Institute, Carnegie Mellon University, 225 – 243. http://doi.org/CMU/SEI-99-TR-022

Kovacic, A. (2004). Business renovation: business rules (still) the missing link. Business Process Management Journal, 10(2), 158 – 170.

Lester, H., & Campbell, S. (2010). Developing Quality and Outcomes Framework (QOF) indicators and the concept of'QOFability. Quality in Primary Care, 18(2), 103 – 109.

Lindvall, M., Tvedt, R. T., & Costa, P. (2003). An empirically-based process for software architecture evaluation. Empirical Software Engineering, 8(1), 83 – 108.

Linstedt, D., Graziano, K., & Hultgren, H. (2009). The New Business Supermodel, The Business of Data Vault modeling (2nd editio). Lulu.com.

Mannaert, H., & Verelst, J. (2009). Normalized systems: re-creating information technology based on laws for software evolvability. Koppa.

Mattsson, M., Grahn, H., & Mårtensson, F. (2006). Software Architecture Evaluation Methods for Performance, Maintainability, Testability, and Portability. In Proceedings of the 2nd International Conference on the Quality of Software Architectures. Retrieved from http://www.bth.se/fou/forskinfo.nsf/0/412d328ffb033edec125730e004b649d?OpenDocument&Click=\nhttp://www.researchgate.net/publication/200622064_Software_Architecture_Evaluation_Methods_for_Performance_Maintainability_Testability_and_Portability/file/72e7e52

Maxwell, K. D., Van Wassenhove, L., & Dutta, S. (1996). Software development productivity of European space, military, and industrial applications. IEEE Transactions on Software Engineering, 22(10), 706 – 718.

Morgan, T. (2002). Business rules and information systems: aligning IT with business goals. Addison-Wesley Professional.

Nelson, M. L., Peterson, J., Rariden, R. L., & Sen, R. (2010). Transitioning to a business rule management service model: Case studies from the property and casualty insurance industry. Information & Management, 47(1), 30–41. http://doi.org/10.1016/j.im.2009.09.007

Nijssen, G. M., & Le Cat, A. (2010). Kennis Gebaseerd Werken. Amsterdam: PNA Publishing B.V.

Ossher, H., & Tarr, P. (2001). Using multidimensional separation of concerns to (re) shape evolving software. Communications of the ACM, 44(10), 43 – 50.

Ross, R. G. (2003). Principles of the business rule approach. Addison-Wesley Longman Publishing Co.

Rowe, D., Leaney, J., & Lowe, D. (1998). Defining systems architecture evolvability - a taxonomy of change. International Conference and Workshop: Engineering of Computer-Based Systems, 45–52. http://doi.org/10.1109/ECBS.1998.10027

Shao, J., & Pound, C. J. (1999). Extracting business rules from information systems. BT Technology Journal, 179 – 186.

Shepperd, M., Mair, C., & Forselius, P. (2006). An Empirical Analysis of Software Productivity. In Proceedings of the 3rd Software Measurement European Forum (SMEF'06). Rome: Southampton Solent University.

Tarantino, A. (2008). Governance, Risk, and Compliance Handbook: Technology, Finance, Environmental, and International Guidance and Best Practices. John Wiley & Sons.

Tarr, P., Ossher, H., Harrison, W., & Sutton Jr, S. M. (1999). N degrees of separation: multi-dimensional separation of concerns. In Proceedings of the 21st international conference on Software engineering (Vol. 21, pp. 107 – 119). ACM.

Vanthienen, J., & Snoeck, M. (1993). Knowledge factoring using normalisation theory. International Symposium on the Management of Industrial and Corporate Knowledge (ISMICK'93), 27 – 28.

Zoet, M. (2014). Methods and Concepts for Business Rules Management (1st ed.). Utrecht: Hogeschool Utrecht.

Zoet, M., de Haan, E., & Smit, K. (2014). Van Wetsanalyse tot Producten en Diensten voor Burgers en Bedrijven. Utrecht.

Zoet, M., Ravesteyn, P., & Versendaal, J. (2011). A structured analysis of business rules representation languages : Defining a normalisation form. Proceedings of the 22nd Australian Conference on Information Systems, Paper 20.

Zoet, M., Smit, K., & Leewis, S. (2015). A Classification of Modification Categories for Business Rules. In Proceedings of the 28th Bled eConference.

Appendix L – ALMA PDD

The PDD in this appendix represents ALMA in both the application for the SA as the BRM domain.