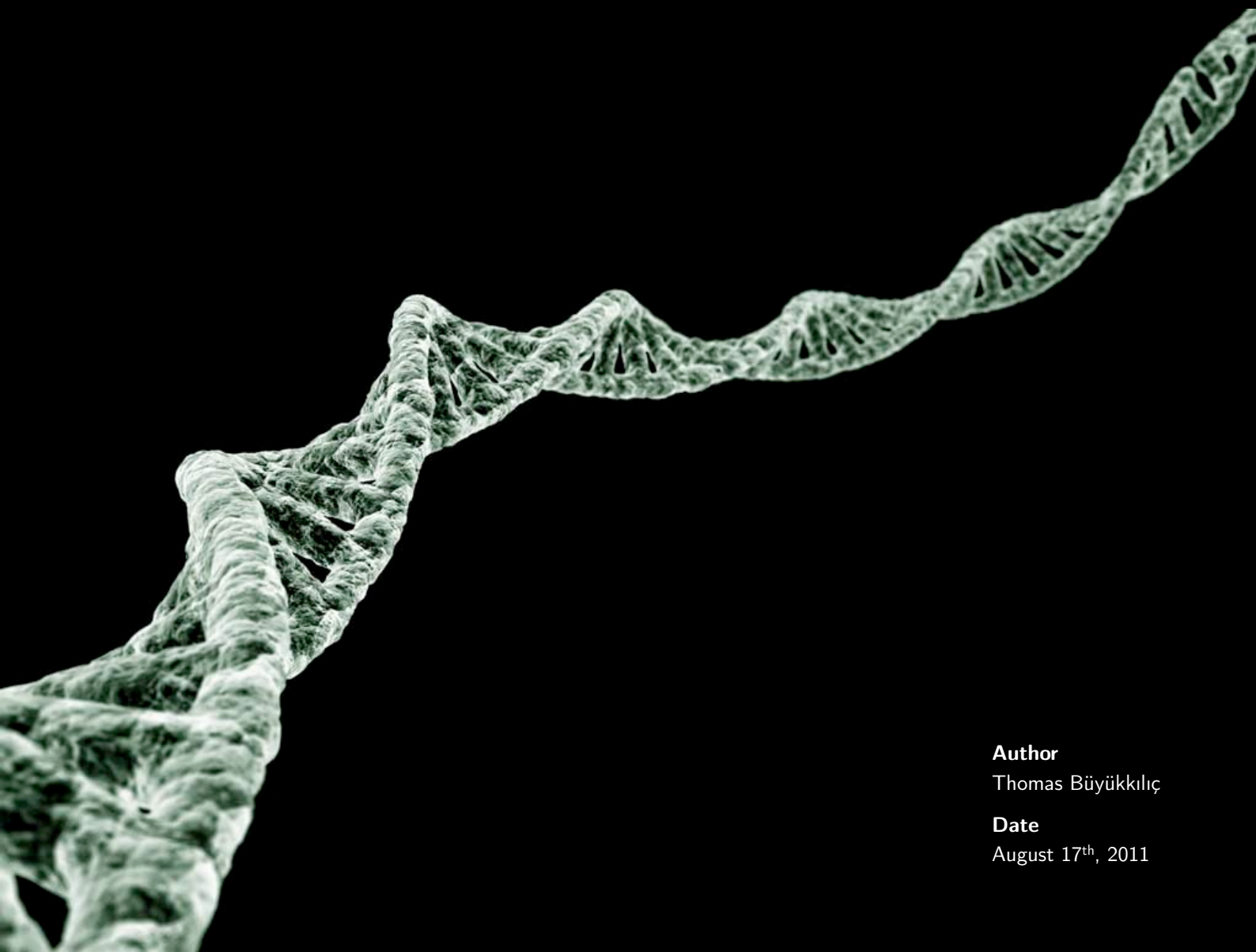
UNIVERSITY OF TWENTE.

Rule-based Semantic IS Standards

A conceptual framework for rule-based semantic IS standards development



Author
Thomas Büyükkılıç

Date
August 17th, 2011

UNIVERSITY OF TWENTE.

Rule-based Semantic Standards

A conceptual framework for rule-based semantic IS standards development

Author

Thomas Büyükkılıç

E-mail: thomas.buyukkilic@gmail.com

Student number: s0011061

Organization

University of Twente

School of Management and Governance

MSc Business Information Technology

Supervisors

ir. E.J.A. Folmer

dr. ing. A. Wombacher

Date

August 17th, 2011

Version

1.0 — Final

Preface

When I first started the *Business Information Technology* program, I had no idea where I was going, but the idea of being a bridge between the business world and the IT world appealed to me. In the years that followed I had setbacks and was close to giving up, but my stubbornness never let me. I have always known that I am capable enough, but I never had the motivation to put the time in it, so it became a big mess. Luckily, I asked Céline Heijnen to help me sort out the mess I had created. She was willing to help me and had put me on the right track. I thank her for that. It has been a long and stressful road from there, but the end is near: You are reading the final deliverable of my Master's in Business Information Technology.

First of all, I would like to thank my supervisors at the University of Twente for putting the time and the effort to help me get this done in time before september 1st. Erwin and Andreas, thank you for steering me in the right direction when I needed it and helping me get the insight I needed.

Also I would like to thank the following people. My wife, who is pregnant and in the hospital while I'm writing this, for being the support and motivation I needed during my studies. It has been a very tough period and I could not have done this without you, thank you! My mother, who has been my rock throughout my life, for never giving up on me, no matter what. Thank you for being you! My sister, who has always had my back, for her kind words and support, God bless you. My brother, for having faith in me and for his wise words, thank you. Finally, I would like to thank my family and my friends for their faith, support, and understanding during this project.

I hope you enjoy reading my thesis.

Enschede, August 14th 2011

Thomas Büyükkılıç

Abstract

It is estimated that only 2% of the businesses worldwide that could benefit from standards, actually adopt and use them. The adoption of a standard is influenced by its quality, which is inherent to the tactics used for developing the standard. Other aspects that influence adoption are the cost of the standard, and the understandability of the standard. In this research we are taking a non-traditional approach (i.e. tactic) to semantic IS standard development — a business rule approach. With a business rule approach we aim to increase the understandability and the quality of semantic IS standards, while reducing the (development/ maintenance) costs.

Business rules are known to function as bridges between business people and IT people. They can be defined in rule modeling languages which can be expressed in both natural language and formal language at the same time, which might increase the *understandability of the standard*. One such rule modeling language is 'RuleSpeak'. RuleSpeak is based on OMG's Semantics of Business Vocabulary and Business Rules (SBVR), which is an platform-independent model. Business rules defined in RuleSpeak can be translated to other Meta-Object Facility models such as Unified Modeling Language. This is especially important for semantic IS standards as their contents often involve models and diagrams. Managing one set of business rules from which models can be generated, will result in a more consistent standard, which is a key characteristic of the *quality of the standard*. Being able to generate models from a single source might also reduce development/maintenance duration, which directly translates to *cost reduction*.

The result of this research is a conceptual framework for and an approach to rule-based semantic IS standards development. The approach is based on defining business rules based on the functionality they have. Business rules can define *structure*, *power*, and *control*. Structure emerges by creating structural rules that define the relationship that concepts have with each other. Power (i.e. processes) is generated by creating declarative processes with pre-, boundary-, and post-conditions. Control is exerted by constraining the power (i.e. processes).

To validate the proposed approach, an experiment was conducted in which the *SETU Standard for Reporting Time & Expenses* was translated to RuleSpeak. The case study served as input for the evaluation of the hypothesis which states that 'a business rule approach to semantic IS standards development can lead to reduction of costs and increase of quality and understandability'. The evaluation was done in the form of an interview and a survey with a focus group consisting of standards developers from SETU. The focus group disagreed that the proposed approach to semantic IS standards development could lead to cost reduction or remove direct adoption barriers, but they agreed that it could lead to quality increase in the form of completeness, consistency, compliance and precision increase.

The increase in the quality characteristics can be explained from the point of view that proper application of SBVR forces the developer to think about *how* a concept is defined in the SBVR vocabulary. Since concepts build on terms, and terms can be used to define other terms, quality characteristics like completeness, consistency, compliancy and precision are enforced. The main conclusion is that applying the proposed approach can lead to quality increase of semantic IS standards, which can influence their adoption.

Contents

1. Introduction

E-business has seen significant growth over the past few years, partly because it leads to better coordination and better interoperability in the supply chain, as well as new business opportunities (Lee & Whang, 2004). The interoperability of the e-business information systems (IS) is a key characteristic to achieve common e-business benefits such as reduced costs, increased flexibility, and faster response times (Lee & Whang, 2004). There is a growing need for standardization as a means to achieve IS interoperability (Folmer & Verhoosel, 2011). In many industries, firms develop e-business standards collaboratively in a standard consortium (Zhao, Xia & Shaw, 2007), these consortia are considered *vertical* standards development organizations (SDOs), or better known as standards settings organizations (SSOs). Standards organizations can typically be categorized into two groups: SDOs and SSOs. SDOs are considered formal/traditional development organizations that develop and maintain standards on national, regional and international level (Folmer & Verhoosel, 2011; Spivak & Brenner, 2001), examples are. *ISO* and *ITU*. All the other organizations that develop standards are considered SSOs, examples are *OASIS*, *W3C*, or *SETU*.

Semantic IS standards are typically developed and maintained by SSOs and they are critical for e-business to e-business transactions because they prescribe guidelines for inter-system communication, with the goal of increasing interoperability of information systems (Folmer & Verhoosel, 2011). However, standards need sponsor support (standards organizations and technology providers) and early adopters to be created (Choia, Raghu & Vinze, 2004) and this is lacking. Semantic IS standards even need support from all stakeholders (Folmer & Verhoosel, 2011). This lack of sponsor support and adopters, translates into a 5% usage of standards by organizations that could benefit from them (Beck & Weitzel, 2005 as cited by Folmer & Verhoosel, 2011). It is estimated that 2% of the business worldwide (that could benefit from standards) actually use them (Wigand, Markus & Steinfield, 2005 as cited by Folmer & Verhoosel, 2011). There many explanations for these low values, one of them is that there are network externalities[1] in effect in the adoption of standards (Boh, Soh & Yeo, 2007; Cathomen & Klein, 1997; Katz & Shapiro, 1985; Thomas, Probets, Dawson & King, 2008; Weitzel, Beimborn & Koenig, 2006), so it necessary for standards to have a "critical mass" of supporters and adopters to be adopted. The lack of adoption can also be caused from barriers such as difficulty understanding the standard, standards revision process, and the cost of the standard (Thomas et al., 2008). It is also important to have all the stakeholders participate and to keep them involved *during* the standardization to increase adoption (Markus, Steinfeld, Wigans & Minton, 2006).

A survey on semantic standards quality involving 37 standards organizations including SSOs like *XBRL*, *HR-XML*, *SETU*, and *Aquo* showed that more than 90% of the respondents agreed that the quality of their standards can be improved (Folmer & Punter, 2010). Also, a vast majority (90%) agrees that quality increase in their standards will increase system interoperability.

The adoption of a standard is influenced by its quality, which is inherent to the tactics used for developing the standard (Markus et al., 2006). In this research we are taking a non-traditional approach (i.e. tactic) to semantic IS standard development — a business rule approach. Major motivations for adopting a business rule approach in organizations are agility and consistency (Halle & Goldberg, 2006: p.21). A business rule approach to cost- and quality-related problems within organizations has shown effective in real-life cases (Halle & Goldberg, 2006; Masud & Lucker, 2009; Myers, 2010), especially in a changing environment where business rules can offer high levels of automation as well as flexibility (Paschke, 2006). We call our approach BRASD (a Business Rule Approach to Semantic IS standards Development) and with it we aim to:

[1] From Wikipedia: *"In economics and business, a network effect (also called network externality or demand-side economies of scale) is the effect that one user of a good or service has on the value of that product to other people. When network effect is present, the value of a product or service is dependent on the number of others using it."*

1. *Increase the understandability of semantic IS standards.* Business rules are known to function as bridges between business people and IT people. They can be defined in rule modeling languages which can be expressed in both natural language and formal language at the same time, which might increase the understandability of the standard.
2. *Increase the quality of semantic IS standards.* Business rules have shown potential for data quality increase (see real-life cases in the previous paragraph). In this research we research if and how a BRASD can contribute to data quality. Data quality is strongly related to standards quality (Folmer & Verhoosel, 2011: p.95). The ISO/IEC 25012 document defines 15 data quality characteristics. We have selected 8 of these characteristics which might be related to semantics IS standards (ISO IEC, 2008): accuracy, completeness, consistency, compliancy, precision, traceability, understandability, and portability. Consistency is especially important for semantic IS standards, as they rely heavily on consistency in inter-related diagrams, models, data (e.g. XML schemas), and sometimes across multiple standards.
3. *Reducing the development/maintenance costs of semantic IS standards.* Business rules are defined centrally in a vocabulary, which can serve as the source of information from which models can be generated. In this research we describe how these models can be generated. This might result in a reduction of development/maintenance duration, which directly translates to cost reduction. In some countries such as the Netherlands, organizations are obliged to implement certain semantic IS standards, e.g. the Dutch government needs to comply with the SETU standards for the staffing industry. Shortening these development/maintenance duration can lead to cost reduction.

2. Research design

In this chapter we describe the design and approach used to perform this research. This is an IS research, which implies that it needs to be reproducible by other people as well. If an IS research is not reproducible, it has little relevance (Hevner, March & Park, 2004).

Just like every chapter after this one, a section outline will be given. First, we will state the problems we mentioned in the introduction explicitly in section 2.1. Our research objectives and expected contributions are explained in section 2.2. The research scope is determined in section 2.3. The research model (i.e. the setup of the research) and the research questions are explained in section 2.4. The scope of our research is defined in section 2.5. The research approach (i.e. strategies used for information acquisition) is detailed in section 2.6. Finally, this chapter concludes with a chapter outline of the report.

2.1. Problem statement

In this research we focus on improving cost-, quality- and adoption aspects of semantic standards. Below, we formulate the problems as goals that need achieving.

- *Semantic IS standard development and maintenance costs need to be reduced.*
- *Semantic IS standard quality needs to be increased.*
- *Semantic IS standard adoption needs to be increased.*

We are not trying to solve these problems completely, because just a different approach to development will not eliminate these problems, instead we focus on solving problems that contribute to our goals. Cost-related problems can be reduced if a BRASD can decrease the development- or maintenance duration of semantic IS standards. Quality-related problems can be reduced if a BRASD can increase accuracy, completeness, consistency, compliancy, precision, traceability, understandability, and/or portability. Adoption-related problems can be reduced if a BRASD can increase understandability of the standard. Fig. 1 illustrates our problems and the proposed solution.

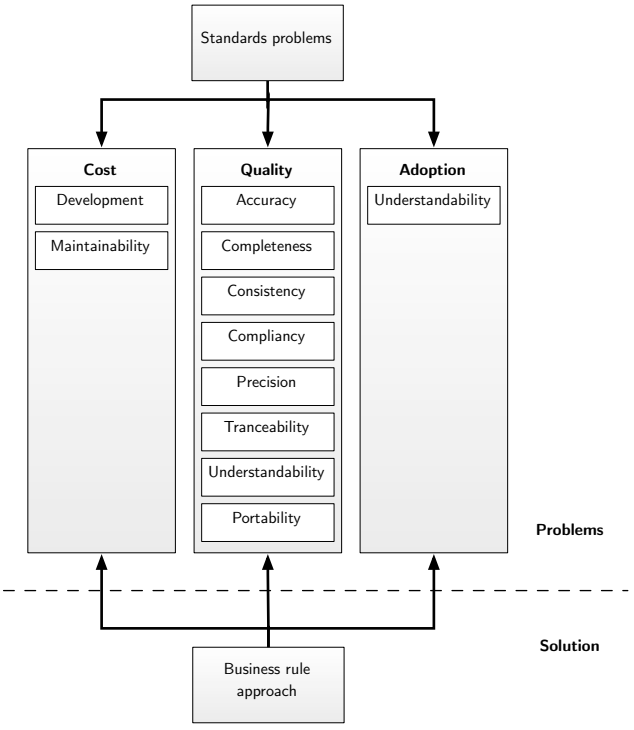


Figure 1. Proposed solutions to standards problems.

2.2. Research objectives

The objective of this research is to determine if a business rule approach can contribute to reducing the cost-, quality-, and adoption-related problems within semantic IS standards. Even though our focus is semantic IS standards, other kinds of standards might benefit from a business rule approach as well.

The contribution of this research will be a conceptual framework and an approach (BRASD) which can be applied in practice. If the BRASD proves itself valuable (i.e. it addresses some of the mentioned problems), then our framework and approach can serve as guide into the world of *business vocabularies*, *business rules*, *concepts*, *fact types*, and much more.

2.3. Research scope

We are not trying to address all of the problems that standards organizations are facing with a "holy grail". Instead we focus on the points mentioned in section 2.1. The goal of this research is to show that a BRASD can help to reduce cost-, quality-, and adoption-related problems within semantic IS standards. Other types of standards are out of scope. We will also not focus on the *implementation* of a rule-based semantic IS standard.

To evaluate our conceptual framework and BRASD, we will conduct an experiment (as part of a case study) in which we translate a semantic IS standard to business rules, that will serve as discussion material in an open interview with experts. The *SETU Standard for Reporting Time & Expenses 1.1* (Folmer, Roes & van Bekkum, 2009) from the 'Dutch list of open standards'[2] will be used for the case study. This standard is developed and maintained by SETU, which is an SSO that has a brach in Enschede. Access to both the standard and the SSO led to the choice of interviewing SETU.

2.4. Research methodology

This research is built as a design-science approach, which is fundamentally a problem solving paradigm (Hevner et al., 2004). An IS research (i.e. design-science research) consists of two main processes and one or more artifacts (March & Smith, 1995). The processes are (1) *development* and (2) *evaluation*. The *artifacts* that are produced are a conceptual framework that is a combination of constructs (vocabulary and symbols) and an approach (BRASD). Fig. 2 illustrates Hevner's framework which is the basis of this research. For better understanding of the problems and to add additional richness and detail, a case study will be added to the evaluation. This way the problems are approached from different perspectives, to provide additional insight. This merger of two approaches is known as 'multi-method' or 'complementary' research (Petter & Gallivan, 2004). In the next sub-sections we will explain how our research relates to Hevner's framework.

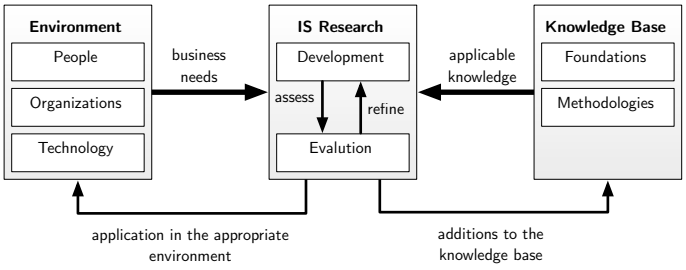


Figure 2. IS research framework (adapted from Hevner et al., 2004).

[2] This is called the 'pas-toe-of-leg-uit' list, which roughly translates to "comply-or-explain", see http://www.open-standaarden.nl/open-standaarden/lijsten-met-open-standaarden/pas-toe-of-leg-uit/ for more information.

2.4.1. Environment

The environment determines what the business needs are. In our case the environment consists standards development organizations (both SDOs and SSOs) and their business needs are formulated in 'problem statement' (section 2.1). We will apply our approach (BRASD) to the SETU standard and determine its utility in the environment it was created. It is good practice to evaluate the application of an artifact in the same environment that experiences the problems.

2.4.2. Knowledge Base

Our knowledge has been formed from different sources: a booklet, search engines, knowledge databases, and related work. The first source is a booklet we had direct access to that describes the state of the art (SOTA) on semantic IS standardization (Folmer & Verhoosel, 2011)[3]. It helped us understand the basic concepts of semantic standards. The second source of information comes from various search engines (e.g. Google Scholar, Web of Knowledge, and Wikipedia), knowledge databases (e.g. IEEExplore, arXiv, JSTOR, and ACM), and related work such as an unpublished thesis on rule-based process design (Stornebrink, 2010), which was another source that we already had direct access to.

Techniques for selecting applicable knowledge were predominantly 'forward citation search' followed by 'narrative reviews' and 'descriptive reviews' (King & He, 2005) followed by 'backward citation search'. 'Vote counting' articles that were pointed out by Folmer and Verhoosel (2011) were used as well.

For a SOTA on the business rule approach Wikipedia was used as a starting point. By searching for the term 'business rules' we have come across many key sources like the *Business Rules Group*, one of the first (if not the very first) organization focusing on business rules. Also key literature like *Principles of the Business Rule Approach* (Ross, 2003) and *The Business Rule Revolution* (Halle & Goldberg, 2006) was found this way. By doing a forward citation search on *Principles of the Business Rule Approach*, various sources have been found on which a backward citation search was done to find the related articles.

We have also specifically reviewed the work of Ross because he is considered a key figure in the development and evolution of the business rules approach. A simple Google search with 'business rules' and 'Ross' led to the *Business Rules Community*[4] which is an active community in the business rules domain and a good source of information on the business rules approach. Stornebrink's work has also served as an important source of information on the business rule approach — declarative process modeling specifically.

2.4.3. Development

The development (or build) process consists of defining the business needs and selecting applicable knowledge (Hevner et al., 2004), see sub-sections 2.4.1 and 2.4.2. The development process leads to two artifacts — a conceptual framework which describes characteristics of rule-based semantic IS standards and an approach (BRASD) which can be used to create rule-based semantic IS standards. By answering the following main research question we will be able to create these artifacts:

- *RQ — "What are the characteristics of a rule-based semantic IS standard that has advantages in relation with current technologies and how can it be created?"*

The technologies here refer to current technologies for standards development and maintenance such as manual remodeling of Unified Modeling Language (UML) diagrams if requirements from the environment change. Our main research question has been divided into multiple (sub) research questions to keep the whole manageable. Answering the research questions, will answer the main research question. The information that we get from answering each research question is explained below.

[3] The digital version of the booklet can be downloaded from http://www.semanticstandards.org/ or by searching Google with the terms 'state of the art', 'semantic', and 'standards'.

[4] http://www.brcommunity.com

- *Q1 — "What is a semantic IS standard?"*

  The answer to Q1 will give us an introduction on the basic concepts of semantic IS standards, which is essential for determining which parts of a semantic IS standard need to be expressed with business rules.

- *Q2 — "Which semantic IS standard concepts need to be expressed with business rules?"*

  Following Q1, the answer to this research question will give us a conceptual model of semantic IS standards, which will be used to determine which parts of the standard need to be expressed with business rules.

- *Q3 — "What is the business rule approach?"*

  The answer to the second research question will give us a SOTA on the business rules approach, which will help us to determine which business rules concepts, languages, best practices and drawbacks there are.

- *Q4 — "Which business rule concepts are needed to express semantic IS standards with business rules?"*

  The answer to this research question will be acquired in a similar way to Q2 and its result (i.e. a conceptual model of a business rule) will help us determine which types of business rules are needed to define the semantic IS standard concepts in business rules.

  The research model in fig. 3 shows which topics will be studied and how the research questions will be answered. The model consists of arrows and rectangles with straight and rounded edges. An arrow depicts a process flow and an arrowhead points to the output or answer to a research question (i.e. 'Qx', with $x \in 1..4$). All rectangular objects represent outputs, except for 'Evaluation', which is a processes. The evaluation is explained in the next sub-section. The framework is defined in chapter 5. Based on the framework, we will create an approach to rule-based semantic IS standards development (BRASD), which will be explained in chapter 5 and applied in chapter 6.

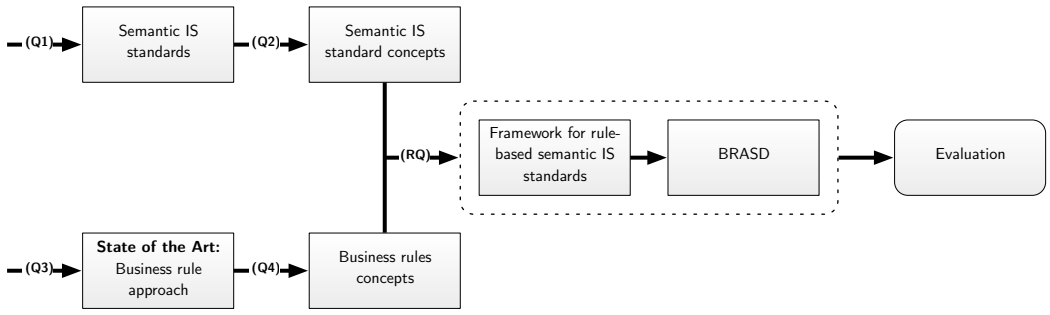


Figure 3. Research model.

2.4.4. Evaluation

The goal of the evaluation is to determine how complete and effective our research is, i.e. in what level our conceptual framework and approach satisfy the requirements and constraints of the problems that it was designed to solve/reduce (Hevner et al., 2004). The evaluation of the artifacts provides us with feedback information and a better understanding of the problems in order to improve our framework and approach (Hevner et al., 2004). The evaluation of our research will consist of a case study and the interpretation of the results. The case study consists of an open interview and a survey at SETU to evaluate if a business rule approach can help reducing the problems mentioned in section 2.1. The case study will be explained in detail in 7. The results of the case study will be assessed and serve as input for improvement of our method (i.e. 'refinement'). This research will be performed sequentially, which indicates that the results of one method serve as input for the other. See fig. 4 for our alternative IS design framework. The findings from the evaluation can either lead to a re-development process or to "lessons learned" in the conclusions.

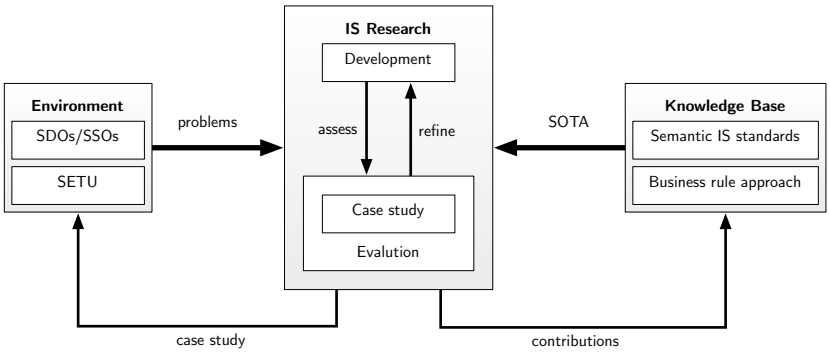


Figure 4. Alternative IS research framework.

2.5. Report structure

The remainder of this report is structured as depicted in fig. 5.

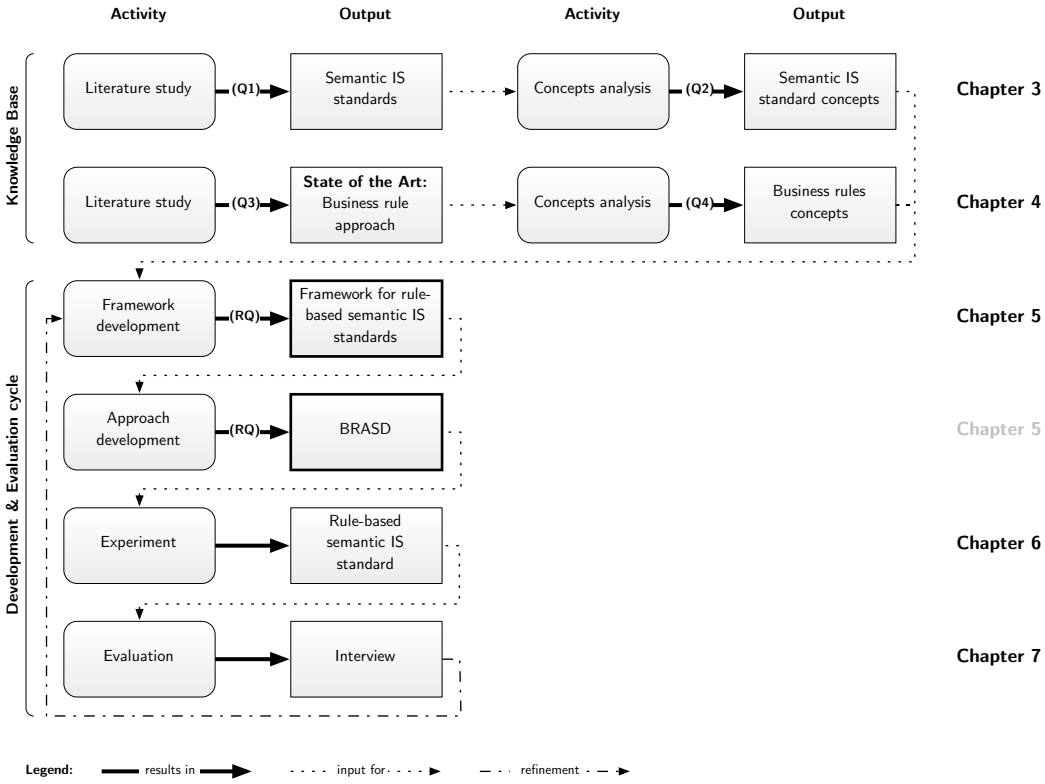


Figure 5. Research process.

- Chapter 3 gives an introduction on the topic of semantic IS standards. This results in a basic understanding of the topic and serves as input for concepts analysis, which leads to a conceptual model of semantic IS standards. This chapter answers research questions *Q1* and *Q2*.
- Chapter 4 gives a SOTA on the topic of the business rule approach. This results in a basic understanding of the topic and serves as input for concepts analysis, which leads to a conceptual model of business rules. This chapter answers research questions *Q3* and *Q4*.
- Chapter 5 provides a conceptual framework describing characteristics of rule-based semantic IS standards and an approach (BRASD) to create rule-based semantic IS standards. The framework consists of characteristics of semantic IS standards, classification of rule sentence templates. The

approach consists of a method to create rule-based semantic IS standards and guidelines for its application.

- Chapter 6 describes an experimental application of our BRASD approach to the SETU standard. The result is a rule-based semantic IS standard.
- Chapter 7 is the evaluation of our research consisting of the interview and a survey at SETU and the interpretation of the results.
- Chapter 8 concludes this research by drawing conclusions and discussing limitations and future work.

3. Semantic IS standards

This chapter provides an introduction to semantic IS standards. It will answer the first research question *"What is a semantic IS standard?"* and the second research question *"Which semantic IS standard concepts need to be expressed with business rules?"*. First, the definition and the aim of standardization in general is explained followed by defining semantic IS standards from literature in section 3.1. Then the second research question is answered by analyzing a conceptual model of semantic IS standards (section 3.2) and determining which concepts can be expressed using business rules (section 3.3). Finally, this chapter concludes with a summary in section 3.4.

3.1. What are standards?

Standards are used 24/7 to create the world around us. From electronic devices like mobile phones, computers, and televisions, to paper, bottles and even french fries at the McDonalds, all are produced using some (international) standard. Standards can support whole nations and even change them (Krislov, 1997). Standards are developed and published by SDOs and SSOs. They exist on national, regional, and global level (Folmer & Verhoosel, 2011). SDOs are considered formal/traditional development organizations (Spivak & Brenner, 2001). All the other standards development organizations are considered SSOs, they also exist on national, regional and global level. The world's largest developer and publisher of international standards is ISO which is an SDO. ISO defines a standard as:

- *"... a document, established by consensus and approved by a recognized body, that provides, for common and repeated use, rules, guidelines, or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context. Standards should be based on the consolidated results of science, technology and experience, and aimed at the promotion of optimum community benefits."*

Even though this definition is considered somewhat biased towards ISO in general (van Wessel, 2008), it is widely accepted. Based on an ISO booklet from 1972, Spivak and Brenner (2011) mention that standardization aims to provide the following:

- *Simplification for society* to prevent unneeded variation in products.
- *Interchangeability*, because simplification tends to limit variation and increases interchangeability.
- *Standards as a means for communication* between producers and customers by specifying with which requirements (ordered) goods will comply.
- *Symbols and codes* to eliminate the effects of differences in languages.
- *Safety by uniforming products*, so variation in failures is minimized.
- *Meeting consumer and community interest* in properties like durability, energy consumption, flammability, etc. by using product labels that report the results of (inter)national tests carried out by certified laboratories.
- *Reduction in trade barriers* to prevent individual nations from excluding the products of others, e.g. by imposing unique standards on imported goods.

Standards are there to help us work/produce in a structured manner to get the result we want in a pre-defined quality. Semantic IS standards (the focus of this research) are considerably different from IT standards, e.g. in IT product standardization it is common that various small groups standardize a certain IT product in their way (Lim, 2006). Within semantic IS standardization this is not likely because semantic IS standard need the support of all stakeholders (Folmer & Verhoosel, 2011).

3.1.1. What are semantic IS standards?

Folmer et al. (2011a) mention that semantic IS standards are often referred to as Vertical Industry Standards (VIS) but that VIS exclude horizontal standards and government-related standards. To define semantic IS standards properly, they have adapted the VIS definition to include horizontal and government-related standards (adapted from Steinfield, Wigand, Markus & Minton, 2007):

- *"Semantic IS standards are designed to promote communication and coordination among the organizations; these standards may address product identification, data definitions, business document layout, and/or business process sequences."*

So in short, semantic IS standards prescribe guidelines for inter-system communication to increase interoperability and the quality of information systems (Folmer & Verhoosel, 2011). The better tactics we use to develop these standards, the better the quality will be, the more likely it is that they will get adopted (Markus et al., 2006). The VIS definition distinguishes the following concepts: *product identification*, *data definitions*, *business document layout*, and *business process sequences*. These concepts gives us an indication of what needs to be expressed using business rules.

3.2. Conceptual model

To be able to evolve semantic IS standards, Folmer et al. (2011b) argued that a better understanding of the conceptual structure of these standards is needed (Folmer, Oude Luttighuis & Van Hillegersberg, 2011). They have composed a conceptual model based on extensive research of 72 scientific papers on semantic IS standards, a fragment of the model is illustrated in fig. 6. The complete model can be found in appendix A.

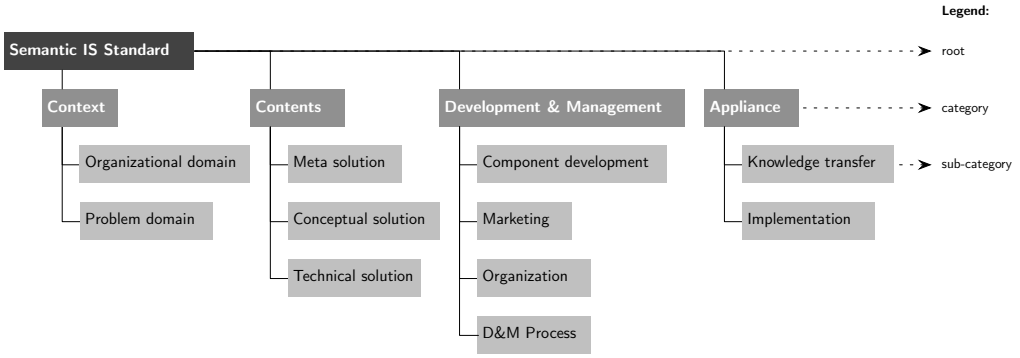


Figure 6. Conceptual model of a semantic IS standard (adapted from Folmer et al., 2011).

There are three levels in this conceptual model: (1) categories; (2) sub-categories; and (3) concepts. Starting from the third level, there are 34 concepts in total. These concepts are assigned to sub-categories to keep them comprehensible, see fig. 49 in appendix A for a complete overview. The sub-categories are assigned to four categories, which are:

- Context
- Contents
- Development & Management
- Appliance

The *context* is the environment where the standard can be applied, i.e. the environment where a solution for a problem is needed. The actual solution is defined in the *contents* of the standard. The third category is *development & management*, because every standard needs to be developed and maintained. Finally, the

usage is described in the *appliance* category. Out of these four categories, the *contents* category is the most important one for this research as it is considered the core of semantic IS standards (Folmer et al., 2011). The other three categories (context, development & management, and appliance) are left out of scope.

3.3. Contents of semantic IS standards

The *contents* category is regarded as the core of a semantic IS standard, which means that it is a determent for quality of the standard. It consists of the sub-categories: (1) meta solution; (2) conceptual solution; and (3) technical solution, see fig. 7. These three concepts are explained in the following sub-sections.

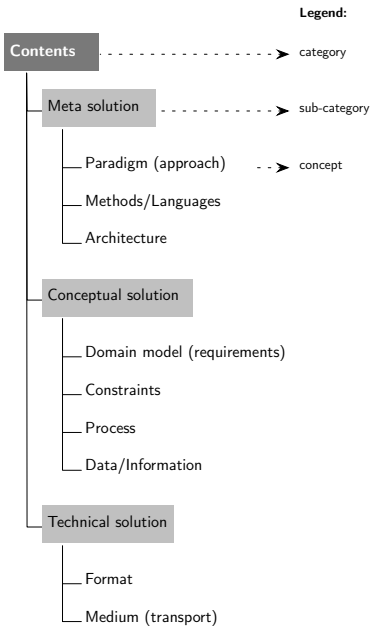


Figure 7. Concepts in the content category of a semantic IS standard.

3.3.1. Meta Solution

The meta solution of a semantic IS standard describes the selected approaches that were used to develop the standard. It has the following concepts:

- Paradigm (approach)
- Methods/Languages
- Architecture

The *paradigm* concept describes high level thoughts behind the design approach. The *methods/languages* concept describe the methods and languages used in designing the standard. The *architecture* concept describes the architectural design choices made in terms of functional architecture, technical architecture, relationship with other standards, or selecting new techniques is part of architecture (Folmer et al., 2011).

3.3.2. Conceptual solution

The conceptual solution of a semantic IS standard describes the design of the standard in terms of descriptions and models, it includes the concepts:

- Domain model (requirements)

- Constraints
- Process
- Data/Information

The *domain model* describes the requirements to the environment in which the standard is applied and the scope of the standard. The *constraints* concept refers to constraints that determine the scope and use of the standard. They can also express data dependencies based on the process status (Folmer et al., 2011). Constraints are usually expressed with business rules. The *process* concepts entails the flow of activities within the standard, e.g. process diagrams, actors, etc. Finally, the *data/information* concept refers to the data and information entities that are represented within the standard. These can be messages, documents, ontologies, etc.

3.3.3. Technical solution

The technical solution of a semantic IS standard describes the design of the standard in a more detailed level of abstraction in terms of technical artifacts. This sub-category includes the following two concepts:

- Format
- Medium (transport)

The *format* concept describes the format of the technical solutions in which the conceptual solution are represented, for semantic IS standards this is typically in UML, Extensible Markup Language (XML) or XML Schema Definition (XSD) document format. The *medium* represents the communication aspects of a standard.

3.4. Summary

We discussed briefly what standards are and why they are used. By looking at the concepts of semantic IS standards, we have set of concepts that our framework needs to be applied to. We highlighted a total of nine concepts that form the *contents* of a semantic IS standard. These concepts cover a fair amount of the concepts of the VIS definition (section 3.1):

- *Product identification* is covered by the concepts *domain model* and *constraints.*
- *Data definitions* is covered by the concept *data/information.*
- *Business document layout* is covered by the concepts *architecture*, *format* and *medium.*
- *Business process sequences* is covered by the concept *process.*

4. State of the Art: Business rule approach

This chapter contains a SOTA on the business rule approach, which answers the third research question *"What is the business rule approach?"* and the fourth research question *"Which business rule concepts are needed to express semantic IS standards in business rules?"*. First, the definition of a business rule is given in section 4.1. This is followed by the principles of the business rules approach in section 4.2. The business rule approach concepts (structure, power, and control) are explained with a 'running example' in section 4.3. A suitable rule modeling language (RML) is selected in section 4.4. The (dis)advantages of business rules are discussed in section 4.5. Business rule management (BRM) is explained in section 4.6. Additionally, tooling is discussed in section 4.7. Finally, this chapter concludes with a summary in section 4.8.

4.1. What are business rules?

Business rules haven been used formally since the late 1980's and they can be found in most organizations on some level, but their effects affect our daily lives. The Business Rules Group (BRG) formulates statements and supporting standards about the nature and structure of business rules, the relationship of business rules with the way an enterprise is organized, and the relationship of business rules with systems' architectures. There are active business rule communities, such as Business Rule Community, The Rule Markup Initiative, the Dutch BR-Platform, and many more. There are also organizations that help the development and implementation of business rules in practice as well, such as Business Rules Solutions. Business rules give the enterprises true agility (Ross, 2010) but they can do true damage as well, if the enterprise is not aware of their existence. Business rules have the most impact (i.e. can do the most damage) when they are unseen or unknown (Halle & Goldberg, 2006). So what are business rules?

4.1.1. Definition

Business rules have been defined numerous times since we became aware of them (Ross, 2003). Business rules can be viewed from different perspectives: (1) from an information system perspective, where business rules relate to entities (i.e. data in the system) and to the constraints on the values of those entities and (2) from a business perspective, business rules relate to the (human) behavior of people in the enterprise. Because these two perspectives are distinct, a definition for a business rule form each perspective is given (Hall & Healy, 2000).

From a *business perspective*, business rules can be defined as:

- *"... a statement that defines or constrains some aspect of the business. It is intended to assert business structure, or to control or influence the behavior of the business."* (Hall & Healy, 2000)

With addition of the following statement:

- *"This [statement] must be either a term or fact, a constraint, or a derivation. It is 'atomic' in that it cannot be broken down or decomposed further into more detailed business rules. If reduced any further, there would be loss of important information about the business."* (Hall & Healy, 2000)

The definition from the business perspective is all about the semantics of a business rule. The terms 'term', 'fact', 'constraint' and 'derivation' are explained in section 4.3. A more idealistic definition from a business perspective comes from Von Halle & Goldberg (2006), they define business rules as:

- *"... the ultimate levers with which business management is able to guide and control the business. In fact, the business's rules are the means by which an organization implements competitive strategy, promotes policy, and complies with legal obligations."*

From an *information systems perspective*, business rules can be defined as:

- *"An atomic piece of re-usable business logic, specified declaratively."* (Ross, 2003)

The term 'atomic' means the same thing as in the additional statement in the business perspective above. The term 'declaratively' is explained below. Business rules determine how an enterprise acts in certain situations, but what do they look like?

There are many systems that have rule engines of the condition-action (CA) or event-condition-action (ECA) types, both of which can be executed by a computer. For example: the most e-mail clients have a rule system to filter out incoming mail.

A CA type rule has the structure:

```
if <some condition is fulfilled> do <perform some activity>
```

An ECA rule has the structure:

```
when <some event has happened> and if <some condition is fulfilled> do <perform some activity>
```

The CA and ECA type of rules are called "imperative rules" or "production rules", because they command actions to be executed by computers which in turn produce actions (Joosten, Wedemeijer & Michels, 2010).

In this research we use business rules as constraints, also known as "declarative rules", because they declare what must happen, but not necessarily prescribe a particular action. This is best explained by an example of a business rule and its *formal rule statement* (i.e. an expression of a business rule in a specific formal grammar).

Statement: An order cannot be empty

This formal rule statement is somewhat ambiguous as we are missing some information, e.g. "what does an order need to have?", "in which situations?", etc. The formal rule statement of the example above can be expressed in an RML (formal expression language) like Structured English, with either one of the following three business rules:

```
Each order always has at least one item
```

The same rule can be defined in another RML like RuleSpeak as:

```
An order must have at least one item
```

Or even in predicates:

```
∀ order ∃ item ∈ order
```

These three business rules mean exactly the same thing and they are all declarative — the do not prescribe *how* things need to be, but *what* must be. This will be discussed in detail in section 4.3.

Not every statement is a rule, which can be confusing sometimes. To decide whether a statement is a rule, Joosten et al. (2010) have defined three definitions with an example, shown in fig. 8.

A business rule is a verifiable statement that some stakeholders intend to obey, within a certain context. Here is an example of a rule:

`In our club, a coat of any guest shall be in the cloakroom, as long as the guest is in the club.`

Let us analyze this statement, to better understand what we mean by a business rule.

1. **Rules have a scope.**
   The context of the stakeholders is our club in which this rule is valid. We call this the scope of this rule.
2. **Rules have stakeholders.**
   Anyone involved in a rule is called stakeholder. If a rule has no stakeholders, then no one is interested if the rule is obeyed or violated. Such a rule has no business merit, and we do not consider it to be a business rules. For example, guests must put their coats in the cloakroom, there may be a bell boy to take them in and hand them out again, there may be staff who sees to it that people who try to smuggle their coats inside are intercepted, etcetera. Stakeholders who "live by the rules" are working to satisfy rules, each in his or her own role.
3. **Rules are verifiable.**
   If there is a guest inside the club who holds a coat, we have a violation of this rule. We call a rule verifiable if its violations can be spotted unambiguously and objectively. That violation could be a signal to someone to take action. A floor manager might summon the offender to take his or her coat out to the cloakroom. Or, a staff member might take the coat from the guest and put it away in the cloakroom. Or even the guest himself might take some action. He might toss his coat out of the window, ensuring that it is beyond the scope of this rule. Technically, that would be an acceptable thing to do, unless there are rules in place that forbid littering the street... Whatever actions happen, the situation should be restored to where the rule is complied with.

.Figure 8. Example of statements and rules (adapted from Joosten et al., 2010).

Now that we know what business rules are we can discuss what the principles of the business rule approach are in the next section.

4.2. Principles of the business rule approach

The business rule approach can be defined as (Halle & Goldberg, 2006):

- *"... a formal way of managing and automating an organization's business rules so that the business behaves and evolves as its leaders intend."*

Von Halle & Goldberg (2006: p.17-31) have conducted a maturity survey among anonymous companies to establish a status quo of the business rules marketplace. The survey showed that major motivations for adopting the business rule approach are: (1) agility; (2) consistency; (3) business control or empowerment (being closer to the authoring, resting and discussing of rules); (4) knowledge retention; and (5) legacy renewal (Halle & Goldberg, 2006: p.21). Business rules offer high levels of automation as well as flexibility to adapt to rapidly changing business requirements (Halle & Goldberg, 2006; Paschke, 2006; Ross, 2003). They have advantages such as being expressed in natural language and thus being understandable by business people. Business rules also force users to think about the rules of the business and reduce miscommunication about business concepts (Ross, 2003).

Ross (2003) defined a set of ten principles to apply the business rule approach in practice. He states that:

1. **Rules should be written and made explicit** — When a rule is important enough, it should be made explicit and written down.

2. **Rules should be expressed in plain language** — If rules can't be interpreted well, they lose their purpose.
3. **Rules should exist independent of procedures and workflows** — A procedure is not always needed and sometimes it is crucial to separate them. For business rules based enterprises it is crucial that business processes and business rules are separated in the architecture. By separating them in this point the architecture enables business rules to be changed independently of process or the other way around (Halle & Goldberg, 2006: p. 57).
4. **Rules should build on facts, and facts should build on concepts as represented by terms** — This is one of the basic principles of business rules.
5. **Rules should guide or influence behavior in desired ways** — Meaning that rules should be complete and should serve the business instead of holding it back.
6. **Rules should be motivated by identifiable and important business factors** — Business rules are *"the ultimate levers with which business management is able to guide and control the business"* (Halle & Goldberg, 2006), they should serve the business to achieve its goals.
7. **Rules should be accessible to authorized parties** — If rules are not accessible they are forgotten and lose their purpose.
8. **Rules should be single-sourced** — This ensures consistency.
9. **Rules should be specified directly by those people who have relevant knowledge** — Rules are based on knowledge and therefore the people with the knowledge should be involved in specifying them.
10. **Rules should be managed** — Because rules can have a great impact on the business, all changes to them must be reviewed, approved, incorporated, and disseminated carefully. Some rules might even become obsolete or even dangerous. Section 4.6 discusses business rule management in more detail.

Ross (2003) points out an interesting thing about the business rule approach:

"...it did not arise as a response to any emerging new class of software tools — knowledge-oriented or otherwise. Rather, the business rule approach is a real-world, grassroots movement whose driving force is business success, not technology".

The principles will help us form our conceptual framework in chapter 5.

4.3. Business rules concepts

Ross (2003), who is recognized internationally as the "father of business rules"[5], identifies three components of business systems, i.e. (1) structure; (2) power; and (3) control. Ross uses the analogy of the human body to describe these components (Ross, 2003). To explain *structure*, *power* and *control*, we have defined a fictive example ("handle claim") which will be used throughout this section. These components will be used to shape BRASD in chapter 5.

For example: Imagine an insurance company that gets claims (i.e. medical bills that need payment) from clients (i.e. medical institutions and individuals). Each received claim is either accepted or rejected based on some decision logic. Every individual has an 'own risk' which has a limit determined by the government. If a claim is accepted, it is processed to see if the 'own risk' has ben exceeded. If it has not been exceeded,

[5] Ronald G. Ross is recognized internationally as the "father of business rules". He was a charter member of the BRG in the 1980s, and an editor of two landmark BRG papers, 'The Business Motivation Model' and the 'Business Rules Manifesto'. He is active in standards development, with core involvement in Semantics of Business Vocabulary and Business Rules (SBVR).

the insurance company pays the medical institution directly, then the process terminates. If the own risk has been exceeded, the insurance company pays the medical institution, but also sends an invoice to the client (i.e. individual) as well, which then if paid, terminates the process. If it is not paid, another invoice is sent to the individual. Fig. 9 illustrates this process in a process flow diagram.

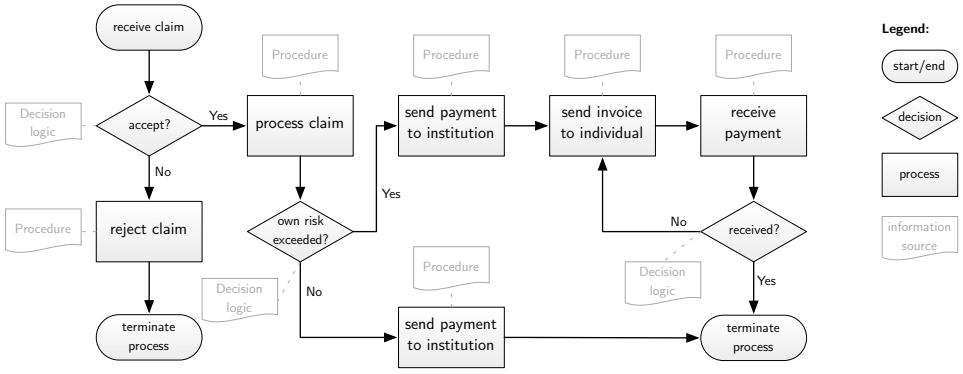


Figure 9. "Handle claim" process flow diagram.

Note that every process has a *procedure* belonging to it, and every decision some *decision logic*. Changes in the decision logic could change the process and vice versa. We will use this example to explain the three components of business systems.

Before starting with defining structure, a business vocabulary needs to be created and relevant concepts should be defined. For this example, we assume that his has already happened.

4.3.1. Structure

The Object Management Group (OMG) states that "rules are built on facts, and facts build on concepts as expressed by terms" (Object Management Group, 2008).

Basic business knowledge is structured with *terms* and *fact types*. Terms form concepts that have a meaning for the business, e.g. customer and fact types express relationships between two or more concepts. It is important to mention the difference between concepts and instances. The business vocabulary represents concepts of entities rather than instances of those entities. For example, an insurance company might have thousands of clients, but they are represented by the concept `client`. A fact type represents relationships between concepts, whereas *facts* represents an *instance* of those relationships, e.g. '`Thomas sends claim#1234`' is of fact type '`client sends claim`'.

Every term that has some value for the business needs to be defined in the business vocabulary. Both terms and fact types can be a type of business rule, which in turn can consist of other terms and fact types. This abstraction goes far as its level has value for the business. For example: a client is a person or a medical institution. It might be valuable for the business to define a what a person is and what a medical institution is to be able to separate individuals from organizations when invoicing. If defining the definition, doesn't add value, then it should stop there.

When concepts get connected to each other through fact types a structure emerges, this is also called a 'structural assertion' (Hall & Healy, 2000). The BRG defines a structural assertion as follows:

- *"... a statement that something of importance to the business either exists as a concept of interest or exists in relationship to another thing of interest. It details a specific, static aspect of the business, expressing things known or how known things fit together."* (Hall & Healy, 2000).

Ross (2003) compares this structural assertion to the human skeleton where the individual bones represent terms (or concepts) and the ligaments that connect the bones represent the fact types. Another word for structural assertion is a fact model; *"a drawing or diagram of the complete human skeleton ... to illustrate the overall structure of terms and facts"* (as cited by Ross, 2003).

In our example, entities like `claim`, `client`, `medical institution`, `individual`, `payment`, `own risk`, `invoice`, etc. are considered concepts (expressed in terms). Relationships like `claim` ***`belongs to`*** `client` and `individual` ***`is treated by`*** `medical institution` are considered fact types. Fig. 10 depicts the fact model (i.e. structure) for our example. The dashed lines represent a relationship, the thick lines represent 'category of', e.g. both a `rejected claim` and `accepted claim` are claims.

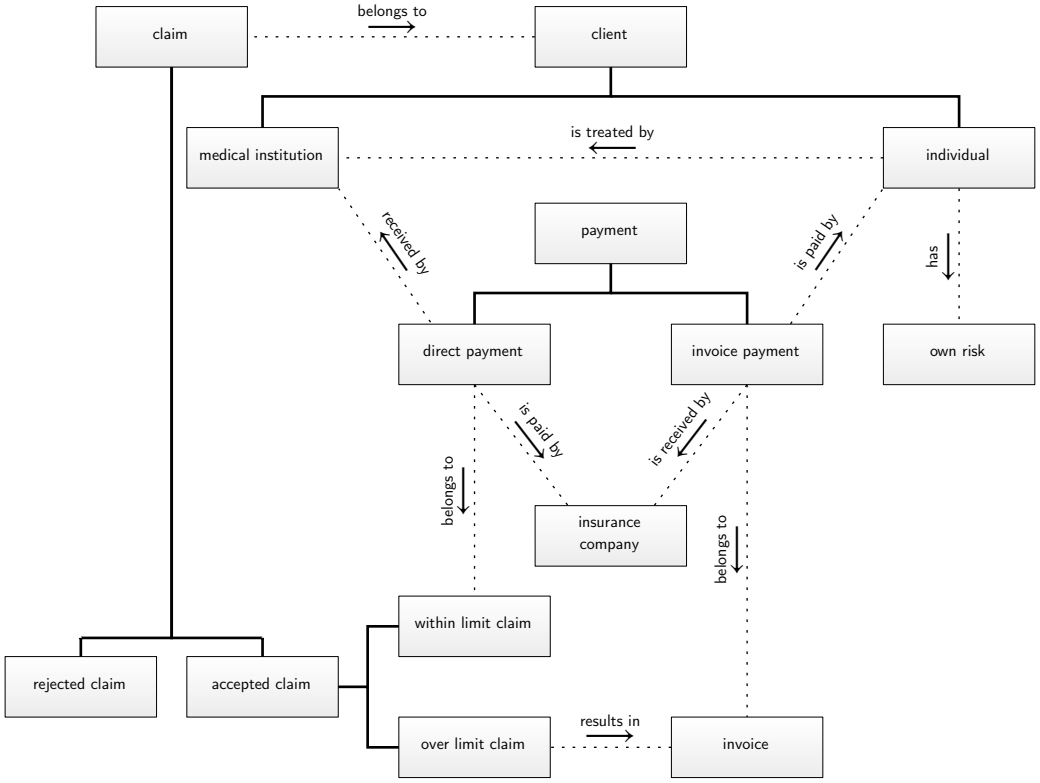


Figure 10. "Handle claim" fact model.

We see that there are four groups of categorization: medical institution and individual are both clients; rejected claim and accepted claim are both claims; within limit claim and over limit claim are both accepted claims; direct payment and invoice payment are both payments. All of these concepts and fact types need to be defined in the business vocabulary to create the structure.

4.3.2. Power

Processes are what produce products or services, they are what give a business system its power. They are compared to the muscles of the human body that can move the skeleton around (Ross, 2003). Just like muscles are connected to the skeleton, so are processes connected to the concepts and fact types of the business. Processes and business rules are highly related but should be separated according to the business rule principles, see section 4.2.

The idea of business process management (BPM) comes from the observation that products and services are the result of a series of activities performed (Weske, 2007). BPM is considered a holistic approach that

focuses on aligning the organization from a business and IT perspective in an effective and efficient way to meet the needs of its customers and to stimulate flexibility and agility at the same time (Brocke & Rosemann, 2009). The series of activities are also called business processes and they are key to understanding how the organization works and reacts to the environment. Business processes are defined as:

- *"A structured, measured set of activities designed to produce a specific output for a particular customer or market."* (Davenport, 1992).

There are two distinct approaches to business process modeling, i.e. *procedural process modeling* and *declarative process modeling*.

Procedural process modeling

Business processes are called procedural when they contain explicit information about *how* the process should proceed (Goedertier & Vanthienen, 2007), i.e. every possible business scenario that is allowed must be modeled explicitly (Stornebrink, 2010). Procedural process modeling is based on imperative languages that specify the "how" and this makes todays systems rigid (Pesic & Aalst, 2006), e.g. Business Process Execution Language (BPEL), Business Process Modeling Notation (BPMN) and UML Activity Diagrams. The disadvantage of procedural modeling is that business rules cannot be formulated independently from process models (Goedertier & Vanthienen, 2007), so when a process changes, all of the process models need to be examined for changes. The process flow model of the "handle claim" process is a procedural process, see fig. 9.

Declarative process modeling

The second kind of approach for business process modeling is a declarative one. A declarative approach specifies the "what" of business process modeling and makes the governing aspects of business processes explicit in the form of business rules. A process is declarative when it explicitly takes into account the business concerns that govern business processes (i.e. business rules) (Goedertier & Vanthienen, 2007), but not *how* they are enforced. Declarative also means that the rule makes no reference to any business event or update event (Ross, 2003). This way business rules modeling can be separated from business rule enforcement (in the form of process flows), which is the third principle of the business rule approach (Goedertier & Vanthienen, 2007; Ross, 2003). More specifically it is crucial for rule-driven enterprises that business processes and business rules are separated in the architecture. By separating them at this point, the business rules can be changed independently of process or the other way around (Halle & Goldberg, 2006: p.57). Table 1 provides an overview of the differences between procedural- and declarative process modeling.

	Procedural modeling	**Declarative modeling**
Rule enforcement	imperative (what, when, how)	declarative (what)
Business concerns	implicit	explicit
Execution scenario	explicit	implicit

	Procedural modeling	Declarative modeling
Execution mechanism	state-driven	goal-driven
Modality	what must	what must, ought, can

Table 1. Procedural vs. declarative modeling (adapted from Goedertier & Vanthienen, 2007).

Based on the work of Joosten & Joosten (2007) and Goedertier (2008), Stornebrink (2010) has proposed a framework for rule-based BPM. This framework enables us to model processes declaratively, which is needed for a *complete* business rule approach to semantic IS standards as process models are a part of semantic standards. The next sub-section will briefly discuss the principle behind rule-based BPM.

Rule-based process management

Joosten & Joosten (2007) describe rule-based process management as follows:

- *"The principle of rule-based process management is that any violation of a business rule may be used to trigger actions".*

This means that every time a business event occurs, a rule engine checks all related business rules for violations. If a violation is detected, some action needs to be taken. This can be an automated action if the violations are maintained by the system. If they are not maintained by the system, it prompts users for action in the form of signals. This functionality is provided by projector rules which will be detailed in section 4.3.3. Fig. 11 illustrates the principle of rule-based BPM. Before we can explain how this works for our example, we need to define business rules that will guide and constrain our process i.e. exert control. For readability purposes, we will define these business rules and illustrate how the principle of rule-based BPM works in section 4.3.3.

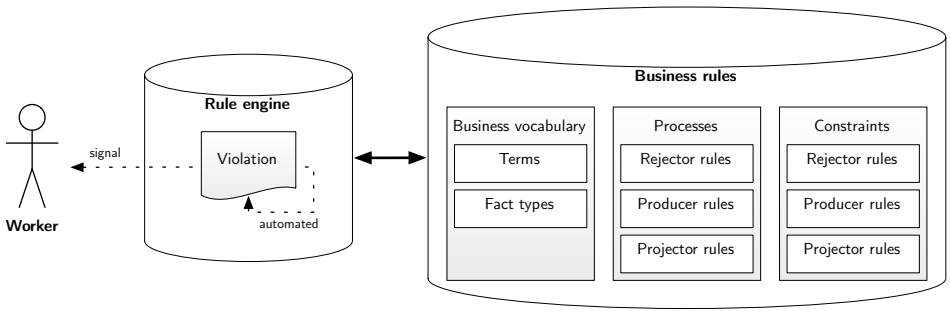


Figure 11. Principle of rule-based process management.

This principle has been translated into a framework for rule-based process design (Stornebrink, 2010). A process can be modeled with three kinds of rules (i.e. conditions): (1) pre-conditions (*precdx*); (2) post-conditions (*postcdx*); and (3) boundary-conditions (*boundcdx*). A process consists of one pre-condition, which defines the condition upon which it may start; one post-condition, which defines the condition upon it may terminate; and any number of boundary-conditions, each of which need to be satisfied within the process at any time. These principles have been translated into a set of 11 characteristics, see fig. 12.

1. **A process can only start if the *precdx* rule is met.**
2. **All work done within a process is aimed at fulfilling the *postcdx* rule.**
3. **All word done within a process is constrained by *boundcdx* rules.**
4. **Complex processes can be decomposed into more basic processes by means of subprocess.**
5. **Subprocess is anti-symmetric and irreflexive**
   An irreflexive process cannot be a subprocess of itself and anti-symmetry prohibits child objects to be linked back to their parent by the same relation.
6. **The *postcdx* rule of a subprocess process must contribute to the *postcdx* rule or *boundcdx* rule of its parent process.**
7. **Every process shall maintain every constraining *boundcdx* rule of its parent processes.**
8. **A business concept is the define-time specification of data sets that are to be treated according to a specific set of rules.**
   A business concept here is the same as a concept (i.e. term) to which the process in question refers to.
9. ***Method*: A process is defined as a method to operate on instances of a business concept.**
10. **An atomic activity is the way of executing a process (i.e. procedure or script).**
11. ***Proclet*: Every process that is not decomposed into sub processes must be assigned an atomic activity.**

Figure 12. Rule-based process model principles (adapted from Stornebrink, 2010).

Fig. 13 depicts the principles for rule-based process design. The conceptual model is depicted in ADL[6].

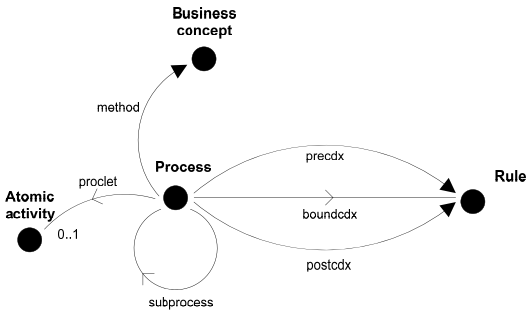


Figure 13. Conceptual model of rule-based process design in ADL (adapted from Stornebrink, 2010).

Note that Stornebrink's framework has been proven to work with a proof-of-concept. However there are no open-source or commercial implementations of rule-based BPM as described by Joosten & Joosten (2007). To be able to provide a complete business rule approach to semantic IS development, we will use the principle of rule-based BPM. How this would work our example, will be explained in the next sub-section.

4.3.3. Control

Control is enforced by business rules, since they constrain a certain aspect of the business. These kinds of business rules are also called 'action assertions' (Hall & Healy, 2000). Where structure (i.e. structural assertions) defines possibilities, control (i.e. action assertions) imposes constraints on the behavior within the business. Control rules concern the dynamic aspect of the business. They are usually recognized

[6] ADL stands for *A Description Language* and is based on relational algebra. It can be used to define a repository in which rules can be stored within their contexts. It has a graphic modeling technique as well, which is depicted here (Joosten et al., 2010).

through the words '`must`' (or '`should`') or '`must not`' (or '`should not`'). The BRG definition of an action assertion is as follows:

- *"A statement that concerns some dynamic aspect of the business. It specifies constraints on the results that actions can produce."* (Hall & Healy, 2000).

Control rules can usually be seen in the processes of a business system (Ross, 2003). Ross compares control rules to the nervous system of the human body. Just like nerves trigger muscles to move in a certain way, so can business rules control or constrain processes. Ross (2003) distinguishes three types of rules, based on the type of control they exercise: (1) rejectors; (2) producers; and (3) projectors. All rules fall in one these three categories. A complete list of control rules, can be found in table 12, appendix B. From this point on, the following typographical convention will be used to define different aspects of business rules. The rules will be depicted in Structured English.

- terms are underlined
- *fact types* are given in italics
- literal values and instance names are shown with double underlines
- keywords are shown in bold font
- uninterpreted text is shown in normal font style

Rejector rules

Most rules are naturally *rejector rules* (or constraints), they simply reject any event that would violate the rule. Looking at our running example, the following business rules (BR) would be rejectors:

BR1 A rejected claim may not *be processed*

BR2 A claim from an unknown client *is rejected*

BR3 A claim with a treatment date *of* claim older than one year *is rejected*

BR4 An invoice payment must *be paid* within the payment limit

BR5 An invoice payment must *be based on* an invoice

Producer rules

Then there are *producer rules* that neither rejects or project events. Producers compute or derivate values based on 'computation rules' and/or 'derivation rules'. Computation rules are any producer-type rules that compute a value with standard arithmetic operations (e.g. sum, multiply, average, etc.). Derivation rules are any producer-type rules that derive a boolean value (i.e. `true` or `false`). based on logical operations (e.g. `AND`, `OR`, `NOT`, `EQUAL TO`, etc.).

The BRG calls these kinds of rules 'derivations' (Hall & Healy, 2000). In our example, the following rules would be producers:

BR6 The payment limit is equal to 14 days

BR7 Rejected claim means status *of* claim is rejected

BR8 Unknown client means clientNumber *of* client does not exist

Projector rules

The third kind of control rules are *projector rules*. Projectors always accept events and "handle" them by taking some action (i.e. forwarding them to another process or business rule), they are also known as

stimulus/response rules. These kinds of rules are the main ingredient for designing rule-based processes (Stornebrink, 2010).

There are three types of projector rules: (1) enablers; (2) copiers; and (3) executives. *Enablers* are projector-type rules that act as a toggle that can be switched on or off. *Copiers* are projector-type rules that copy actual values. *Executives* are projector-type rules that cause an operation, process or procedure to execute or a rule to fire. In our example, the following rules would be projector rules:

BR9 Every claim must be either accepted or rejected

BR10 Every accepted claim must *be processed*

BR11 Every accepted claim must *result in* an invoice or payment

BR12 Every accepted claim must be either within the own risk limit or over the own risk limit

BR13 Every direct payment must be executed for every within limit claim

BR14 An invoice must *be sent* for every over limit claim

BR15 An invoice *is sent* to an individual

BR16 Every invoice must *be paid*

These sets of rules allow us to explain rule-based BPM. The principle of rule-based BPM is that any violation of a business rule may be used to trigger actions. This principle implements the well known 'Plan-Do-Check-Act' cycle. Fig. 14 depicts the engine cycle for a rule-based process engine.

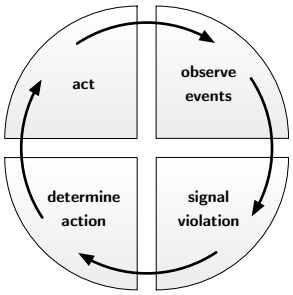


Figure 14. Engine cycle for a rule-based process engine (adapted from Joosten et al., 2010).

A detector observes events and analyzes them as they occur in the IT system and detects when rules are violated. The logic to detect violations is derived from the business rules database. Whenever a rule is violated, a signal is raised to trigger either automated or human action. These actions can cause other rules to produce signals by which other actors are triggered. In our example, the following scenario is possible with a rule-based process engine:

1. A client sends a claim (event). The rule engine detects a claim which is not accepted or rejected. This is a violation of BR9. A signal is sent to the user to either accept or reject the order.
2. The user accepts the claim (event) and BR9 is no longer violated. Now the rule engine detects that BR10 is violated. A signal is sent to the user to process the claim.
3. The user processes the claim (BR10 no longer violated) and sets that the client has exceeded his 'own risk' (BR12 is no longer violated). The rule engine detects that no invoice is sent for the 'over limit claim' (which this claim has become) and thus BR14 has been violated. A signal is sent to the user to send an invoice to the individual.

4. The user sends the invoice to the individual (event) and `BR14` is no longer violated. The rule engine detects (periodically) that an invoice has not been paid, which is a violation of `BR16`. This violation can be signaled as often as specified or it can result in some other action, depending on what the business wants. For this example we assume that this is signaled once a week.
5. The client pays the invoice and the rule engine detects this, `BR16` is no longer violated. There are no violations left, the process is complete.

This illustrates the rule-based BPM approach, however we still haven't seen a rule-based process yet, below we define one using Stornerbrink's (2010) framework. More rule-based processes will be defined in the case study in chapter 6.

```
send-invoice
  Pre-condition:          send-invoice may be executed when all of the following are true:
                            • individual is known
                            • totalClaim of individual must be more than own risk limit
  Boundary-condition:     -
  Post-condition:         invoice must be sent to address of individual
```

This is the '*send invoice to individual*' process as shown in fig. 9. We have assumed that an individual must have a client number, that is ***is known***; that `totalClaim` the attribute is where the total amount of his/her claims is stored; and that an `invoice` is sent to the address of the `individual`, which is stored in the `address` attribute.

This data perspective to processes modeling comes from claims-based working (Shao, 2010), which was used to create Stornebrink's (2010) framework. Shao (2010) took a data perspective to business activity access control, which is illustrated in fig. 15. A control mechanism is placed at the start of an activity and the criteria for granting access to the process is based on the data it requires. This is a declarative way of managing business activities. It does not prescribe "how" to produce the required data, only which data (the "what") is required in order to start the activity.

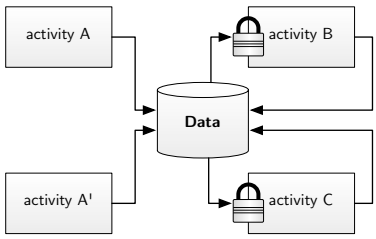


Figure 15. Data perspective (adapted from Stornebrink, 2010).

For example: Activity A produces the data which is needed as input for activity B. Within the control-flow mechanism only activity A can be performed to deliver the required data. The data perspective on the other hand enables a more flexible approach to deliver the required data, since it is not specified which activity should be performed. Therefore it is possible that activity A' will produce the data instead of activity A.

4.3.4. Conceptual model

We have now defined structure using *concepts* expressed in *terms* and their relationships expressed in *fact types*. We have defined power declaratively with rule-based BPM. We have defined three distinct categories

of control rules: (1) rejector rules; (2) producer rules; and (3) projector rules. Most of these rules have sub-categories which can be redivided. The classification of these rules is based on Ross' (2003) *BRS rule classification scheme*, which can be found in appendix B. These rules have been illustrated in a conceptual model in fig. 16. This model will serve as a reference to which kind of business rule to use in which situation in chapter 5.

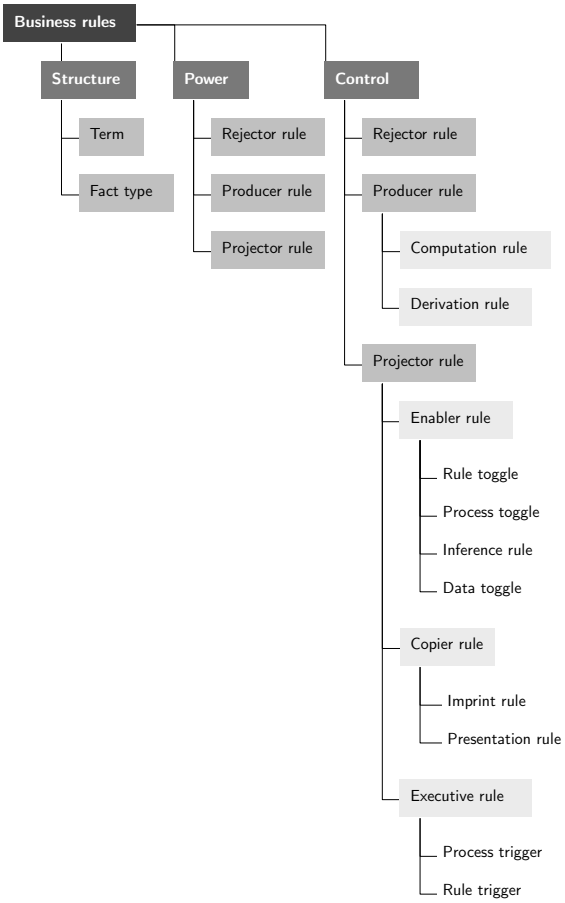


Figure 16. A conceptual model of business rules.

4.4. Rule modeling languages

Business rules are modeled in RMLs that offer a framework to represent business operations and constraints. A recent analysis on the completeness of process models and RML revealed that *Semantics of Business Vocabulary and Business Rules* (SBVR) to be the most complete based on maximum ontological completeness and minimum ontological overlap (Muehlen & Indulska, 2010). This conclusion is based on the Bunge-Wand-Weber[7] capabilities of four different RMLs: SMRL[8], SWRL[9], PRR[10] and SBVR. Within OMG's three-layer Model Driven Architecture (MDA), the SBVR specification fits in the Computation-Independent Model (CIM) (Linehan, 2008), which indicates that it is possible to transfer SBVR to any model in the layers below. SBVR describes concepts and constraints without specifying *how* they should be

[7] Well established framework for analysis and conceptualization of real world objects.

[8] Simple Rule Markup Language, for more information visit http://xml.coverpages.org/srml.html

[9] Semantic Web Rules Language, for more information visit http://www.w3.org/Submission/SWRL/

[10] Production Rule Representation, for more information visit http://www.omg.org/spec/PRR/1.0/

implemented. This allows for conceptual thinking instead of choosing a single solution for rule modeling. These are reasons for us to use SBVR in our case study. Fig. 17 illustrates how SBVR relates to the other two MDA layers.

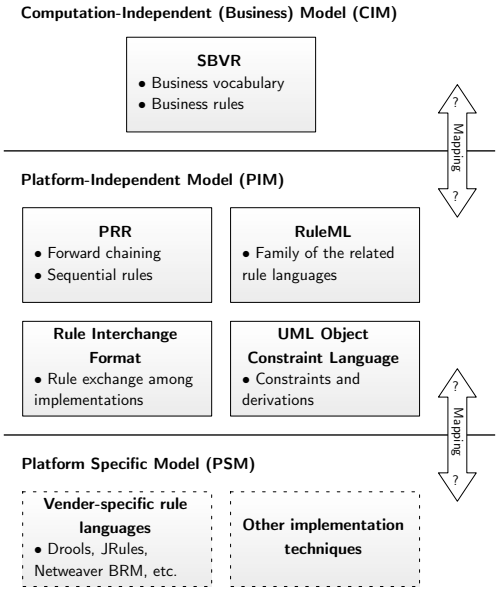


Figure 17. SBVR and Model Driven Architecture (adapted from Linehan, 2008).

4.4.1. SBVR

SBVR comes the closest to the definition of a standardized RML (Object Management Group, 2008) and it has much support[11]. It combines ideas from multiple topics, e.g. modeling, ontologies, mathematics, and linguistics. SBVR represents a vocabulary that is intended to become a standard upon which many grammars (i.e. languages) can be based (Linehan, 2008). A part of SBVR called 'Logical Formulations of Semantics' provides a structured vocabulary for describing *meaning* (Baisley, Hall & Chapin, 2005). SBVR is not an RML per se, but it comes with 'Structured English', an RML which can be used to capture business vocabulary and business rules.

Business vocabulary

A business vocabulary defines concepts (the equivalent of classes in UML), fact types (the equivalent of associations in UML), instances of concepts and fact types, and other concepts such as generalizations.

There are different kinds of fact types: (1) unary; (2) binary; (3) ternary or more; and (4) attributive fact types. Unary fact types are called characteristics of a single instance, e.g. `invoice` ***`is sent`***. Binary fact types are the most common ones we mentioned earlier, they define relationships between two concepts, e.g. `individual` ***`pays`*** `invoice`. *Ternary and larger fact types* are also possible, e.g. `institution` ***`sends`*** `claim` ***`of`*** `client`. Attributive fact types are equivalent of attributes in UML, e.g. `invoice` ***`has`*** `totalAmount`.

[11] A list of supporters from Wikipedia: *Automated Reasoning Corporation, Business Rules Group, Fujitsu Ltd, Hewlett-Packard Company, InConcept, LibRT, KnowGravity Inc, Model Systems, Ness Technologies, Perpetual Data Systems, Sandia National Laboratories, The Rule Markup Initiative, and X-Change Technologies Group.*

Concepts

Concepts can have definitions that describe derivations of the concept, e.g. grandfather *of* person might be defined as a parent *of* a parent *of* the person (Linehan, 2008). Concepts also have constraints, e.g. the fact type invoice *has* totalAmount, might have an additional (necessity) rule stating that each invoice *has* exactly one totalAmount. Concepts can have synonyms as well, e.g. claim might have a synonym called declaration, which means exactly the same thing.

Business rules

Business rules have been discussed thoroughly in this chapter. The following conceptual model illustrated in fig. 18, provides an overview of business rules in SBVR.

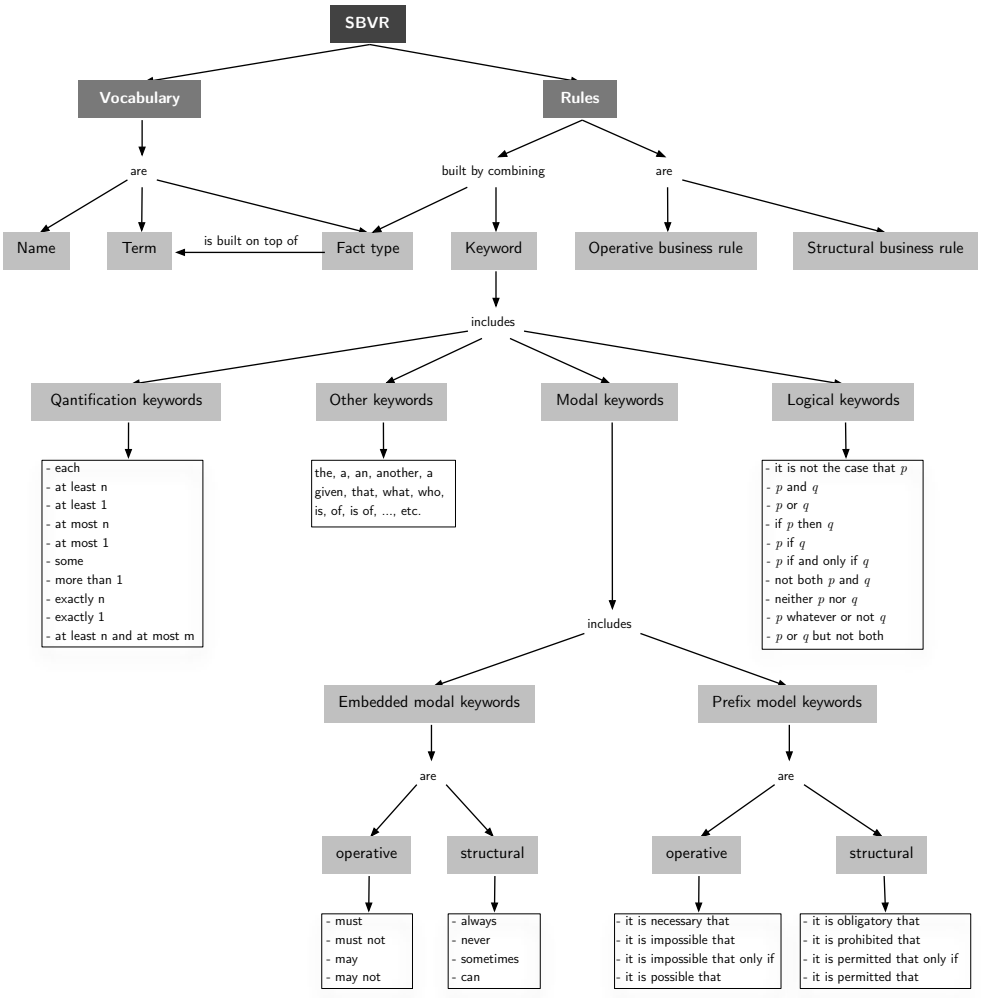


Figure 18. SBVR business rules — concept map (adapted from Musham, Singh, Bahal & Tv, 2008).

The business vocabularies and business rules form a conceptual schema, which then can be translated in to logical statements that can be interpreted by IT-systems — the SBVR specification is a MetaObject Facility- (MOF) and XML Metadata Interchange (XMI) based document format that can be used to exchange business vocabularies and business rules among SBVR tools (Linehan, 2008). The full SBVR XSD can be downloaded from OMG's website[12]. Tools specifically for SBVR are discussed in section 4.7.

[12] http://www.omg.org/spec/SBVR/1.0/

SBVR is not a syntax for rule modeling, but a highly structured set of fundamental concepts, i.e. it separates concepts and names. This approach enables multi-language support while keeping the same consistency. Multinational organizations can benefit greatly from this. This approach also ensures support for a variety of representational schemes (Ross, 2006a), such as RuleSpeak.

4.4.2. RuleSpeak

Developed by Ronald G. Ross, RuleSpeak is a well-documented RML developed by Business Rule Solutions, LLC (BRS) that has been used in practice since the 1990's (Ross, 2006a). The RuleSpeak scheme is based on SBVR and it is consistent with it (Ross, 2009a). RuleSpeak can use the SBVR constructs but embeds equivalent keywords, making it as compact and as focused as possible, which is crucial for communication among business people (Ross, 2006b). The difference lies in the use of keywords for 'modal operations' related to business rules, see table 2.

Modality claim type	**Statement form**	**SBVR structured English keywords**	**RuleSpeak keywords**
obligation claim	'obligative statement' form	it is obligatory that *p*	*r* must *s*
obligation claim embedding a logical negation	'prohibitive statement' form	it is prohibited that *p*	*r* must not *s*
	'restricted permission statement' form	it is permitted that *p* only if *q*	*r* may *s* only *t*
permissibility claim	'unrestricted permission statement' form	it is permitted that *p*	*r* may *s* *r* need not *s*
necessity claim	'necessity statement' form	it is necessary that *p*	*r* always *s*
necessity claim embedding a logical negation	'impossibility statement' form	it is impossible that *p*	*r* never *s*
	'restricted possibility statement' form	it is impossible that *p* only if *q*	*r* can *s* only *t*
possibility claim	'unrestricted possibility statement' form	it is possible that *p*	*r* sometimes *s* *r* can *s*

Table 2. Modal keywords in RuleSpeak (adapted from Object Management Group, 2008).

RuleSpeak consists of a set of templates (i.e. rule sentence templates) to represent different types of rules in a structured way (Ross, 2009b; 2009a), which are provided in appendix C. Because it's based on SBVR, it can be used to represent business rules in languages other than English as well (e.g. Dutch, German, and Spanish).

We have chosen to do the experiment in chapter 6 in SBVR with RuleSpeak. SBVR (i.e. 'Structured English') has proven to be the most complete RML (Muehlen & Indulska, 2010) and RuleSpeak is a more accessible scheme (than 'Structured English') to business people because of the compact nature, active community and available material to work with (Ross, 2003; 2006a; 2006b; 2009a; 2009b; 2010).

4.5. Disadvantages of rule-based systems

No approach or method, has only advantages. Searching for 'disadvantages of business rules' and 'disadvantages of rule-based systems' on Google led us to some disadvantages which aren't mentioned in current literature. Often cited disadvantages are (Aldewereld, 2009; González & Dankel, 1993; *OSWEGO State University of New York, 2009*):

- Opacity — Combinations of rules are difficult to see, link between individual rules and overall strategy and lack of hierarchy.
- Infinite chaining — A term of fact type can refer to another term, which can refer to another term.
- Conflicting new or updated rules — Adding new rules can lead to conflicts with existing rules.
- Ineffective search — Exhaustive search is slow if there is a large rule set, and it is hard to apply a new rule strategy.
- Coverage of domain — Rules may not be able to describe the complete ontology of objects relevant to the problem, not applicable to all problems.
- Inability to learn — Rules cannot modify themselves.
- Slowness — Rules can be slow to execute.

These disadvantages remind us to be aware of the shortcoming of a business rule approach. However most of these rules are based on González & Dankel's (1993) book which might be outdated because business rules have evolved since then and they are still being developed with support from the likes of *SAP*, *IBM* and *JBoss*. Business rule management systems (BRMSs) have been created to address the problems of *opacity*, *conflicting rules*, *ineffective search* and *slowness*. Other problems like *infinite chaining* can be avoided by defining only that which is valuable in the business domain. In the next section we will take a look at a few BRMSs too see how they address these problems.

4.6. Business rules management

Von Halle & Goldberg (2006) showed with their survey that the starting point for the process of rule discovery are existing use cases, process models, documents and even program code (Halle & Goldberg, 2006: p.23-24). Sources for business rules are people, code and documents (Halle & Goldberg, 2006: p. 24). Business rules are scattered and need to organized, that's where business rules management (BRM) comes into play. It describes the identification, definition and management of rules using BRMSs (Muehlen & Indulska, 2010). A BRMS is used to define, deploy, execute, monitor and maintain decision logic (i.e. business rules). A few known BRMSs are: *SAP NetWeaver BRM*, *IBM WebSphere JRules*, *JBoss Enterprise BRMS*, *JBoss Drools Guvnor*, and *RuleArts RuleXpress*. BRMS have the ability to express decision logic, using a business vocabulary and (graphical) RML (e.g. decision tables, trees, scorecards and flows). Typically, a BRMS includes the following:

- A repository, allowing decision logic to be separated from application code.
- Tools, allowing both IT people and business people to define and manage decision logic.
- A runtime environment, allowing systems to invoke decision logic and execute it using a rule engine.
- Consistency and completeness checks

Most BRMS vendors have evolved from rule engine vendors to providers of complete software development lifecycle solutions. We won't go into detail of all of these BRMSs, but we will highlight the JBoss Drools suite. Note that the JBoss Drools suite does not support SBVR (yet), but it works with the same principle of business rules expressed in concepts (terms) and fact types (relationships).

Drools Guvnor is the BRMS of the JBoss Drools suite which offers centralized knowledge repository where the so called Drools Knowledge Bases (DKB) are stored. A DKB can contain rules, models, functions, processes, etc. These knowledge bases can be managed though user defined categories, see fig. 19 and fig. 20. The guided editor provides a set-by-step way of creating and editing rules through a graphical interface, see fig. 21. Categories can also be use to control visibility of items (and hide features from users that don't need access to them).

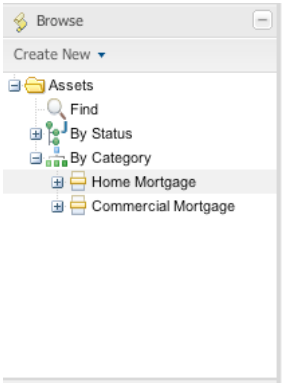


Figure 19. Drools Guvnor browsing (JBoss, 2011).

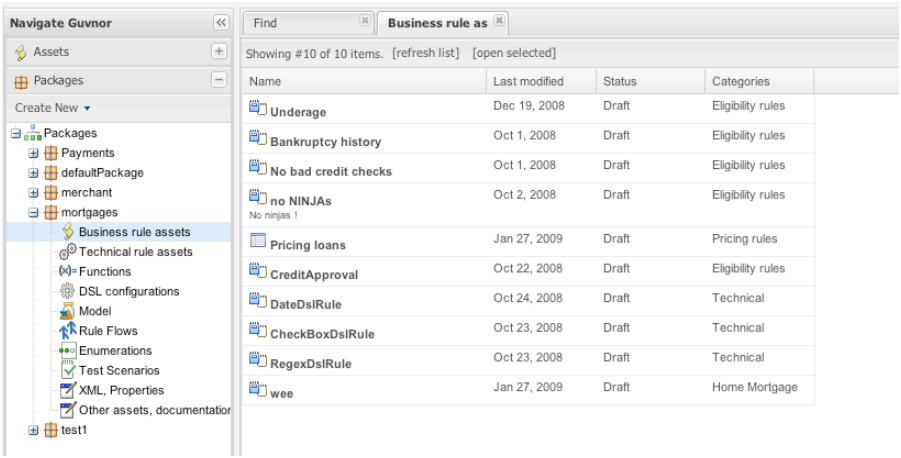


Figure 20. Drools Guvnor business rules navigation (JBoss, 2011).

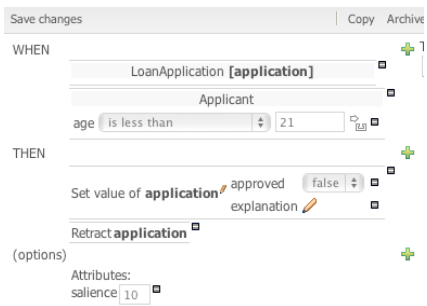


Figure 21. Drools Guvnor rules editor GUI (JBoss, 2011).

Rules can also be modeled using decision tables, see fig. 22. This Corticon BRMS shows the process of automated validation checking. In the example of fig. 22, the scenarios for 'Applicants' that are *less than 35* or who are skydivers are defined. Automatic rule checking shows that a scenario is missing for 'Applicants' that are *35 and older*, see fig. 23. Various consistency and completeness checks are provided by BRMSs. These features address the problems mentioned in section 4.5.

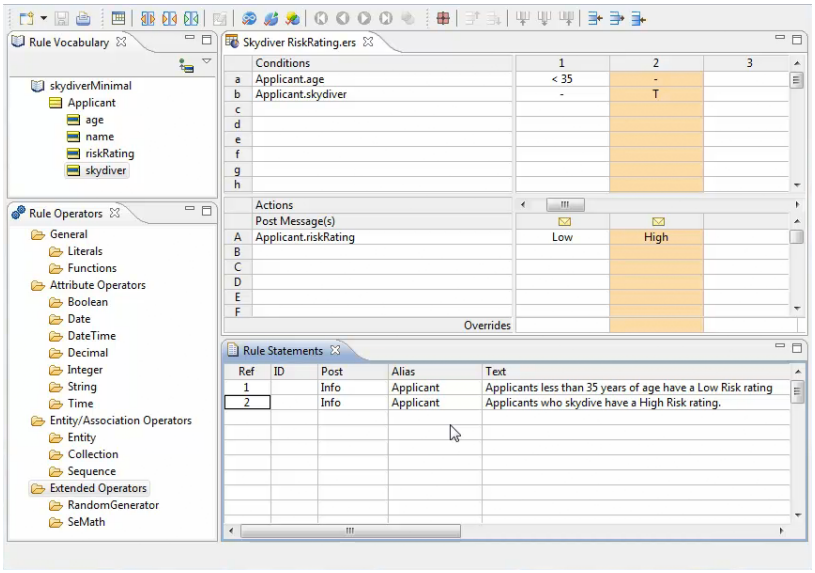


Figure 22. Corticon's "no-coding" business rule development (Corticon, 2011).

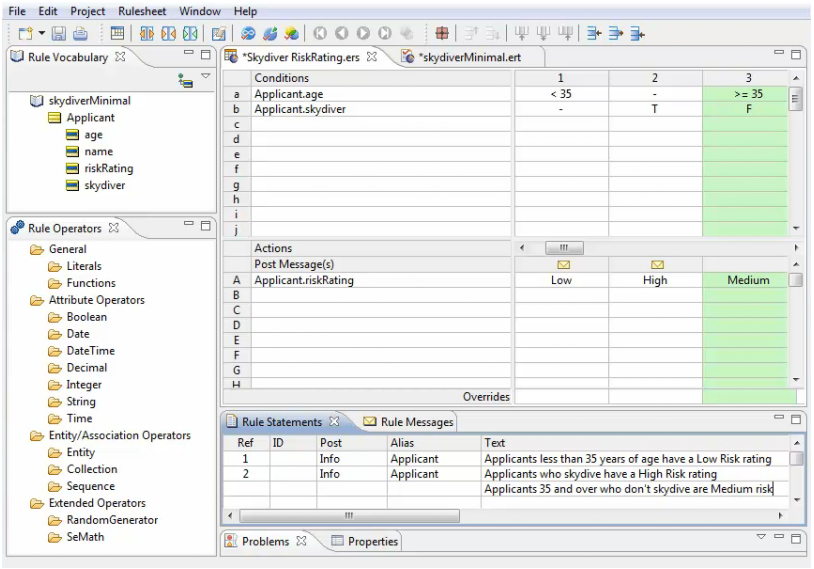


Figure 23. Corticon's integrity mechanism (Corticon, 2011).

4.7. SBVR-specific tools

The BRMSs we mentioned in section 4.6 are on the PIM/PSM level in the MDA (see fig. 17) and none of them has implemented SBVR (yet). This means that there is no proper development environment for

SBVR business vocabularies and rules. The SBeaVeR Eclipse project[13] can be used to model business vocabularies and business rules and save them in XMI the format, however this project has not been updated since 2006 and the plugin is buggy, which prevents proper usage. However this absence of tools is likely to change through the use of *ATL Transformation Language* (ATL). ATL is an extension to the popular *Eclipse* Integrated Development Environment (IDE) which is a model transformation language and a toolkit that can produce a set of target models from a set of source models. An SBVR component for Eclipse is being actively developed[14] as a part of the *Eclipse* Model Development Tools (MDT) which should make SBVR transitions to a variety of models available. There have been experiments with model → model (M2M) transformations, e.g. UML → SBVR (Pau, Cabot & Raventós, 2009), Syntax → SBVR → UML (Kleiner, 2009; Kleiner, Albert & Bézivin, 2009), SBVR → UML (Raj, Prabhakar & Hendryx, 2008) and model → text (M2T) transformations, e.g. SBVR → Natural Language (Bajwa, Lee & Bordbar, 2011). However text → model (T2M) transformations for SBVR still haven't been developed, the SBVR component for Eclipse should change that.

Since the SBVR specification is a MOF- and XMI based document format on CIM level (see fig. 17), it is possible to create these M2M, M2T and even T2M transformations. A proof of concept use case was presented by Mathias Kleiner (2009) that showed an M2M transformation from Syntax → UML with SBVR as intermediary. The transformation is done with ATL[15]. Syntax is an originally proposed meta-model for capturing syntax, grammatical dependencies and semantics of English sentences. The following figures represent the transformation of the the following business rule:

```
Each company sells at least one product
```

The transformation into the Syntax model is done manually. Syntax 'categories' can also express SBVR semantics through the "expresses" optional association. First, our example business rule is expressed in Syntax, see fig. 25. This model is then translated to SBVR in fig. 28. The SBVR model then can be mapped to UML, which can be seen in fig. 29.

[13] For more information, see http://sbeaver.sourceforge.net/.

[14] For more information, visit http://wiki.eclipse.org/MDT-SBVR-Proposal.

[15] The project files can be found at http://www.eclipse.org/m2m/atl/usecases/Syntax2SBVR2UML/.

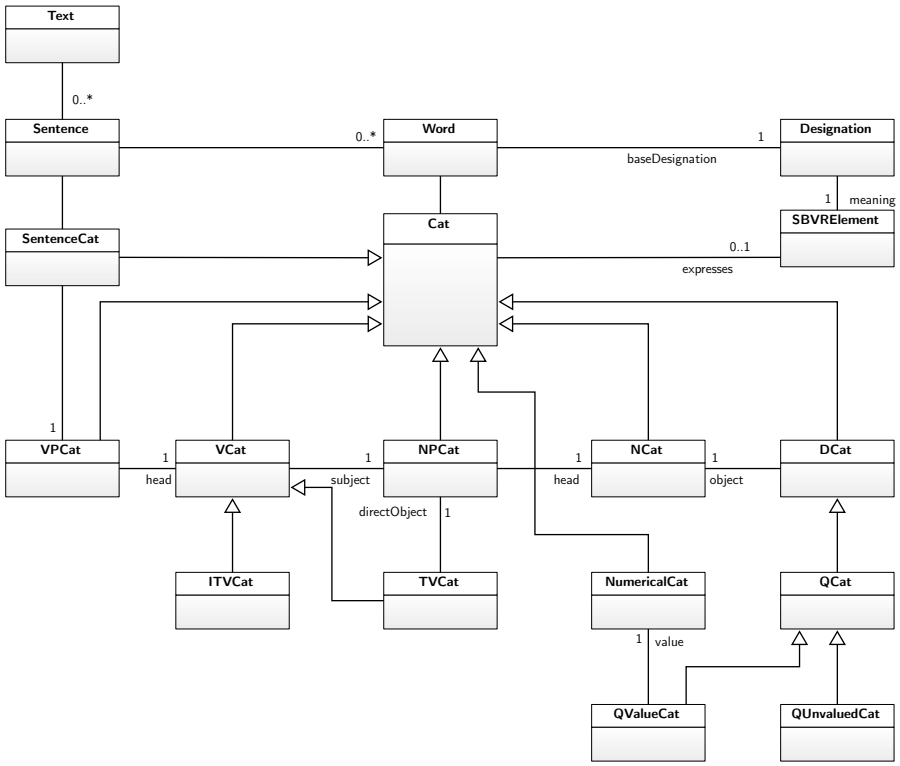


Figure 24. Excerpt of the Syntax meta-model (adapted from Kleiner, 2009a).

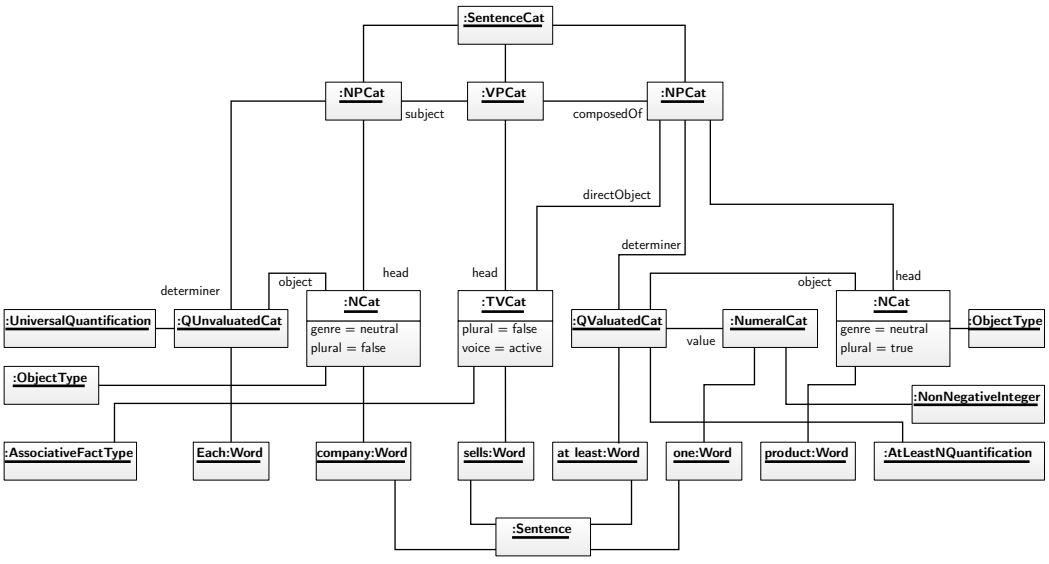


Figure 25. Fragment of a Syntax model for our business rule (adapted from Kleiner, 2009a).

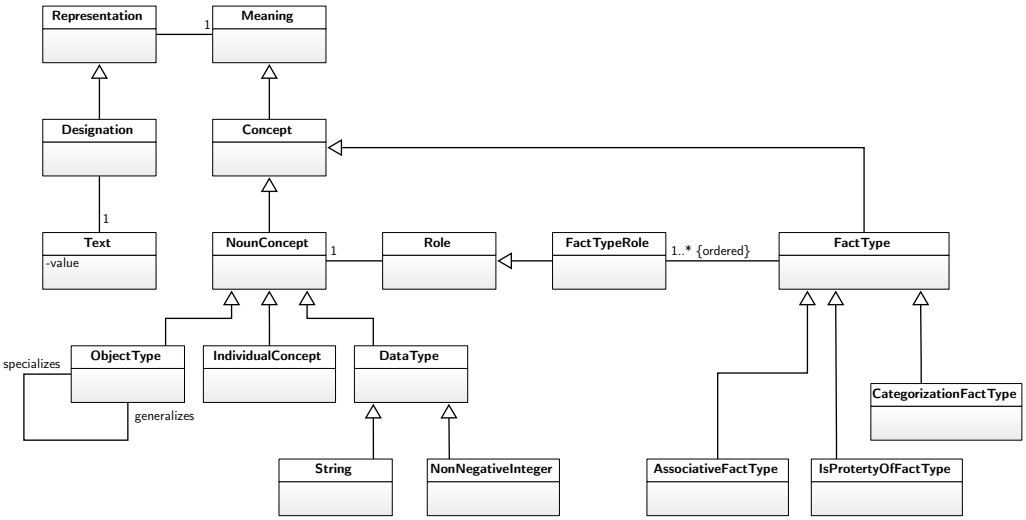


Figure 26. Excerpt of the SBVR meta-model: meanings (adapted from Kleiner, 2009a).

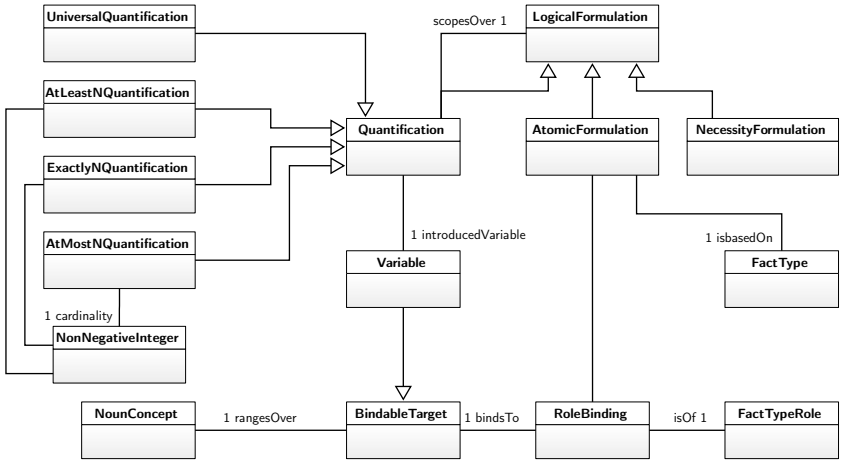


Figure 27. Excerpt of the SBVR meta-model: logical formulations (adapted from Kleiner, 2009a).

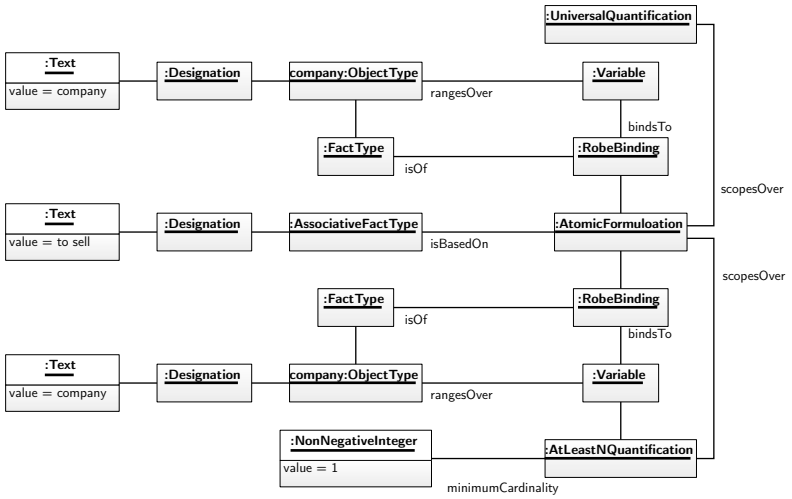


Figure 28. Fragment of a SBVR model for our business rule (adapted from Kleiner, 2009a).

Figure 29. Fragment of a SBVR model for our business rule (adapted from Kleiner, 2009a).

Note that this is a proof of concept, the end result is not fully UML compliant — it is a simplified version. The XMI files of these models (Syntax, SBVR and UML) can be found in appendix D. Even though this transformation shows that T2M transformations are possible, these tools are not (yet) usable for this research because they either don't work properly or haven't been released yet. A more complete manual SBVR T2M transformation provided by OMG can be found in appendix E.

4.8. Summary

In this chapter we defined what business rules are and we described the business rules approach. By analyzing the categories and types of business rules, a conceptual model is produced, see fig. 16. The concepts of structure, power, and control have been explained. Rule-based BPM has been explained, which has the potential to replace traditional procedural process modeling in the future. These concepts will be used in chapter 5 to define a conceptual framework for rule-based semantic IS standards and our BRASD. We have chosen to use SBVR with RuleSpeak as an RML because of (1) the completeness of SBVR — it gives us the modeling concepts to define most, if not all business rules that can be encountered within organizations and it is a major step forward in the process of making business domain knowledge explicit and transferable (Linehan, 2008) and (2) and because of the compact nature of the notation, active community and available material of RuleSpeak, which prescribes an approach to arrive at complete and correct SBVR models, which SBVR lacks (Linehan, 2008). The transformation from SBVR 'Structured English' to XMI or UML will have to be done manually since there are no proper tools to do it yet, however Eclipse and JBoss are in the process of supporting SBVR.

5. Creating rule-based semantic IS standards

This chapter provides the answer to the main research question: *"What are the characteristics of a rule-based semantic IS standard that has advantages in relation with current technologies and how can it be created?"*. In this chapter we will define a conceptual framework which is based on guidelines and best practices for creating rule-based semantic IS standards. Furthermore we will provide a method called BRASD for applying the business rule approach to semantic IS standards. Section 5.1 will describe the framework. Section 5.2 describes our business rule approach — BRASD. This chapter concludes with a summary in section 5.3

5.1. Framework

Our framework is based on the principles of the business rule approach and best practices in rule development. These characteristics and principles will be summed up for the *contents* of semantic IS standards. The contents can be categorized in the *meta solution*, which describes the approaches selected as fundament for the design of the standard; *the conceptual solution*, which describes the design of the solution in descriptions and models; and *the technical solution*, which describes the design of the solution in technical artifacts. The meta solution describes the results of the analysis phase of standards development, the conceptual solution describes the results of the design phase of standards development, and the technical solution defines the communication aspects of the standard, which become relevant once the standard is ready for use. Fig. 30 depicts this relationship of the contents of semantic IS standards with the phase of standard development.

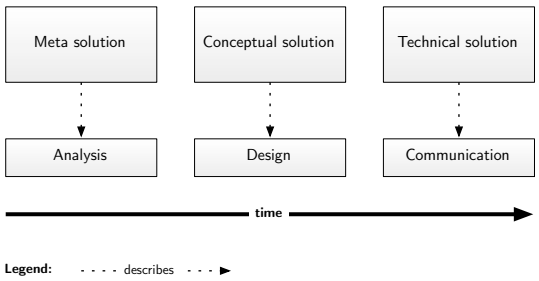


Figure 30. Relation of contents with standards development phases.

Characteristics and principles are numbered in arabic numerals: *(1)*, *(2)*, ..., *(n)* throughout the text.

5.1.1. Meta Solution

Paradigm (approach)

In the paradigm, the developer explains on a high level how the standard was designed. For a rule-based semantic IS standard, the developer can explain what kind of business rule approach (e.g. BRASD) is used to create the standard. The explanation can be a conceptual framework that explains, either graphically or in narrative form, the main things that can be found in the standard — the key factors, constructs or variables and the relationships among them (Huberman & Miles, 1994). However, including a conceptual framework in the standard is not a common practice to describe the approach to the standard design.

Methods/Languages

The selection of methods and languages used in the design phase are described here. For a rule-based semantic IS standard, a business vocabulary, structural rules and operative rules need to be defined. We suggest using BRASD to define structural and operative rules. For a rule-based semantic IS standard, a developer has to choose a rule modeling language that (at least) allows for rule expression in plain language (principle #2 of the business rule approach). We suggest using OMG's SBVR standard because it is a computation-independent model which allows for a model-to-model transformation. SBVR can also be expressed in different rule modeling languages (e.g. Structured English and RuleSpeak) and different natural languages (e.g. English, Dutch, German, Spanish, etc.). These characteristics allow for understandability by both business- and IT people, and for transformation to other models such as UML. Last but not least, all rules should be defined declaratively, meaning that only *what* must happen is specified, not *how* it must happen.

Architecture

In the architecture, the developer motivates and defines architectural design choices of the standard, including functional and technical architecture and relationships with other standards. For rule-based semantic IS standards, the architecture is clear in that there is a business vocabulary which is the source of all rules. This is important as a business rule should to be single-sourced if possible (principle #8 of the business rule approach).

For the meta solution of semantic IS standards we now can define the following characteristics:

(1) A business rule approach (e.g. BRASD) should be used to create the standard

(2) A rule modeling language based on SBVR (e.g. Structured English) should be used to create the contents of the standard

(3) A business rule should be declarative (i.e. only 'what' must happen is specified, not 'how' it must happen)

(4) A business vocabulary, structural rules, and operative rules should be included in the standard

(5) A business rule should be single-sourced if possible (i.e. the contents come from one business vocabulary)

5.1.2. Conceptual solution

The conceptual solution is considered to be the core of semantic IS standards — it explains *how* the standard works and *what* needs to be implemented.

Domain model (requirements)

For a rule-based semantic IS standards, the domain requirements are no different than the domain requirements for a regular standard. However the way they are defined might differ as requirements are essentially business rules. If the requirements need automation or concern an IT-system, then they have to be defined in a rule modeling language. These rules can be both structural and operative. For templates of operative (control) rules, see appendix C. Structural rules can looks like `resource must have one user`. General requirements concerning the organization or laws that don't need to be enforced with an IT-system can be written down in natural language.

Constraints

Business rules are naturally constraints (Folmer et al., 2011) and all constraints must be modeled with a rule modeling language. Based on the characteristics of constraints, they must be defined as rejector type rules or permission statements. Rejector type rules reject any event that causes the rule to be violated. Rejectors shield the business from incorrect data (or incorrect state). Permission statement allow or disallow a certain course of action. See appendix C, table 13 for the RuleSpeak template for rejector rules and table 14 for the permission statements.

Process

Process in a semantic IS standard can refer to process models, activity diagrams, and sequence diagrams. To truly benefit from a business rule approach, processes need to be defined declaratively with business rules as well. A process consists of one pre-condition, which defines the condition upon which it may start; one post-condition, which defines the condition upon it may terminate; and any number of boundary-conditions, each of which need to be satisfied within the process at any time. The pre-condition is always expressed with an enabler rule or an executive rule. The boundary-condition(s) can be expressed with any kind of control rule (i.e. rejector rules, producer rules, and projector rules). The post-condition is always expressed with a rejector rule. For example: We have a process called *process* and it may be executed when *data* is available, and it may terminate when the *value of attribute* is set to *null*. Its declarative rule definition in RuleSpeak would be as follows:

```
Pre-condition:       process may be executed when data is available
Boundary-condition:  -
Post-condition:      value of attribute must be set to null
```

Note that the pre-condition is a combination of a permission statement and a process trigger, see tables 14 and 23 in appendix C for the RuleSpeak templates. The post-condition is a combination of a rejector rule and an imprint rule, see table 21 in appendix C for the RuleSpeak template. Other types of control rules that can be used in processes are rule toggle, see table 18; process toggle, see table 19; and rule trigger, see table 24.

Rule based processes can be decomposed by means of sub-processes. Sub-processes are built on a single atomic (imperative) action, e.g. "send the form", see fig. 13 in sub-section 4.3.2. A subprocess is *anti-symmetric* and *irreflexive*. An irreflexive process cannot be a subprocess of itself and anti-symmetry prohibits child objects to be linked back to their parent by the same relation. Furthermore, a process is considered a method belonging to a concept.

Data/Information

Data/information refers to the static information within the standard (e.g. messages/documents, ontologies, code lists, taxonomies, data dictionary, sharable data components, etc.). For a rule-based semantic IS standard, only data or information that can be expressed with business rules is relevant. Which basically means, all data that is used for a transaction or execution of a rule, must be modeled as a business rule. Data structure (from a database perspective) can naturally be modeled with concepts (which are defined in terms) and fact types (the relationships concepts have with each other). Concepts are the equivalent of classes in UML and fact types are the equivalent of associations in UML. Concepts and fact types are used to create fact models (which is the equivalent of an UML class diagram). A fact model is considered a high level data model, but that is not its purpose (Ross, 2003), however a good fact model can be used as a data model. Creating a good fact model means capturing business knowledge from the business-side workers and managers who possess it (principle #9 of the business rule approach). Business

rules can also express data dependencies based on the process status (Folmer et al., 2011). Data can be expressed with one or more of the following types of rules:

- Rejector rules — see appendix C, table 13 for the RuleSpeak template.
- Permission statements — see appendix C, table 14 for the RuleSpeak template.
- Computation rules — see appendix C, table 15 for the RuleSpeak template.
- Derivation rules — see appendix C, table 16 for the RuleSpeak template.
- Inference rules — see appendix C, table 17 for the RuleSpeak template.
- Data toggles — see appendix C, table 20 for the RuleSpeak template.
- Imprint rules — see appendix C, table 21 for the RuleSpeak template.
- Presentation rules — see appendix C, table 22 for the RuleSpeak template.

For the conceptual solution of semantic IS standards we now can define the following characteristics:

(6) A requirement that needs automation, should be defined as a business rule

(7) A constraint should be defined as a rejector type rule

(8) A process must be modeled declaratively with business rules

(9) A process can only start if the pre-condition rule is met

(10) All work done within a process is aimed at fulfilling the post-condition rule

(11) All work done within a process is constrained by boundary condition rules

(12) Complex processes can be decomposed into more basic processes by means of subprocess

(13) Subprocess is anti-symmetric and irreflexive

(14) The post condition rule of a subprocess process must contribute to the post condition rule or boundary condition rule of its parent process

(15) Every process shall maintain every constraining boundary condition rule of its parent processes

(16) A process is defined as a method to operate on instances of a concept

(17) An atomic activity is the way of executing a process

(18) Every process that is not decomposed into sub processes must be assigned an atomic activity

5.1.3. Technical solution

The technical solution describes the design of the standard in a more detailed level of abstraction in terms of technical artifacts or deliverables. The format concept describes the format of the technical solutions in which the conceptual solution are represented. The medium represents the communication aspects of a standard.

Format

All the data/information mentioned in the conceptual solution of a semantic IS standard, must be modeled in a rule modeling language such as RuleSpeak. Business vocabularies and business rules should be written in plain language (principle #2 of the business rule approach). SBVR uses a certain layout/format to express the business rules, see tables 3 and 4. This format allows for understandability by both business people and IT people. However to be usable in IT systems, the rules should be available in an interchangeable data format such as XML as well. See appendix E for the XML version of a business rule.

```
human resource
```

Definition:	a party that *works* for another staffing company
Concept Type:	object type
General Concept:	party
Necessity:	staffing company must *be known*
Note:	Original definition: *The person performing the activities that are reported on the Timecard.*

Table 3. SBVR rule template - Concept.

agent *processes* claim	
Definition:	the agent checks if the sender *of* claim *is* a known client
Synonymous Form:	agent *checks* claim
Concept Type:	associative fact type

Table 4. SBVR rule template - Fact type.

The XML document in appendix E is in fact an XMI document, which enables model-to-model transformations. For example: An SBVR business vocabulary can be transformed to an UML class diagram, as shown in section 4.7. Currently there is no (truly working) commercial or open-source tools to model these rules in natural language and save them as XML. SBeaVeR is an Eclipse plugin to define business rules in Structured English and save them in XMI, for more information see http://sbeaver.sourceforge.net/. With the Eclipse IDE extension ATL, it will be possible to define business vocabularies and business rules in SBVR compliant languages such as Structured English and RuleSpeak in the near future. In the meanwhile this SBVR → XML transformation will have to be done manually. Both section 4.7 and appendix E provide examples how to do this.

Medium (transport)

The transport of rule-based semantic IS standards is no different from the regular semantic IS standards. However, the amount of business rules makes an offline (i.e. printed) medium impractical, and copying them (i.e. typing them over) is error prone. We suggest making available the XML version of the business vocabulary and business rules in the formats XMI and XSD.

For the technical solution of semantic IS standards we now can define the following characteristics:

(19) Business vocabularies and business rules should be written in a pre-defined template in a rule modeling language, such as the SBVR rule template

(20) Business vocabularies and business rules should be modeled in an Integrated Development Environment

(21) Business vocabularies and business rules should be documented in a well-formed and complete XML document format such as XMI or XSD

5.1.4. Conceptual model

Fig. 31 shows how the contents of semantic IS standards relate to the approach (BRASD), rule modeling language (SBVR), and the document format (XML). A dashed line indicates what kind of solution best to use with the semantic IS standards concepts.

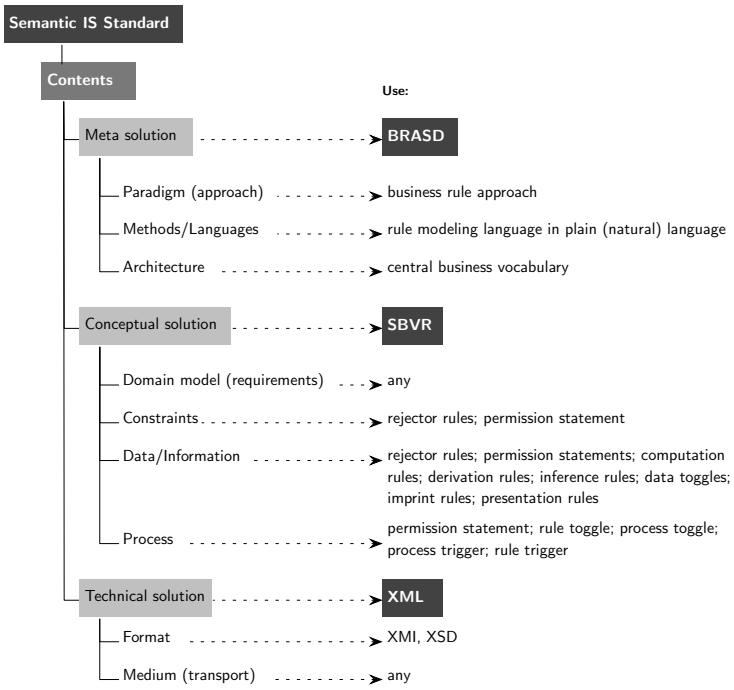


Figure 31. Relational concept model of semantic IS standard vs. BRASD, SBVR and XML.

5.2. BRASD

In this section we describe our business rule approach to semantic IS standards development (BRASD). To explain the principles of a business rule approach and the artifacts created, we will use the SBVR standard with RuleSpeak as rule modeling language. However BRASD can be applied with other rule modeling languages as well. The following typographical convention will be used to define different aspects of business rules. The rules will be depicted in RuleSpeak.

- `terms` are underlined
- ***`fact types`*** are given in italics
- **`literal values`** and **`instance names`** are shown with double underlines
- **`keywords`** are shown in bold font
- uninterpreted text is shown in normal font style

BRASD is a method that structurally guides standards developers to create a rule-based semantic IS standard. Our framework (section 5.1) should be used to guide the development process. The end result is the creation of five artifacts:

a. Business vocabulary

b. Structural rules

c. Fact model

d. Process rules

e. Operative rules

Before explaining these artifacts and their function, we will briefly discuss principle #4 of the business rule approach: *"Rules should build on facts, and facts should build on concepts as represented by terms"*. SBVR-based business vocabularies define (noun) concepts (the equivalent of UML classes), fact types (the

equivalent of UML associations), individual instances of both concepts and fact types, as well as various specialized concepts such as categorizations. For example: customer *places* order. The term 'customer' represents the concept of customers and 'order' represents the concepts of orders. The fact type 'places' represents the relationship between the two concepts. The UML version is represented in fig. 32.

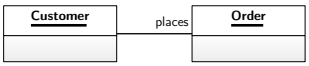


Figure 32. UML representation of our example.

Concepts may have definitions given as rules that describe derivations for the concepts. For example: grandfather *of* person might be defined as `a` parent *of* `a` parent *of* `the` person.

Our first artifact, the business vocabulary, contains the definition of concepts (terms) and names. The second artifact is a set of structural rules. Structural rules define the relationship between concepts (fact types) and their properties (i.e. attributes), for example: order *has* `exactly` one order number. The third artifact, the fact model serves as a tool to create overview of the different concepts and their relationships, see fig. 10 in sub-section 4.3.1. Note that an UML class diagram (with all the relationships defined) can be used as well. The fourth artifact, process rules, are rules that declaratively described pre-, boundary-, and post-conditions for a process. These rules can be used to create a rule-based process flow. The final artifact, operative rules, define what is allowed and what is not.

In the next sub-sections we will explain the process of creating a rule-based semantic IS standard by defining a business vocabulary, structure, power and control, fig. 33 represents our method. Before we can create our artifacts, we first need to select a rule modeling language. The actors involved in the whole process are stakeholders (people who the standard is created for) and developers (people who develop the rule-based semantic IS standard). The developer consults with the stakeholder(s) about which rule modeling language to use. The stakeholder is the one who is going to implement the standard, so it should be useable for the stakeholder. Depending on the available tools and stakeholder preferences, a rule modeling language is chosen. Once the developer and stakeholders come to an agreement, we can proceed with defining the business vocabulary.

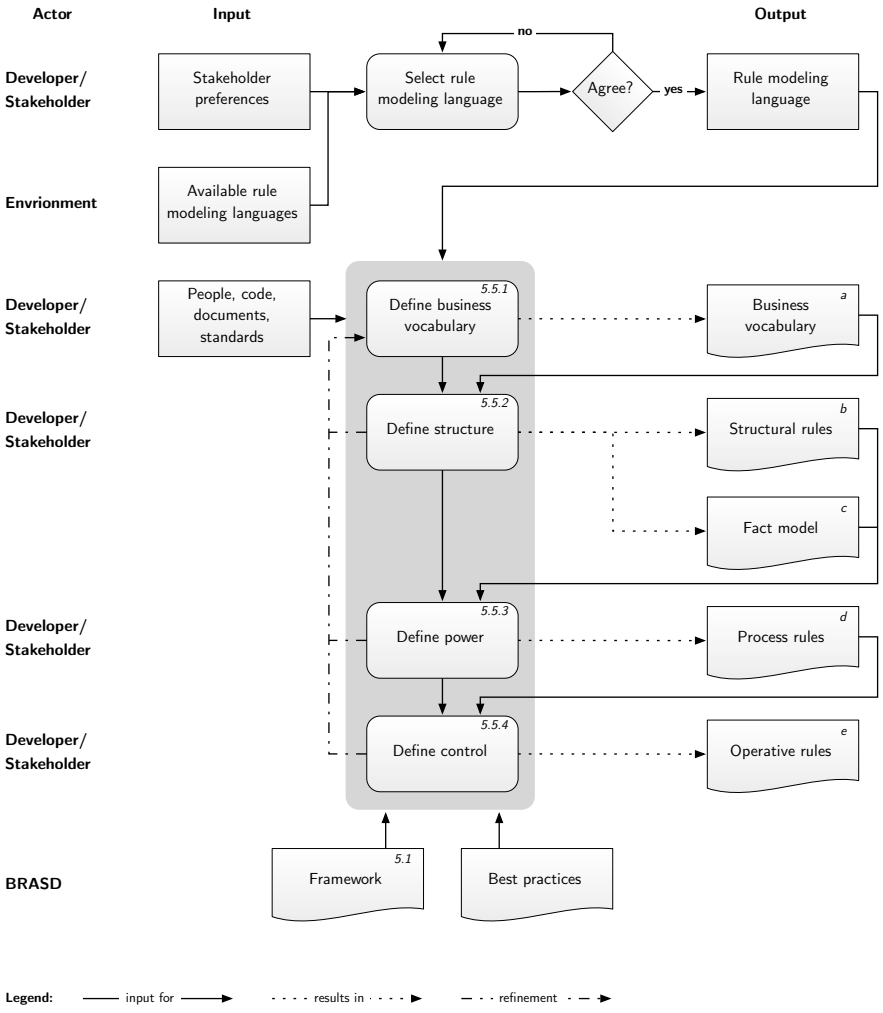


Figure 33. Business rule approach process.

5.2.1. Define business vocabulary

Typical sources for business rules are people, code and documents (Halle & Goldberg, 2006). To create a business vocabulary, concepts need to be defined with terms, relationships between concepts need to defined with fact types, and names of individual objects need to be defined as well.

(1) Define concepts with terms.

Concepts that have value for the stakeholder should be defined in the business vocabulary in terms, just like classes in UML. For example, customers should be defined with the term `customer`.

(2) Define the names of individual objects.

The names of individual objects should be defined explicitly. For example, `McDonalds` is a name (double underlines), which can be used in the following (characteristic) rule: `restaurant` ***`having a McDrive`***. The definition of this rule could contain something like `a branch that` ***`is owned by`*** `McDonalds`, meaning that restaurant that have a *McDrive*, are owned by *McDonalds*.

Ross (2003) defines two core practices regarding definitions and constraints (i.e. business rules).

Every concept should be defined by its *essence definition*. This means that a definition of a concept should always focus on the core essence of the concept i.e. "on fundamental meaning that is unlikely to change" (Ross, 2006b). Language used in a common dictionary is strongly advised.

- ***Tip 1:*** *Definitions should focus on the core essence of a concept.*

All constraints should be expressed as rules and separated from definitions. The boundaries of a concept are defined by rules and since these concepts can change over time, so will their boundaries, therefore it is not advised to embed them into definitions (Ross, 2006b). However, the business concepts *can* be embedded in definitions. Experience in large-scale projects indicates that these core practices (Ross, 2006b) ensure good business communication, produce friendly and highly stable definitions, and scale extremely well for complex business problems featuring hundreds or thousands of rules.

- ***Tip 2:*** *Constraints should be expressed as rules and separated from definitions.*

Once all the concepts are defined in a business vocabulary, the structure can be defined with structural rules.

5.2.2. Define structure

(1) Create a fact model.

A fact model or UML class diagram helps to see how concepts relate to each other and it allows for keeping overview of the "business landscape". Both models should always include every relationship and modality constraints between concepts to discover missing or incorrect relationships. It is not compulsory to create a model, but it is strongly advised.

(2) Define relationships between concepts with fact types.

Concepts have relationships with other concepts. These relationships should be modeled conform the rule modeling language. For SBVR these relationships are called fact types. There are associative-, is-property-of-, and categorization fact types, see the SBVR standard for more information. For example, the rule `customer has customerID` is an is-property-of fact type, meaning that the term customer has a customerID attribute. The rule `customer places order` is an associative fact type, meaning that customers place orders, and that orders are placed by customers.

(3) Define structural rules.

Using the fact or UML model, we can define structural rules. Structural rules define boundaries for a concept. For example, in RuleSpeak a structural rule for customer could be `a customer can have at most two orders with status of order in progress`. A different example shows multiple rules applied at once:

`All of the following are always true for a customer:`

- `It has a customerID`
- `It can have at most two orders with status of order in progress`

(4) Define structural rules derived from terms.

Structural rules can also derive from terms. For example, we have defined that a McDonalds branch can have a third-party location, this would mean that we have to define what a `third-party location` is. It can be expressed with the following rule:

A location is to be considered a third-party location if *located at* a McDonalds site that *is owned by* a third-party.

- ***Tip 3:*** *Abstraction in definition should go as far as it has value for the stakeholders of the standard.*

(5) Define binary fact types.

Concepts have attributes. These can be defined with binary fact types. For example, customer *has* customerID, customer *has* name, etc. Attributes often have synonymous forms such as customerId *of* customer, name *of* customer, etc.

(6) Define unary fact types.

Properties of attributes can be modeled with unary fact types. For example, customerID *is empty*. This characteristic of customer can be defined as customerID *is empty* when customerID *of* customer = null.

All the other non-derived terms should be defined in the business vocabulary with their core essence.

- ***Tip 4:*** *Ensure that all non-derived terms have essence definitions.*

Once we have a structure, we can start defining the processes that act upon the structure, forming power.

5.2.3. Define power

Processes are what deliver "power" in a business, they make things move. Most process models are procedural, meaning that they describe *when* and *how* something must happen. These processes can be modeled with (Event-)Condition-Action (ECA) rules. Procedural process models can lead to manual redesign which can be a lengthy an error prone process if a model is used across multiple standards. Processes can also be modeled declaratively with business rules if they define *what* must happen, instead of *how* it must happen. A declarative process always has one pre-condition, one post-condition and optionally boundary-conditions. The pre-condition is always expressed with an enabler rule or an executive rule. The boundary-condition(s) can be expressed with any kind of control rule (i.e. rejector rules, producer rules, and projector rules). The post-condition is always expressed with a rejector rule. The RuleSpeak templates for these rules can be found in appendix C.

For example: We have a process called *process* and it may be executed when *data* is available, and it may terminate when the *value of attribute* is set to *null*. Its declarative rule definition in RuleSpeak would be as follows:

```
Pre-condition:       process may be executed when data is available
Boundary-condition:  -
Post-condition:      value of attribute must be set to null
```

(1) Define the process pre-condition.

(2) Define the process boundary-conditions.

(3) Define the process post-condition.

Once all the processes are defined, we can create control with the control rules. These rules are used to create an artificial process flow (artificial because declarative processes are not procedural).

5.2.4. *Define control*

In this phase we define the operative rules, i.e. control rules and rules that create artificial flow. The following rules are operative rules: each order that has totalAmount of order more than 100 euros has no shipping costs or each rental car must be owned by exactly one branch. Appendix C contains RuleSpeak templates for control rules.

With projector rules we can create an artificial process flow, without defining how to get to the next process. For example, an order must be either accepted or rejected. Within an automated environment that has a rule engine, a user cannot proceed until a violation of a rule stops existing. In the example above, if an order is neither accepted or rejected, then that rule is violated and the user is requested to take action.

(1) Define operative rules (i.e. control rules).

(2) Generate artificial process flow using projector rules.

5.2.5. *Best practices*

We include 5 best practices when using RuleSpeak (Ross, 2006a). These might apply to other rule modeling languages as well.

- ***Tip 5:*** *should must not be used in place of must in expressing a business rule if the business rule has an enforcement level and the enforcement level of the business rule is inconsistent with the English sense of 'should'.*

- ***Tip 6:*** *may must be used in the sense of permitted to in RuleSpeak. may must not be used in the sense of might.*

- ***Tip 7:*** *An affirmation or admonition must not include a rule keyword.*

- ***Tip 8:*** *A statement expressing a rule or affirmation or admonition should not begin with a condition. 'Condition' as used here means a qualification set off by if, while, when, etc. (e.g., "If a rental is open...").*

- ***Tip 9:*** *A double negative should be avoided in expressing a rule.*

5.3. Summary

In this chapter we have defined a framework guiding standards developers by describing 21 characteristics of rule-based semantic IS standards. We have also presented an approach to systematically build the business vocabulary, structure (i.e. structural rules), power (i.e. processes) and control (i.e. control rules) of semantic IS standards called BRASD. BRASD contains best practices in the form of tips to ensure good business communication, friendly and highly stable definitions, and a scalable business vocabulary. The framework and BRASD can also be used to translate existing semantic IS standards to a rule-based version. In chapter 6, we will apply our framework and approach to a real-life semantic IS standard to a rule-based version.

One limitation of the BRASD approach is that the power aspect of the approach (defining declarative processes) is not usable in practice yet. To out knowledge there are no rule engines that can handle declarative business process modeling. Until these features are available we suggest using a process modeling notation such as BPMN. Fig. 34 summarized the BRASD approach.

BRASD

1. **Define business vocabulary**
   - *1.1. Define concepts with terms*
   - *1.2. Define the names of individual objects*
2. **Define structure**
   - *2.1. Create a fact model*
   - *2.2. Define relationships between concepts with fact types*
   - *2.3. Define structural rules*
   - *2.4. Define structural rules derived from terms*
   - *2.5. Define binary fact types*
   - *2.6. Define unary fact types*
3. **Define power**
   - *3.1. Define the process pre-condition*
   - *3.2. Define the process boundary-conditions*
   - *3.3. Define the process post-condition*
4. **Define control**
   - *4.1. Define operative rules (i.e. control rules)*
   - *4.2. Generate artificial process flow using projector rules*

- ***Tip 1:*** *Definitions should focus on the core essence of a concept.*
- ***Tip 2:*** *Constraints should be expressed as rules and separated from definitions.*
- ***Tip 5:*** `should` *must not be used in place of* `must` *in expressing a business rule if the business rule has an enforcement level and the enforcement level of the business rule is inconsistent with the English sense of 'should'.*
- ***Tip 6:*** `may` *must be used in the sense of* `permitted to` *in RuleSpeak.* `may` *must not be used in the sense of* `might`*.*
- ***Tip 7:*** *An affirmation or admonition must not include a rule keyword.*
- ***Tip 8:*** *A statement expressing a rule or affirmation or admonition should not begin with a condition. 'Condition' as used here means a qualification set off by* `if`*,* `while`*,* `when`*, etc. (e.g., "*`If` *a rental is open...").*
- ***Tip 9:*** *A double negative should be avoided in expressing a rule.*

Figure 34. BRASD.

6. Experiment — Rule-based SETU standard

In this chapter we will apply our framework and BRASD to a real-life semantic IS standard — the 'SETU Standard for Reporting Time & Expenses 1.1' ('SETU standard' hereafter). First we will give some background information about SETU and what they do, followed by an introduction to the SETU standard in section 6.1. Our BRASD approach is applied: a business vocabulary is defined in section 6.2; the structural rules are defined in section 6.3; the power rules are defined in section 6.4; the operative rules (i.e. control rules) are defined in section 6.5. Finally this chapter concludes with a summary in section 6.6. Please note that not all business rules are defined in this chapter, for a full rule-based version of the SETU standard, see appendices F, G, H and I.

6.1. SETU Standard for Reporting Time & Expenses 1.1

SETU is the Dutch acronym for 'Stichting Elektronische Transacties Uitzendbranche', which translates to "Foundation for Electronic Transactions in the Staffing Industry". SETU is a non-profit organization and SSO that develops and maintains standards for exchange of electronic information in the Dutch staffing industry. They also facilitate the standardization. Based on open standards such as HR-XML SIDES (HR-XML Consortium, 2007) and collaboration between staffing companies, staffing customers and scientific institutes, SETU has developed four semantic IS standards for the staffing industry in the Netherlands:

- Ordering & Selection of temporary personnel
- Assignment
- Reporting Time & Expenses
- Invoicing

These standards support typical staffing industry transactions like (adapted from Hofman, Holtkamp & van Bekkum, 2010):

1. Employers and staffing suppliers most often have framework contracts in which an employer submits an order for work with its particular requirements to a staffing supplier.
2. A staffing supplier checks its available workforce and a (number of) employee(s).
3. A selection is made and a (number of) employee(s) is (are) hired by the employer, this results in an actual assignment under certain conditions for a specific time.
4. An employee starts working on the agreed assignment.
5. An employee keeps track of the time worked on a certain assignment.
6. A staffing supplier invoices an employer for the work as fulfilled by the employee.
7. A staffing supplier pays an employee for the fulfilled work.

The entire process, from ordering and selection, to invoicing is executed with IT-support. Every standard comes with a set of models (domain and object models), diagrams (class and context diagrams), scenarios (activity, use case, sequence and state transition diagrams), message definitions, XSDs and business rules to describe the moments (when) and the content (what) of information exchange. These standards are based on the XSDs developed by the HR-XML Consortium, which were extended with local requirements of the Dutch staffing industry (Hofman et al., 2010).

For this case study our framework is applied to the SETU standard, which is targeted at the process of reporting time & expenses in the staffing industry. It deals with electronically sending invoicing

information, including corrections. This standard is intended for use within the domain of the Dutch staffing industry (Folmer et al., 2009).

The standard supports the transfer of information, from one sender (information registrar) to a receiver. The receiver may then proceed to process the information registered, e.g. for invoicing, payroll activities or for retransmission to another party.

In the next section we will discuss the business vocabulary, structure, power and control of the SETU standard. For every section, only a few examples are shown and explained. The full version can be found in the appendix F, G, H, and I. The activities in these sections are constraint by the framework in chapter 5.

6.2. Business vocabulary

We start by defining the business concepts in a business vocabulary. The vocabulary will give us an overview of business concepts (terms) and their relationships with each other (fact types). The RuleSpeak approach to this phase is the same as the SBVR approach (Object Management Group, 2008; Ross, 2006b). To define business concepts by their *core essence*, we will use a standard dictionary, e.g. the *Merriam-Webster Unabridged Dictionary* (MWUD)[16]. Other dictionaries or resources (e.g. Wikipedia) can be used as well.

Just like the other SETU standards, the 'SETU Standard for Reporting Time & Expenses 1.1' consists of a 'Model for Reporting Time & Expenses' and an XML mapping to HR-XML SIDES. The XML mapping is not used because the mapping of the SETU standard to HR-XML SIDES is not relevant for the objective of this research, which is to determine if a business rule approach to semantic IS standards development can contribute to cost-, quality-, and understandability-related problems. The 'model' is divided into three categories: (1) structure, (2) interaction & behavior; and (3) messages. For this phase we will predominantly use the first category — structure. It contains a context diagram and ER diagrams with the appropriate descriptions. Note that the definitions of the concepts are determined by the author, and don't necessarily apply in real-life situations in the staffing industry.

The context diagram in fig. 35 depicts the parties that can 'register information'. This is somewhat ambiguous as 'register information' is not mentioned anywhere in the text. The roles that these parties can take on are explained in table 5. For practical reasons, only a fragment of the business rules is shown in this chapter, the rest can be found in appendix F.

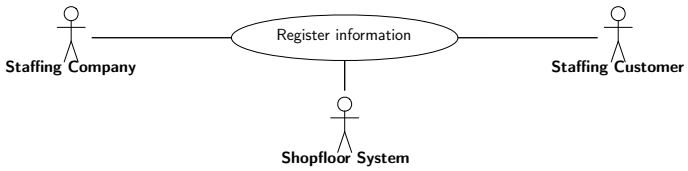


Figure 35. Parties and roles context model (adapted from Folmer et al., 2009).

[16] http://www.merriam-webster.com/

Party	Role	Description
Staffing company	Recorder	The staffing company can perform registration of time/expense information. A human resource may or may not register their information with systems in the organizational domain of the staffing company.
	Submitter	The staffing company may resubmit the information registered to another party for administrative tasks (payrolling, approval, etc).
	Receiver	The staffing company may act as a receiver for the information, when registered with another party.
Shop floor recording system	Recorder	Shop floor recording systems provide automated recording of presence information (time recording) on site with the staffing customer.
	Submitter	The recording system will typically also act as a submitter, providing input to the administrative processes of the organization it is associated with.
Staffing customer	Recorder	The staffing customer can perform registration of time/expense information. A human resource may or may not register their information with systems in the organizational domain of the staffing customer.
	Submitter	The staffing customer will submit the information to another party if the information is registered at its premises.
	Receiver	The staffing customer will typically authorize registered time/expense information before it is input to the invoicing process. In order to do so, it needs to act as a receiver if information is registered with another party.

Table 5. Parties and roles definitions (adapted from Folmer et al., 2009).

We see that there are three parties i.e. *staffing company*, *staffing customer*, and *shop floor recording system*. Note that this is the same as 'Shopfloor System' in fig. 35. A good practice in this case is to define the *core essence* of a party first.

party

Definition:	a person or group *participating* in a transaction
Concept Type:	object type
Dictionary Basis:	a person or group participating in an action or affair — *a mountain-climbing party, a party to the transaction.*

Party is a class, so it would be a noun concept of the category object type in SBVR terminology. Terms like *person*, *group* or *transaction* should be defined as well as they can have value for the business, however they don't serve a direct purpose in our case, these are not further defined. Now we know what a party is, we can define the different parties. The term 'staffing company' was not found in the MWUD. Searching for 'staffing agency' on Wikipedia did turn up a synonymous definition (i.e. employment agency).

staffing company

Definition:	a party that *matches* human resources to staffing customers
Concept Type:	object type
General Concept:	party
Dictionary basis:	An employment agency is an organization which matches employers to employees — *an advertising agency, an employment agency* (Wikipedia, 2011).

The *general concept* indicates that staffing company is generalized from party. The same goes for staffing customer and shop floor recording system. Note that from here on no dictionary will be used as this doesn't add value to the research process anymore.

staffing customer	
Definition:	a party that *hires* human resources from staffing companies
Concept Type:	object type
General Concept:	party

shop floor recording system	
Definition:	a party that *records* time information at the staffing consumer location
Concept Type:	object type
General Concept:	party

The terms time information and staffing customer location were a part of an iterative process as we went through the SETU standard from beginning to end and discovered new things along the way. The human resource (i.e. the fourth party) is not included in table 5 because he/she doesn't play a role in the exchange of information between the other three parties. However, it must be defined as the term is used in other terms, e.g. in timecardLine. Note that some business concepts have been defined differently to keep the business vocabulary compact and consistent (see the '**Note:**' row).

human resource	
Definition:	a party that *works* for another staffing company
Concept Type:	object type
General Concept:	party
Note:	Original definition: *The person performing the activities that are reported on the Timecard.*

Now we have defined the parties, we will further define the roles that these parties can have. The roles are *submitter*, *receiver*, and *recorder*. All of these roles are generalizations of the business concept party and they are noun concepts. Note that RuleSpeak keywords are **bold**.

submitter	
Definition:	a party that **can** *send* a timecard
Concept Type:	noun concept
General Concept:	party
Note:	Original definition: *The person, organization or business entity responsible for submitting the Timecard.*

The other roles can be found in appendix F. The 'Model for Reporting Time & Expenses' contains an UML domain model encapsulating various entities that are related to the reporting time & expenses processes, see fig. 36.

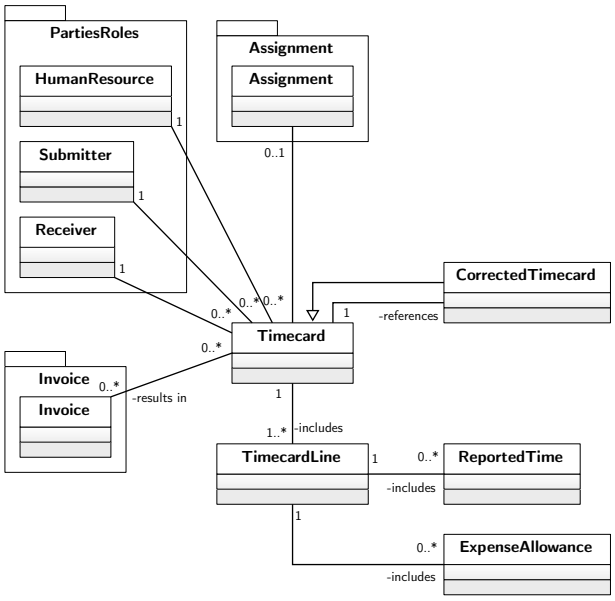


Figure 36. Reporting time & expenses domain model in UML (adapted from Folmer et al., 2009).

This model is supplied with a table of the relevant objects for this SETU standard, see table 6.

Object	Description
Timecard	The main object of this diagram. It is used to specify all time & expenses information relevant to activities as described in the Assignment.
CorrectionTimecard	The correction timecard is a specialization of the Timecard object. It serves the purpose of conveying information for a correction process.
TimecardLine	A container with reported time and/or expense data for a person or resource in a given period.
Invoice	The timecard induces the creation of an Invoice in the invoicing process, in order to execute a financial transaction for services rendered by the staffing company to its customer.
Assignment	The Assignment describes the details for the activities and rewards agreed upon between staffing company and staffing customer.
ReportedTime	A container used to report the duration of work for a specified time period.
ExpenseAllowance	Container for expenses incurred and/or allowances for a specific time period.
HumanResource	The person performing the activities that are reported on the Timecard.
Submitter	The person, organization or business entity responsible for submitting the Timecard.
Receiver	The person, organization or business entity responsible for receiving the Timecard

Table 6. Domain object (adapted from Folmer et al., 2009).

The business rules definitions for the objects that haven't been written yet, can be found below.

```
timecard
    Definition:             an object that contains timecardLines
    Concept Type:           object type
```

correctionTimecard	
Definition:	a timecard that *has* a status *of* timecard that *is* corrected
Concept Type:	timecard
Synonymous Form:	corrected timecard

The status 'corrected' is mentioned only in an activity diagram as a note. It is missing from the state transition diagram. The iterative process of defining the standard in business rules led to the usage of the 'corrected' status here. Note that corrected timecard is synonymous with correctionTimecard, a flexibility that SBVR has.

timecardLine	
Definition:	a container that can *have* reportedTime or expenseAllowance for a human resource for a specific period
Concept Type:	object type
Synonymous Form:	timecard line

The term 'container' is not defined in the SETU standard, but it can be viewed as a class, thus a noun concept of the category object type.

We have defined the business vocabulary for all the (relevant) terms and their relation with each other to some extent, an overview can be found in appendix F. These terms serve as the bones in our "skeleton" (i.e. fact model) of the SETU standard. The final fact model for the SETU standard is shown in fig. 37 to keep oversight. Note that the model is normally not complete at this stage because there are some *fact types* and *structure rules* that still need to be defined.

6.3. Structure

The next step is to define structural rules for the business concepts (i.e. terms) we have defined in the business vocabulary. Structural rules define *boundaries* for the business concepts (Ross, 2006b). The structural rules are defined from both the 'interaction & behavior' section as well as the 'structure' section of the SETU standard. Fig. 37 depicts the final fact model after defining the structure of the SETU standard. In this section we will explain how we came to this model.

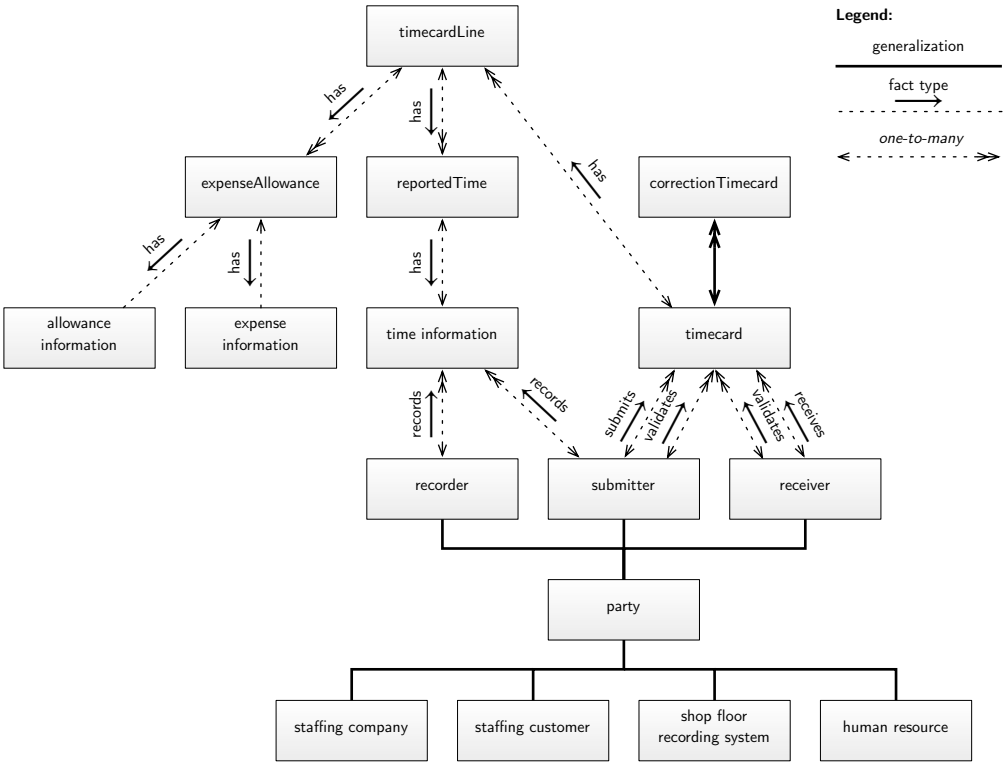


Figure 37. SETU standard fact model.

The fact model only contains *terms* with *binary fact types* (e.g. submitter can *submit* timecard) and *ternary fact types* (e.g. expenseAllowance can *have* allowance information and expense information). The fact model does not contain *attributes* (even though they are binary fact types, they don't add value to the model) and *unary fact types* (e.g. identifier *is unique*). The attributes can be found in the timecard object model, shown in fig. 38, their description can be found in appendix J.

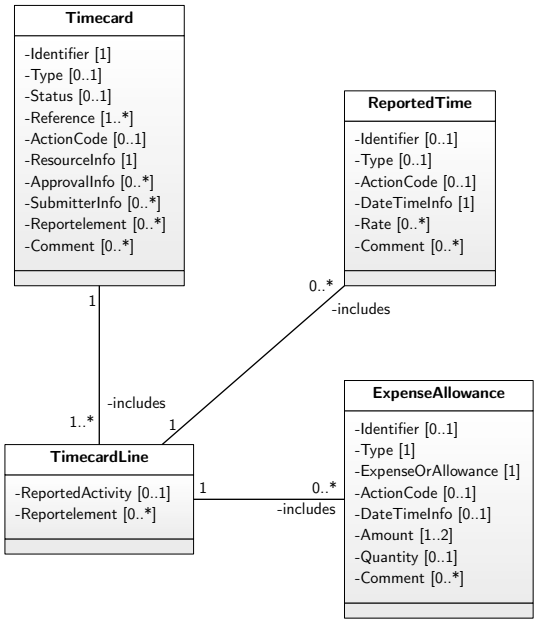


Figure 38. Timecard object model in UML (adapted from Folmer, Roes & van Bekkum, 2010).

These attributes and relationships in fig. 38 have cardinality constraints. The business rules formulation is shown in table 7.

Cardinality	RuleSpeak keywords
[1]	**must** *have* one
[1..*]	**must** *have*
[0..1]	**can** *have* **at most** one
[0..*]	**can** *have*
[1..2]	**must** *have* one or two

Table 7. Cardinalities in RuleSpeak.

Now we will detail the Timecard class with first its *attributes*, then the *structural rules* and then the *unary fact types* (characteristics). The other classes can be found in appendix G. Firstly, the attributes are summed up. Every attribute belongs to a timecard, these kinds of relations are considered is-property-of fact types. Cardinalities are defined as necessities.

identifier Concept Type: attribute	timecard *has* identifier Concept Type: is-property-of fact type Synonymous Form: identifier *of* timecard Necessity: timecard **must** *have* one identifier
type Concept Type: attribute	timecard *has* type Concept Type: is-property-of fact type Synonymous Form: type *of* timecard Necessity: timecard **can** *have* **at most** one type
status Concept Type: attribute	timecard *has* status Concept Type: is-property-of fact type Synonymous Form: status *of* timecard Necessity timecard **can** *have* **at most** one status
reference Concept Type: attribute	timecard *has* reference Concept Type: is-property-of fact type Synonymous Form: reference *of* timecard Necessity: timecard **must** *have* reference
actionCode Concept Type: attribute	timecard *has* actionCode Concept Type: is-property-of fact type Synonymous Form: actionCode *of* timecard Necessity: timecard **can** *have* **at most** one actionCode
resourceInfo Concept Type: attribute	timecard *has* resourceInfo Concept Type: is-property-of fact type Synonymous Form: resourceInfo *of* timecard Necessity: timecard **must** *have* one resourceInfo
approvalInfo Concept Type: attribute	timecard *has* approvalInfo Concept Type: is-property-of fact type Synonymous Form: approvalInfo *of* timecard Necessity: timecard **can** *have* approvalInfo

```
submitterInfo
  Concept Type:          attribute
```

```
timecard has submitterInfo
  Concept Type:          is-property-of fact type
  Synonymous Form:       submitterInfo of timecard
  Necessity:             timecard can have at most one
                         submitterInfo
```

```
reportElement
  Concept Type:          attribute
```

```
timecard has reportElement
  Concept Type:          is-property-of fact type
  Synonymous Form:       reportElement of timecard
  Necessity:             timecard can have reportElement
```

```
comment
  Concept Type:          attribute
```

```
timecard has comment
  Concept Type:          is-property-of fact type
  Synonymous Form:       comment of timecard
```

Secondly, the relations with other business concepts are defined.

```
timecard must have one human resource
timecard must have one submitter
timecard must have at least one receiver
timecard can have assignment
timecard has at least one timecardLine
timecard can have invoice
```

Thirdly, unary fact types are defined. The definition of the unary fact type *is unique*, comes from BR2 of section 6.4.

```
identifier is unique
  Definition:            identifier is unique when identifier of timecard = customer number
  Concept Type:          characteristic
```

The 'correctionTimecard' is not depicted in the timecard object model, as can be seen in fig. 38. No reason is given why this was left out, so we can assume that this has been forgotten. Since the details of the correctionTimecard are unknown in the SETU standard, we have assumed that a correctionTimecard must have one 'identifier' so it can have a reference to a timecard object. The unary fact type (i.e. identifier *is unique*) was not necessary here as it was already defined for timecard. This is also depicted in the timecard object model in fig. 38.

There are some structure rules for the parties and roles as well. From the interaction & behavior section we have used the context diagram with its (use case) description, see fig. 39 and table 8 respectively.

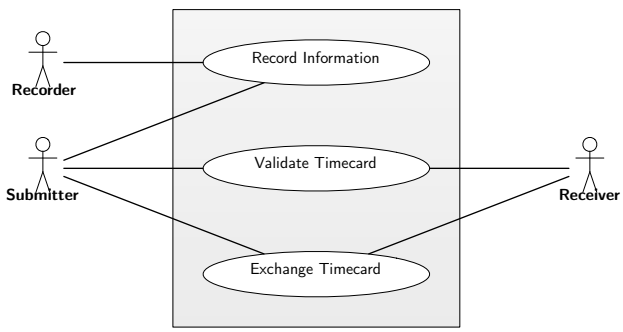


Figure 39. Reporting time & expenses context diagram in UML (adapted from Folmer et al., 2009).

Use case	Role
Record information	*Submitter* The submitter records and collects the time & expenses information and compiles it into a comprehensive record.
	Recorder The recorder merely records registration information.
Validate timecard	*Submitter* The submitter validates the time & expenses information on the timecard against the agreement.
	Receiver The receiver validates the timecard against the agreement as referenced in the timecard and accepts or rejects the information contained in the timecard.
Exchange timecard	*Submitter* The submitter sends time & expenses information on a timecard to a receiver.
	Receiver The receiver receives time & expenses information on a timecard from a submitter.

Table 8. Use case descriptions (adapted from Folmer et al., 2009).

This information on roles led to the following structural rules submitter. The rest of the rules can be found in appendix G.

```
All of the following are always true for a submitter:
  • It can record time information
  • It can validate timecard
  • It can send timecard
```

There are structural rules concerning parties that have not been defined yet. There is a generalization between *party* and *shop floor recording system*, *staffing consumer*, and *staffing company*. Because not every party can be a *recorder* for instance, some structural rules need to defined. From table 5 of the 'structure' section of the SETU standard we have defined the following structural rules for parties.

```
All of the following are always true for a shop floor recording system:
  • It can be a recorder
  • It can be a submitter
```

Just like every SETU standard, this one has a relation with other standards as well, specifically with other business concepts, i.e. invoice and assignment. The relationship with these concepts is discussed briefly, stating that a timecard can be used for invoice generation and that assignment contains agreements on working conditions. These relationships are not modeled as too little information is available and they are not relevant for this SETU standard.

We have now modeled all the relevant structure rules, which can be found in appendix G. The next step is to define the power i.e. processes.

6.4. Power

In this section we will define the processes ("muscles") and activities declaratively using Stornebrink's framework (2010). The processes are derived from the activity diagram, state transition diagram and their descriptions from the 'interaction & behavior' section of the SETU standard. The activity diagram and and its description are shown in fig. 40 and table 9 respectively.

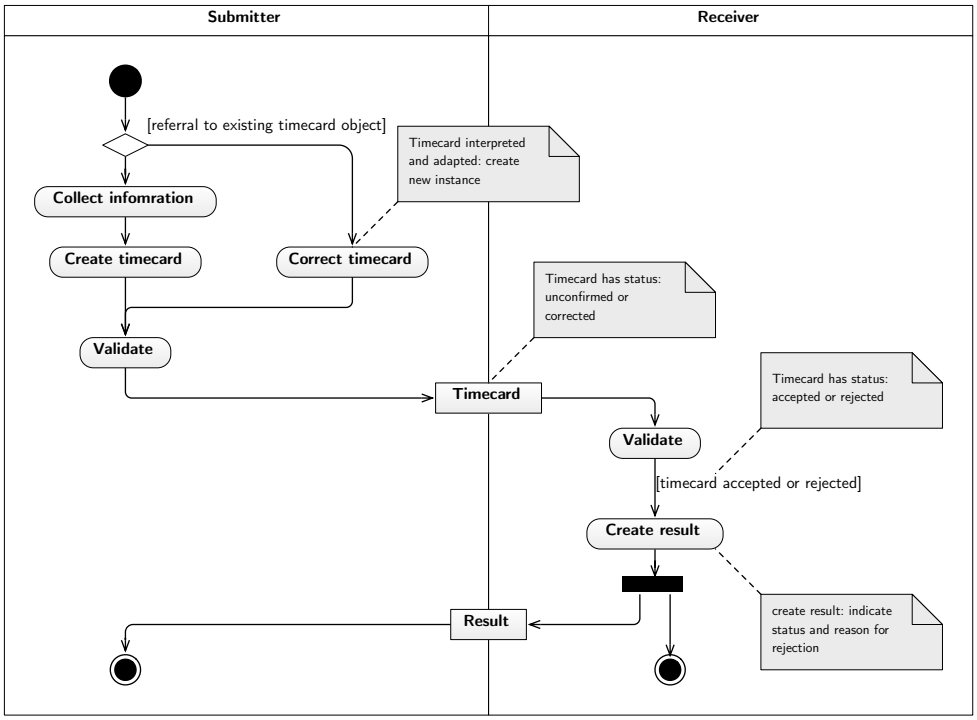


Figure 40. Reporting time & expenses activity diagram (adapted from Folmer et al., 2009).

The activities in fig. 40 are all processes. There is little inconsistency with the rest of the descriptions in the SETU standard, so sometimes an alternative process has been described. The notes in the activity diagram provide additional information. Table 8 describes these activities and is a valuable source of information as well.

Activity	Description
Collect information	The submitter role collects the registration information and processes it if necessary, before it puts the timecard together.
Create timecard	The submitter creates a timecard that includes the relevant information from its collection activities.
Correct timecard	If there is a request from a receiver for an updated or corrected version of a specific timecard, the submitter has to interpret the reason for the request, correct the timecard that is being referred and put together a new timecard.
Validate (supplying party)	Validation is required on information present on the timecard before the timecard is being sent out to a receiver.

Activity	Description
Validate (processing party)	The receiver will validate the timecard on reception and will accept or reject it accordingly.
Create result	After validation, the receiver will put together a result when it rejects the timecard and send it back to the original submitter, along with the specific reason for rejection.

Table 9. Reporting time & expenses activities (adapted from Folmer et al., 2009).

Statuses of timecards are depicted in a state transition diagram shown in fig. 41. Its description can be found in table 10.

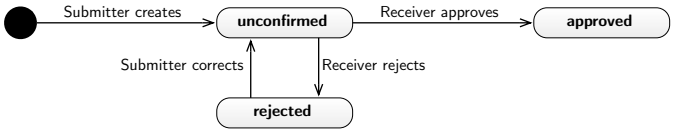


Figure 41. Timecard state transition diagram (adapted from Folmer et al., 2009).

Source state	Destination state	Event	Description
unconfirmed	rejected	Receiver rejects	The receiver rejects the timecard and sends a reason for disapproval.
unconfirmed	approved	Receiver approves	The receiver confirms the timecard to be correct.
rejected	unconfirmed	Submitter corrects	The submitter replaces or adds to the contents of the timecard by providing corrected information.

Table 10. Timecard states and events (adapted from Folmer et al., 2009).

Below we describe some of the activities found in fig. 40. Some activities have been decomposed because they consist of multiple processes. The rest of these activities can be found in appendix H. Note that there are no spaces in process names, they are replaced by dashes to show that they are processes, e.g. process name is written as process-name.

collect-information
  Pre-condition: -
  Boundary-condition: -
  Post-condition: reportedTime *is available*

  expenseAllowance *is available*

create-timecard
  Pre-condition: create-timecard may be executed when any of the following is true:

  - reportedTime *is available*
  - expenseAllowance *is available*

  Boundary-condition: -
  Post-condition: timecard must *be created*

  status *of* timecard must be set to unconfirmed

We have now modeled every activity with the possible statuses a timecard object can have. The unary fact types such as *is available*, *is known*, *is created*, etc. should be defined in the business vocabulary as well, but they don't contribute much to the objective of this research, and are therefor not further defined. There are no boundary-conditions defined here, because they "govern the work that is done during a process" (Stornebrink, 2010), the SETU standard does not define these kinds of rules, since their aim is the semantics of data exchanged.

We have encountered a few inconsistencies in the activity diagram, e.g. the status 'corrected' does not exist in contrast to what the timecard note says, in fig. 40. Also the validate activity does not have a post-condition because it is not defined when a timecard is valid. A few questions that remain unanswered concerning 'interaction and behavior' are:

- When do you start collecting information?
- When is a timecard valid?
- What is considered 'registration information' (fig. 35 and table 5)?
- The 'correct timecard' activity suggests that there is a difference between an *updated* and a *corrected* timecard, and since every change to a timecard should result in a correctionTimecard, and there is only one type of correctionTimecard defined, we are missing some information which is not defined or explained in the SETU standard.

6.5. Control

In this final phase we will define the operative business rules that exercise control ("the nervous system"). The SETU standard provides a set of four business rules that constrain the models, see table 11. These rules are derived from the mapping process of the SETU standard to HR-XML SIDES.

Business rule	Description
BR1	Timecard objects need to be uniquely identifiable. Within the context of the issuer of the Timecard the Identifier therefore has to be unique.
BR2	The combination of the customer number and the identifier ensures that the timecard can be uniquely identified at a staffing agency.
BR3	The attribute ResourceInfo of the Timecard references a HumanResource instance/message that can contain an elaborate description of the temporary worker. Elaborate information about the temporary worker therefore should not be contained in a Timecard.
BR4	DateTimeInfo is specified by a start datetime and an end datetime, with the end datetime being exclusive of the timeframe.

Table 11. Business rules (adapted from Folmer et al., 2009).

These business rules can be translated to the RuleSpeak as well. They would be defined as (in the same order as table 11):

```
identifier of timecard is unique
```

```
identifier is unique when identifier of timecard = customer number
```

```
elaborate information should not be in resourceInfo of timecard
```

```
dateTimeInfo
  Definition:              a period
```

Note: Period is defined as: *"A time interval measured from a start date/time to an end date/time"* (Object Management Group, 2008).

There are several business rules that should exercise control. They are based on the models and descriptions of the SETU standard. These rules create an "artificial flow".

```
an unconfirmed timecard must be either accepted or rejected
```

```
a validated timecard must be sent to a receiver
```

The complete set of rules can be found in appendix I.

6.6. Summary

In this experiment we have applied BRASD to the real-life semantic IS standard, the *SETU Standard for Reporting Time & Expenses 1.1*. Approaching development from a business rules perspective forces the developer to think about what a concept and fact type means. A good business vocabulary ensures that there is no misconception about what something means. This approach has led to the discovery of some errors in the standard which haven't been fixed yet. For example, throughout the standard the attribute `ActionCode` of Timecard was forgotten in tables and models. There has been an errata to fix the forgotten attribute (Folmer et al., 2010), however there is still a model (fig. 7 in the standard) that doesn't have the new attribute, see fig. 42 for a corrected version. The fact model allowed us to see that there is no one to record or submit `ExpenseAllowance`, see fig. 37 in section 6.3.

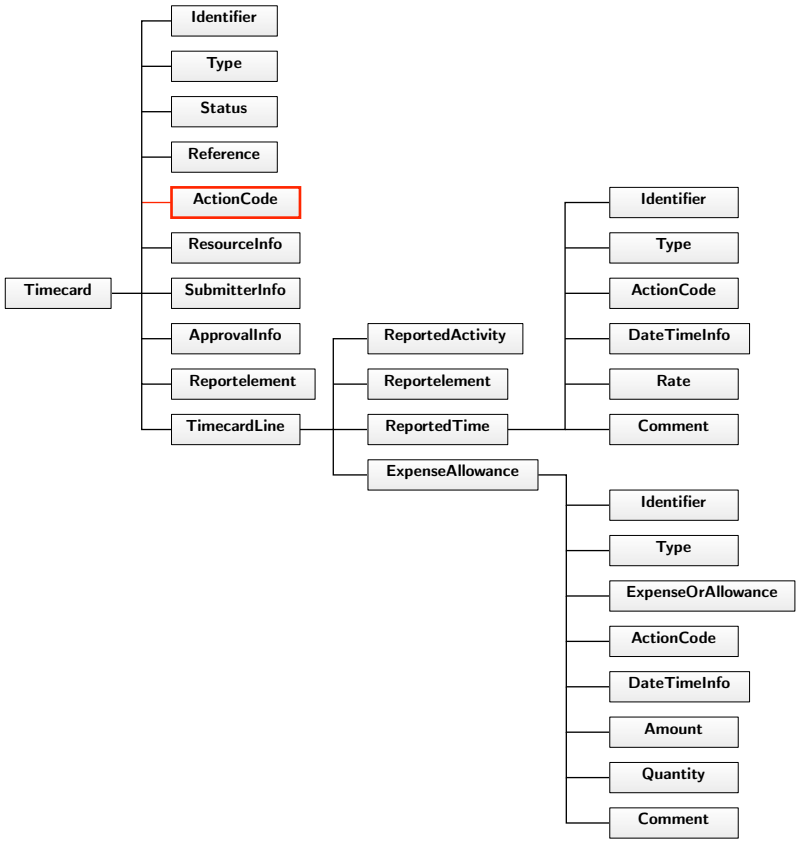


Figure 42. Corrected timecard message definition (adapted from Folmer et al., 2009).

These inconsistencies can come from the fact that each SETU standard has its specific model, which reuses some classes or attributes of other models. The relation between any two class diagrams is visualized by including classes of one diagram in another, see 'PartiesRoles', 'Invoice' and 'Assignment' in fig. 36, section 6.2. This makes maintenance and extension of these separate models a labor intensive and error prone process (Hofman et al., 2010). The SETU standards for the staffing industry are based on the 'message' paradigm, this means that "one way or another, the use of the standards eventually results in a 'controlled' sequence of electronic messages" (Hofman et al., 2010). This leads to less attention for the services that actors involved should provide and the constraints to these services.

We have also discovered that a timecard can never have the status `corrected` even though it is clearly stated in the state transition diagram in fig. 40 of section 6.4. This case study shows that a business rule approach to semantic IS standard development is possible and its thorough nature enables us to create more consistency throughout the standard.

7. Evaluation

In this chapter we evaluate if a business rule approach can reduce development/maintenance duration, increase quality, and increase understandability. The experiment of chapter 6 served as input for an open interview at SETU in Enschede. First we will describe why we have chosen to do a case study in section 7.1. Then we will explain how this case study was approached and designed in section 7.2. The interview is discussed and interpreted in section 7.3. The survey and its results are discussed in section 7.4 Finally we conclude with a summary in section 7.5.

7.1. Motivation

After experimenting with BRASD we have seen that a business rules approach to semantic IS standards development is quite doable and that their concepts translate well to business rules. However, to show if this approach can contribute to solving the problems identified in section 2.1, we have to evaluate our work with real-life experts from the standards development domain. This form of *respondent validation* lessens the misinterpretation of self-reported views (Yin, 2010). Having a good case is crucial because *"selecting case(s) serves as possibly the most critical step in doing case study research"* (Stake, 1994, p.243 as cited by Green, Camilli, Elmore & American Educational Research Association, 2006). Our case is 'a business rule approach to semantic IS standards development'. The case selection comes from (preliminary) literature review. This case is interesting for standards development organizations such as SETU because it tries to address current issues with standards development, see section 2.1.

7.2. Approach

Where the purpose of the experiment was *validation* of our BRASD approach, the purpose of this part is to *evaluate* if a business rules approach on semantic IS standards development can lead to cost reduction, and to quality and understandability increase. Yin's (2010) framework was used to design the case study. Our case study research approach is illustrated in fig. 43.

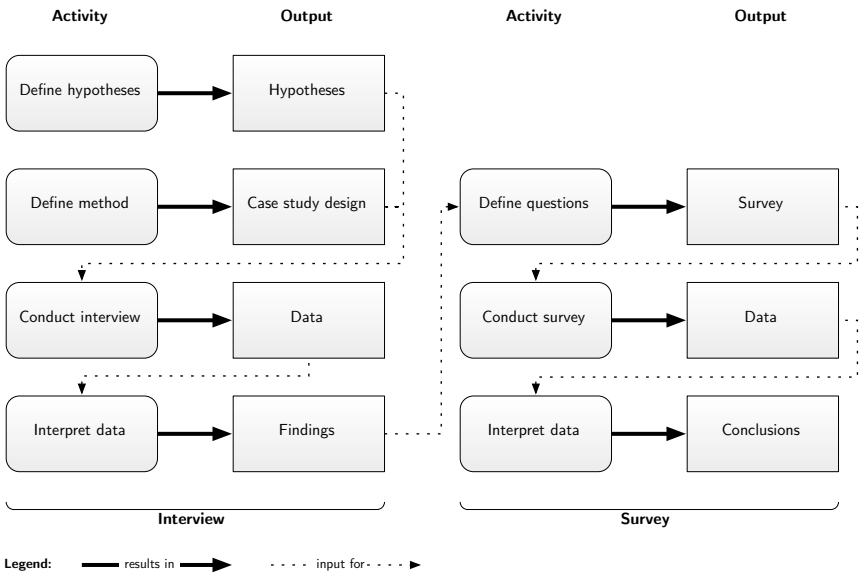


Figure 43. Case study research approach.

The case study is split into two parts. The first part of the case study is an open interview where we test 3 out of 4 hypotheses. We expect that quality will be where BRASD will have significant meaning, therefore we will test the quality hypothesis in a separate survey. Depending on the findings in the open interview, we will define questions/hypotheses which relate to the quality characteristics (i.e. accuracy, completeness, consistency, compliancy, precision, traceability, understandability and portability). The second part of the case study is the survey to test the hypotheses about quality. The contents of both the hypotheses and questions will be explained in the next sub-sections.

7.3. Interview

We have chosen to do an open interview with the developers and maintainers of the SETU standard ('SETU Standard for Reporting Time & Expenses 1.1'). SETU's knowledge and experience is considered valuable for this research because they have first-hand experience in the process, (im)possibilities and shortcomings of semantic IS standards development.

The developers present were Michael van Bekkum, Erwin Folmer, Dennis Krukkert and Jasper Roes. All of them have affinity with standards development, specifically semantic IS standards. Van Bekkum, Folmer and Roes have developed the 'SETU Standard for Reporting Time & Expenses 1.1' — Their perspective on the first part of the case study is considered valuable.

7.3.1. Hypotheses

Our main hypothesis was:

- *"A business rule approach to semantic IS standards development can lead to reduction of costs and increase of quality and understandability."*

This hypothesis was either accepted or rejected by answering the main question:

- *"Can a business rule approach to semantic IS standards development lead to: cost reduction; quality increase; and/or understandability increase?"*

The main hypothesis has been divided into 4 sub-hypotheses that specifically target the problems of section 2.1. Fig. 44 illustrates how the hypotheses relate to the problems. Note that hypothesis 3 is tested with a survey, but we will decide from the results of the interview if we will conduct a survey as well.

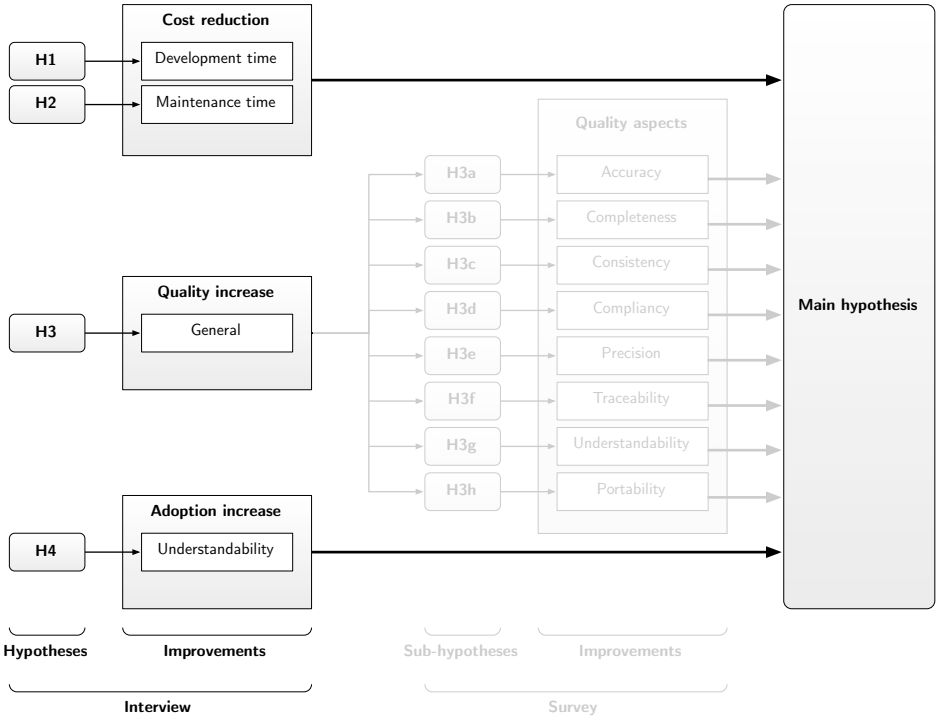


Figure 44. Hypotheses vs. improvement.

Cost reduction:

- *H1 — "A business rule approach helps reducing semantic IS standards development time."*
- *H2 — "A business rule approach helps reducing semantic IS standards maintenance time."*

Quality increase:

- *H3 — "A business rule approach helps increasing semantic IS standards quality."*

Adoption increase:

- *H4 — "A business rule approach helps increasing understandability by decreasing communication barriers between business people and IT people."*

7.3.2. Case study design

Our case ('a business rule approach to semantic IS standards development') is a *single case* type because it can test both the applicability (chapter 6) and utility (this chapter) of BRASD. The *units of analysis* are *embedded designs* in the form of sub-hypotheses. Our data source is an open-ended interview with the standards developers at SETU, followed by an online survey. The interview is recorded with a smartphone and a laptop.

7.3.3. Conducting interview & interpretation of data

The interview was conducted on July 21st 2011 at SETU in Enschede. Having four respondents means that there are three or more sources of data, which increases validity of our data collection (i.e. triangulation principle). The rule-based semantic IS standard (from chapter 6) helped creating a basic understanding of the business rule approach. After the interview there was enough ground to conduct a survey about the

quality characteristics which might be increased with BRASD. Also some important concerns have been pointed out by the respondents. Some of the key points were addressed during the interview and some needed further research. The key points are discussed below in a 'problem' and 'solution' form:

1. *Business rule maintenance could be a problem if done manually with a large amount of rules.*

Business rules creation and maintenance can (and should be) done in an automated fashion within a BRMS, see section 4.6 for examples.

2. *"Concepts are defined with terms and fact types, which are defined with terms and fact types again, how far do you go?"*

Definitions should be as detailed as the business needs them to be, see point #2 in sub-section 5.1.2. This point led to refinement of the framework/BRASD.

3. *The respondents stated that standards maintenance duration is mainly influenced by requirements elicitation and definition and not process modeling.*

Further research on this point led to the conclusion that a business rule approach doesn't reduce development time of a standard, thus *H1* was rejected.

4. *One respondent was wondering if one can differentiate between a concept and an instance of that concept.*

Like stated in section 4.3.1, a fact type represents relationships between concepts, whereas facts represents an instance of those relationships, e.g. `Thomas` ***sends*** `claim#1234` is of fact type `client` ***sends*** `claim`. This point led to refinement of the framework/BRASD.

5. *One respondent questioned if circular reasoning could be avoided when describing processes.*

There is currently no solution to this point, because there are no proper tools available yet. However Stornebrink's framework states that: a process is anti-symmetric and irreflexive. An irreflexive process cannot be a subprocess of itself and anti-symmetry prohibits child objects to be linked back to their parent by the same relation. A rule engine could detect these problems while modeling, however, there is no system that can do this yet.

6. *Respondents were wondering if the generation of (UML) models from business rules is possible.*

The generation of UML is possible because SBVR is a CIM which uses the XML-based XMI to exchange meta-data information, i.e. models. Translators like ATL or XSL Transformations (XSLT) can make this transition. Section 4.7 shows how a SBVR → UML transformation is done and appendix E shows how 'Structured English' can be transformed to an XMI document. Once the model is defined in XMI, transitions to other models are possible. Fig. 45 illustrates this.

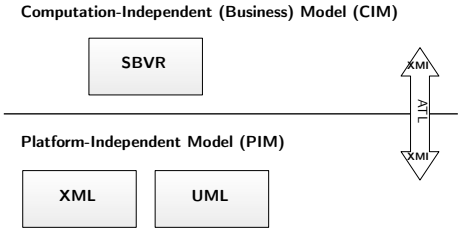


Figure 45. SBVR to UML transformation.

7. *The respondents questioned if the order of attributes in XSD files can be preserved from when using SBVR.*

The order of attributes refers to the XSD `<sequence>` indicator element which specifies that the child elements must appear in a specific order. The `<sequence>` element is not supported by XMI or SBVR.

At the end of the interview, it because obvious that there was not enough ground to support *H1*, *H2*, and *H4*. As expected, BRASD might increase quality attributes of semantic IS standards, which are discussed in the next section.

7.4. Survey

After consulting with the experts at SETU, they agreed to fill out an online survey on quality characteristics of semantic IS standards which a business rule approach might enhance. The ISO/IEC 25012 document defines 15 data quality characteristics. We have selected 8 of these characteristics which might be related to semantics IS standards (ISO IEC, 2008):

1. **Accuracy** – The extent to which data has attributes that correctly represent the true value of the intended attribute of a concept of event in a specific context of use.
2. **Completeness** – The extent to which subjects associated with an entity have values for all expected attributes and related entity instances in a specific context of use.
3. **Consistency** – The extent to which data has attributes that are free from contradiction and coherent with other data in a specific context of use
4. **Compliancy** – The extent to which data has attributes that adhere to standards, conventions, or regulations in force and similar rules relating to data quality in a specific context of use.
5. **Precision** – The extent to which data has attributes that are exact or that provide discrimination in a specific context of use.
6. **Traceability** – The extent to which data has attributes that provide an audit trail of accesses to the data and of any changes made to the data in a specific context of use.
7. **Understandability** – The extent to which data (and associated metadata) has attributes that enable it to be read and easily interpreted by users and are expressed in appropriate languages, symbols, and units in a specific context of use.
8. **Portability** – The extent to which data has attributes that enable it to be moved from one platform to another, preserving the existing quality in a specific context of use.

This list of characteristics was also provided with the online survey. The next sub-section explain the survey questions and results.

7.4.1. Survey questions

Hypothesis 3 is formulated as follows:

- *H3 — "A business rule approach helps increasing semantic IS standards quality."*

The following "agree/disagree questions" were asked to the respondents to determine if a business rules approach could lead to quality increase. The hypotheses corresponding with the new survey questions are labeled *H3a*, *H3b*, ..., *H3h* in fig. 46. The following questions were posed to the respondents.

Do you think that a business rules approach to semantic IS standards development, could/will lead to:

a. Accuracy increase?

b. Completeness increase?

c. Consistency increase?

d. Compliancy increase?

e. Precision increase?

f. Traceability increase?

g. Understandability increase?

h. Portability increase?

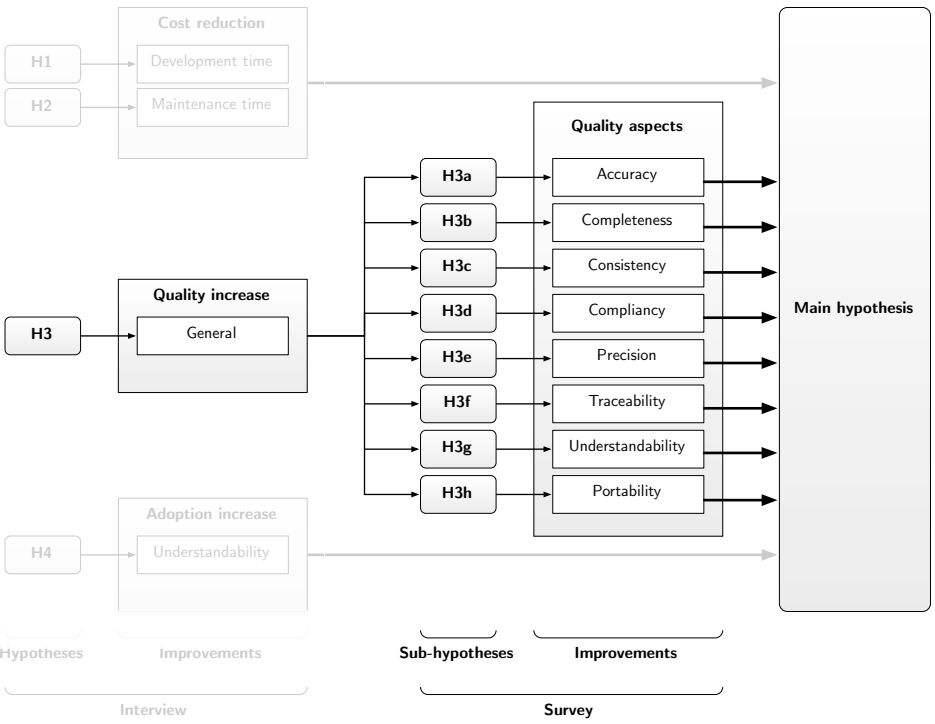


Figure 46. Sub-hypotheses vs. improvements.

7.4.2. Survey results

Case studies don't represent a formal sample from some larger universe, it is no statistical generalization (Yin, 2010). Instead, generalizing from case studies reflects substantive topics or issues of interest, and the making of logical inferences (analytic generalization). In this case the focus was on quality-related aspects

only. The survey results can be seen in fig. 47. As we can see, almost all of the respondents disagreed with *H3g* (understandability) and *H3h* (portability). A reason for a disagreement on *H3g* was that models are still necessarily for the understandability of the standard, even though UML models *can* be generated from business rules (Kleiner et al., 2009; Raj et al., 2008), the experts did not agree on this. The opinion of the respondents was split on *H3a* (accuracy) and *H3f* (traceability). The respondents that agreed, have not provided any motivation for their choice, while the other two explained that any formal method could enhance accuracy, this doesn't necessarily have to come from a business rules approach. Three respondents agreed that a business rules approach forced a more complete result (*H3b*), while one respondent mentioned that a modeling technique does not enforce completeness, however, the business rule approach is not a modeling technique, it is a way of approaching and defining a business context with its elements. The respondents did not unanimously agree on sub-hypotheses *H3a* (accuracy), *H3b* (completeness) and *H3f* (traceability), this means that there is some potential for quality increase in those aspects, but that it is not evident. Except for *H3b* (completeness), we think that completeness is related to consistency from a business rule perspective. We think that one respondent misunderstood the question and thus his answer should not dismiss this hypothesis. The respondents unanimously agreed on *H3c* (consistency), *H3d* (compliance), and *H3e* (precision). Even though one respondent found it hard to either agree or disagree on *H3e*. The quality aspects *completeness*, *consistency*, *compliance* and *precision* are the only strong points that are recognized by our focus group.

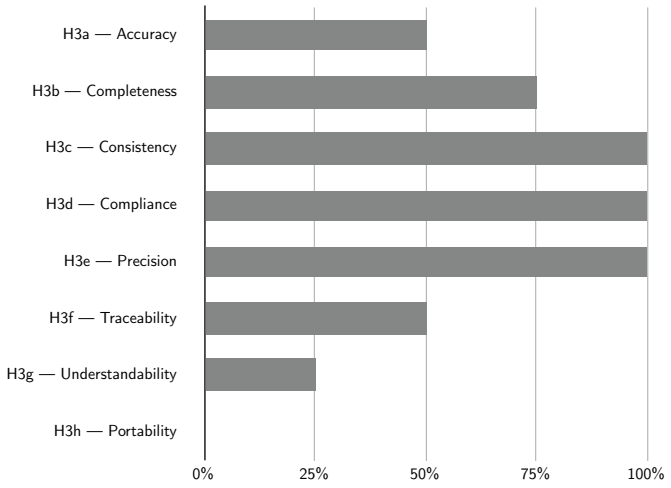


Figure 47. Survey results.

7.5. Summary

This case study has shows us that a business rule approach to semantic IS standards development can lead to quality improvement in terms of *completeness*, *consistency*, *compliance*. and *precision*. Our approach (BRASD) has shown that quality aspects as completeness and consistency are enforced during development. A business rule approach can also provide more compliance and precision in semantic IS standards development, but the usage of a BRMS will be necessary to reap the benefits. Other quality aspects were not convincing enough (i.e. no triangulation) to be able to state that a business rule approach will address that problem. Fig. 48 illustrates which hypothesis were not supported (grey lines are not supported). A summary of the survey results and motivation can be found in appendix K.

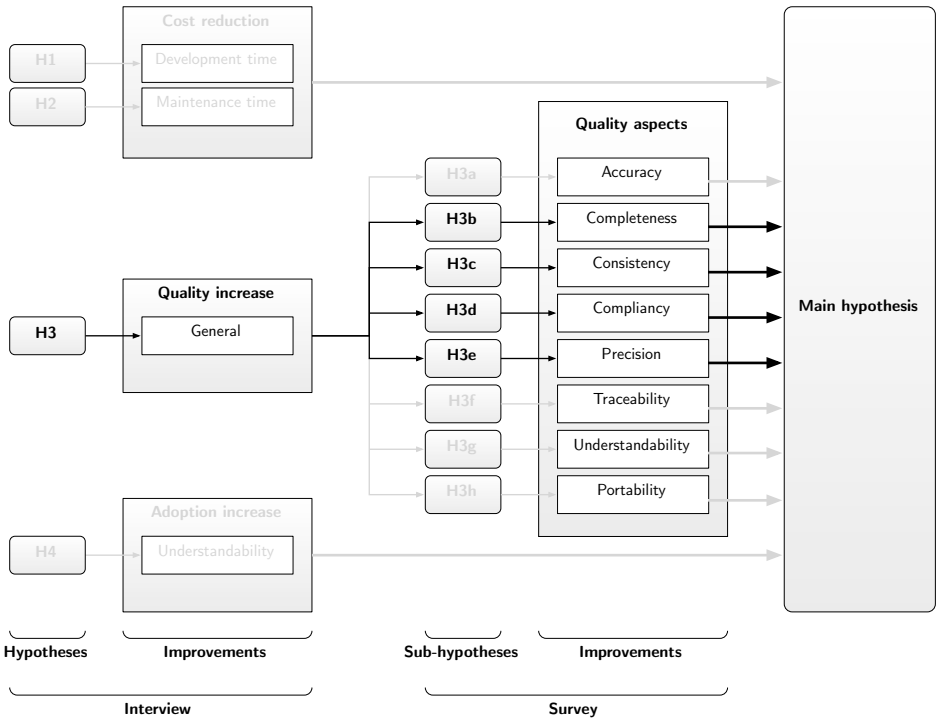


Figure 48. Interview and survey results.

8. Conclusions

This chapter concludes this research by looking back at the main research question and how is has been answered in section 8.1. Then the limitations of the research are discussed in section 8.2. Finally possible future researches are that could originate from this one are discussed in section 8.3.

8.1. A business rule approach to semantic IS standards development

The objective of this research was to determine if a business rule approach can contribute to reducing cost-related problems such as the development duration and maintenance duration of the standard; improving quality characteristics; and increasing understandability of the standard. To achieve this objective, a rule-based semantic IS standard was needed, therefore we defined the following main research question:

- *RQ — "What are the characteristics of a rule-based semantic IS standard that has advantages in relation with current technologies and how can it be created?"*

By answering four research questions, we have been able to identify the concepts of semantic IS standards and how they should be expressed with business rules. This resulted in our conceptual framework (section 5.2). The framework served as input for creating our business rule approach: BRASD. BRASD describes in a structured way how to create rule-based semantic IS standards. Executing four (main) steps results in a complete rule-based semantic IS standard. The steps are: (1) define business vocabulary; (2) define structure; (3) define power; and (4) define control. Fig. 34 in section 5.2 summarizes these steps in a vignette.

To validate our method, an experiment was conducted by transforming a semantic IS standard, the *SETU Standard for Reporting Time & Expenses 1.1*, to a rule-based version. The rule-based version served as reference/input for our case study at SETU. This case study was conducted to determine if a business rule approach can lead to const reduction, quality increase and understandability increase. The case study consisted of an open interview and a survey. The results indicate that a business rule approach such as BRASD, can contribute to the increase of quality in terms of completeness, consistency, compliance and precision. This is particularly important because quality influences the adoption of standards. Our framework and approach can aid standards developers to create rule-based semantic IS standards which can potentially lead to the creation of higher quality semantic IS standards. There was not enough ground to state that a business rule approach like BRASD can lead to cost reduction or understandability increase.

Even though BRASD has potential for quality increase, it has its limitations as well. BRASD describes declarative process modeling, which is a flexible way of modeling processes compared to the traditional procedural process modeling techniques. However, there are no commercial or open-source rule engines that support these kinds of processes yet. The survey also suggested that traditional process models are more understandable than declarative ones, meaning that until a proper rule engine for declarative process modeling is available, traditional process modeling will have to used.

Other types of models such as UML sequence- and activity diagrams, can in theory be generated from business rules (Raj et al., 2008), there is no commercial or open-source implementation for this kind of transformation. However, the Eclipse IDE is actively developing an extension to allow for SBVR → MOF model transformations which would make SBVR → UML transformations a reality.

SBVR is a powerful standard which offers a higher-order of logic than the most rule modeling languages (which offer first-order predicate logic). SBVR supports the concept of meaning, which can describe any real-life situation. This allowed us to completely transform the SETU standard into RuleSpeak (which is a rule modeling language based on SBVR). SBVR vocabularies and rules are documented in the XMI format, which enables model-to-model transformations (appendix E) and transformations to other XML formats such as XSD. However a technical aspect could not be addressed: the absence of the XSD <sequence>

indicator element in XMI. We haven't been able to find information if the absence of this element is really a problem when a business rule approach is used for semantic IS standards development.

SBVR has other limitations as well. Business rule management is a principle of the business rule approach, especially when the number of rules increase. There are open-source and commercial implementations of BRMSs that can manage large quantities of business rules. However, we have not found a BRMS that can handle SBVR vocabularies and rules. So this remains a concern for using SBVR.

8.2. Limitations of the research

This research has several limitations which will be summed up here. The first is that BRASD was applied to one semantic IS standard. Even though the contents of semantic IS standards have been identified through extensive research, it does not mean that our framework works with other standards as well as it did with the SETU standard. Further research is needed to validate our framework on a larger scale.

The second limitation is that our framework was used to *translate* the SETU standard to RuleSpeak instead of using it to *develop* a semantic IS standard. This limitation could not have been avoided because we are not able to apply BRASD during a development phase of a real-life semantic IS standard.

A third limitation is that we haven't been able to do an automated SBVR → XMI → XSD transformation to determine the (in)consistency with current XSDs of the SETU standard. XSLT might be a solution once there is a well-formed and valid XMI representation of the SETU standard available.

A fourth limitation is that the focus group consisted of four respondents, which can impact the results greatly as each vote can impact the final result with 25%. Further research is needed to verify the benefits of BRASD with other SSOs.

8.3. Future work

Being able to generate UML models from an SBVR vocabulary and rules is a powerful functionality as it leads to more consistent models because they originate from a single source. Additional research is needed to determine the benefits of BRASD from a technical perspective once SBVR-to-model transformations are available with tools like the Eclipse IDE in combination with ATL.

Another interesting subject is the creation of BPMN models from rule-based process models (like the ones in appendix H). The ATL extension of the Eclipse IDE supports model-to-model transformation. Doing this would require a rule-based business process meta-model (Stornebrink, 2010) and a meta-model of BPMN. The BPMN meta-model is available through OMG, the rule-based business process meta-model needs to be created from the the already existing ADL model. Enabling this transformation would mean that a BPMN process models can be generated from SBVR.

Another interesting topic of research could be *"What are the characteristics of a proper SBVR business rule management environment?"*. During the interview it became clear that rule management was a concern of the experts. Addressing this could create more adoption to rule-based semantic IS standards through SBVR.

References

Aldewereld, H. (2009, September 22). Rule Based Knowledge Systems. *Universiteit Utrecht*. Utrecht. Retrieved from http://www.cs.uu.nl/docs/vakken/mdks/lectures/lecture4.pdf

Baisley, D. E., Hall, J. & Chapin, D. (2005). Semantic Formulations in SBVR. Paper presented at the W3C Workshop on Rule Languages for Interoperability, Washington D.C.

Bajwa, I. S., Lee, M. G. & Bordbar, B. (2011). SBVR Business Rules Generation From Natural Language Specification. In *AAAI 2011 Spring Symposium Series - AI for Business Agility (AI4BA)* (pp. 2–8). Paper presented at the AAAI 2011 Spring Symposium (SS-11-03), Birmingham.

Beck, R. & Weitzel, T. (2005). Some Economics of Vertical Standards: Integrating SMEs in EDI Supply Chains. *Electronic Markets*, *15*(4), 313–322. doi:10.1080/10196780500302781

Boh, W. F., Soh, C. & Yeo, S. (2007). Standards development and diffusion: A case study of RosettaNet. *Communications of the Acm*, *50*(12), 57–62.

Brocke, vom, J. & Rosemann, M. (2009). *Handbook on Business Process Management 1*. Introduction, Methods, and Information Systems (pp. 1-600). Berlin: Springer-Verlag.

Cathomen, I. & Klein, S. (1997). The Development of FEDI in Switzerland: A Life-cycle Approach. *International Journal of Electronic Commerce*, *1*(4), 129–145. M. E. Sharpe, Inc.

Choia, B., Raghu, T. S. & Vinze, A. (2004). Addressing a standards creation process: a focus on ebXML. *International Journal of Human-Computer Studies*, *61*(5), 627–648.

Davenport, T. H. (1992). *Process Innovation*. Reengineering Work Through Information Technology. Boston: Harvard Business School Press.

Folmer, E. & Punter, M. (2010). *Beheer- en OntwikkelModel voor Open Standaarden (BOMOS) versie 2* (pp. 1–128). Den Haag: NOiV.

Folmer, E. & Verhoosel, J. (2011). *State of the Art on Semantic IS Standardization, Interoperability & Quality* (pp. 1-167). Enschede, Den Haag: TNO, Universiteit Twente, NOiV, CTIT.

Folmer, E., Oude Luttighuis, P. & Van Hillegersberg, J. (2011). Paper presented at the 16th EURAS Annual Standardization Conference, Kaunas.

Folmer, E., Roes, J. & van Bekkum, M. (2009). *SETU Standard*. Standard for Reporting Time & Expenses 1.1 (pp. 1-86). Badhoevedorp. Retrieved from http://www.setu.nl

Folmer, E., Roes, J. & van Bekkum, M. (2010). *SETU Standard Errata*. Errata Standard for Reporting Time & Expenses v1.1 (pp. 1-3). Badhoevedorp. Retrieved from http://www.setu.nl

Goedertier, S. & Vanthienen, J. (2007). Declarative Process Modeling with Business Vocabulary and Business Rules, 1–10.

González, A. J. & Dankel, D. D. (1993). *The Engineering of Knowledge-based Systems*. Theory and Practice (pp. 1–593). Englewood Cliffs: Prentice Hall.

Green, J. L., Camilli, G., Elmore, P. B. & American Educational Research Association (2006). Case Study Methods. In *Handbook of Complementary Methods in Education Research* (pp. 111–121). Mahwah: Lawrence Erlbaum.

Hall, J. & Healy, K. A. (2000). *Defining Business Rules: What Are They Really?* (pp. 1–77). Business Rule Group. Retrieved from http://www.businessrulesgroup.org

Hevner, A., March, S. & Park, J. (2004). Design science in information systems research. *MIS Quarterly*.

Hofman, W., Holtkamp, M. & Bekkum, M. (2010). Specification of SETU with WSMO. In K. Popplewell, J. Harding, R. Poler & R. Chalmeta (Eds.), *Enterprise Interoperability IV* (pp. 341–354). London: Springer London. doi:10.1007/978-1-84996-257-5_32

HR-XML Consortium. (2007). *Staffing Industry Data Exchange Standards (SIDES)*. (K. Bartkus, Ed.). Retrieved from http://ns.hr-xml.org/2_5/HR-XML-2_5/SIDES/SIDES.html

Huberman, M. B. & Miles, M. A. (1994). *Qualitative Data Analysis*. (Second Edition. pp. 1-185). Thousand Oaks: SAGE Publications.

ISO IEC. (2008). *ISO/IEC 25012: Software Engineering-Software product Quality Requirements and Evaluation (SQuaRE)- Data Quality Model*.

Joosten, S., Wedemeijer, L. & Michels, G. (2010). *Rule Based Design* (Draft. pp. 1–183). Retrieved from http://ampersand.sourceforge.net/ampersandWiki/index.php/Books

Katz, M. L. & Shapiro, C. (1985). Network Externalities, Competition, and Compatibility. *The American Economic Review*, *75*(3), 424–440. American Economic Association.

King, W. R. & He, J. (2005). Understanding the Role and Methods of Meta-Analysis in IS Research. *Communications of the Association for Information Systems*, *16*(1), 665–686. Retrieved from http://aisel.aisnet.org/cais/vol16/iss1/32

Kleiner, M. (2009, April). ATL Use Case. *eclipse.org*, Production of UML class diagrams from syntactical models of english text or SBVR models. Retrieved from http://www.eclipse.org/m2m/atl/usecases/Syntax2SBVR2UML/

Kleiner, M., Albert, P. & Bézivin, J. (2009). Parsing SBVR-Based Controlled Languages. In A. Schürr & B. Selic (Eds.), *Lecture Notes in Computer Science* (Vol. 5795, pp. 122–136). New York: Springer-Verlag Berlin Heidelberg. doi:10.1007/978-3-642-04425-0_10

Krislov, S. (1997). *How Nations Choose Product Standards and Standards Change Nations* (pp. 1-264). Pittsburg: University of Pittsburgh Press.

Lee, H. L. & Whang, S. (2004). *International Series in Operations Research & Management Science*. International Series in Operations Research & Management Science (Vol. 62, pp. 123–138). New York: Springer-Verlag. doi:10.1007/0-387-27275-5_8

Lim, A. S. (2006). *Power Battles in ICT Standards-Setting Process*. Lessons from Mobile Payments (pp. 1-246). Technische Universiteit Eindhoven, Eindhoven.

Linehan, M. H. (2008). SBVR Use Cases. In *The International RuleML Symposium on Rule Interchange and Applications* (Vol. 5321, pp. 182–196). Paper presented at the Proceedings of the International Symposium on Rule Interchange and Applications (RuleML '08), Orlando.

March, S. T. & Smith, G. F. (1995). Design and Natural-Science Research on Information Technology. In *Decision Support Systems* (Vol. 15, pp. 251–266). doi:10.1016/0167-9236(94)00041-2

Markus, M. L., Steinfeld, C. W., Wigans, R. T. & Minton, G. (2006). Industry-Wide Information Systems Standardization as Collective Action: The Case of the US Residential Mortgage Industry. *MIS Quarterly*, *30*, 439–465.

Masud, M. & Lucker, J. (2009). Commercial Insurance Carrier's Road to Business Agility. *Business Rules Journal*, *10*(10). Retrieved from http://www.brcommunity.com/a2009/b502.html

Musham, P., Singh, S., Bahal, R. & Tv, P. (2008). Visual SBVR. In *Digital Information Management, 2008* (pp. 676–683). Paper presented at Third International Conference on Digital Information Management (ICDIM 2008). doi:10.1109/ICDIM.2008.4746768

Myers, B. (2010). Success Story. *Business Rules Journal*, *11*(10). Retrieved from http://www.brcommunity.com/a2010/b555.html

Object Management Group. (2008). *Semantics of Business Vocabulary and Business Rules (SBVR)* (pp. 1–434). Retrieved from http://www.omg.org/spec/SBVR/1.0/

Odendahl, D. M. (Spring, 2009). ISC 320 Expert Systems and Knowledge Engineering. *OSWEGO State University of New York*. New York. Retrieved from http://www.cs.oswego.edu/~odendahl/coursework/isc320/notes/jackson/01-ab-characteristics.html

Paschke, A. (2006, September 21). Rule-based Knowledge Representation for Service Level Agreement. Paper presented at Doctoral Symposium of the 4th German Conference on Multiagent System Technologies (MATES '06), Germany.

Pau, R., Cabot, J. & Raventós, R. (2009). UMLtoSBVR: An SBVR-based tool to validate UML conceptual schemas. *Revista de Informática Teórica e Aplicada*, *16*(2), 105–108.

Pesic, M. & Aalst, W. M. P. (2006). A Declarative Approach for Flexible Business Processes Management. In J. Eder & S. Dustdar (Eds.), *Lecture Notes in Computer Science* (Vol. 4103, pp. 169–180). New York: Springer-Verlag Berlin Heidelberg. doi:10.1007/11837862_18

Petter, S. C. & Gallivan, M. J. (2004). Toward a Framework for Classifying and Guiding Mixed Method Research in Information Systems. In *Hawaii International Conference on System Sciences* (Track 8. pp. 80257a). Paper presented at the Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04). doi:10.1109/HICSS.2004.1265614

Raj, A., Prabhakar, T. V. & Hendryx, S. (2008). Transformation of SBVR business design to UML models. Paper presented at the Proceedings of the 1st India Software Engineering Conference (ISEC '08), Hyderabad, India. doi:10.1145/1342211.1342221

Ross, R. G. (2003). *Principles of the business rule approach* (pp. 1-372). Boston: Addison-Wesley Professional.

Ross, R. G. (2006a). The RuleSpeak® Business Rule Notation. *Business Rule Journal*, *7*(4), 1–6. Retrieved from http://www.BRCommunity.com/a2006/b282.html

Ross, R. G. (2006b). Concepts, Definitions, and Rules: RuleSpeak® Practices. *Business Rules Journal*, *7*(5), 1–6. Retrieved from http://www.BRCommunity.com/a2006/b289.html

Ross, R. G. (2009a). *Basic RuleSpeak® Guidelines*. Do's and Don'ts in Expressing Natural-Language Business Rules in English (2nd ed. pp. 1–23). Business Rule Solutions, LLC.

Ross, R. G. (2009b). *RuleSpeak® Sentence Forms*. Specifying Natural-Language Business Rules in English (2nd ed. pp. 1–10). Business Rules Solutions, LLC.

Ross, R. G. (2010, January). The Point of Knowledge. *Business Rules Journal*. Retrieved from http://www.brcommunity.com/a2010/b515.html

Shao, H. (2010). *Claims-Based Working*. (W. Pieters, M. E. Iacob & R. Joosten, Eds.). Access to business activities (pp. 1–81). University of Twente, Enschede.

Spivak, S. M. & Brenner, F. C. (2001). *Standardization Essentials: Principles and Practice* (pp. 1-316). New York: Marcel Dekker.

Steinfield, C. W., Wigand, R. T., Markus, M. L. & Minton, G. (2007). Principles of the business rule approach. In S. Greenstein & V. Stango (Eds.), *Standards and Public Policy* (pp. 160–207). Cambridge: Cambridge University Press.

Stornebrink, M. H. M. (2010). *Rule-based Process Design: A framework for rule-based process management* (pp. 1–87). University of Groningen, Groningen.

Thomas, J. W., Probets, S., Dawson, R. & King, T. (2008). A Case Study of the Adoption and Implementation of STEP. In Egyedi, T. M. & Blind, K. (Eds.), *The Dynamics of Standards* (pp. 117–134). Cheltenham: Edward Elgar.

Van Wessel, R. M. (2008). *Realizing Business Benefits from Company IT Standardization*. Case study research into the organizational value of IT standards, towards a company IT standardization management framework (pp. 1-248). Tilburg University, Tilburg.

Von Halle, B. & Goldberg, L. (2006). *Business Rule Revolution*. Running Business the Right Way (pp. 1-221). Silicon Valley: Happy About.

Weitzel, T., Beimborn, D. & Koenig, W. (2006). A Unified Economic Model of Standard Diffusion: The Impact of Standardization Cost, Network Effects, and Network Topology. *MIS Quarterly*, *30*, 489–514.

Weske, M. (2007). *Business Process Management*. Concepts, Languages, Architectures (p. 368). New York: Springer-Verlag Berlin Heidelberg.

Wigand, R. T., Markus, M. L. & Steinfield, C. W. (2005). Preface to the Focus Theme Section: "Vertical Industry Information Technology Standards and Standardization." *Electronic Markets*, *15*(4), 285–288. doi: 10.1080/10196780500302641

Yin, R. K. (2010). *Qualitative Research from Start to Finish* (pp. 1-348). New York: The Guilford Press.

Zhao, K., Xia, M. & Shaw, M. J. (2007). An integrated model of consortium-based e-business standardization: Collaborative development and adoption with network externalities. *Journal of Management Information Systems*, *23*(4), 247–271. doi:10.2753/MIS0742-1222230411

Zur Muehlen, M. & Indulska, M. (2010). Modeling languages for business processes and business rules: A representational analysis. *Information Systems*, *35*(4), 379–390. doi:10.1016/j.is.2009.02.006

Appendix A — Conceptual model of semantic IS standards

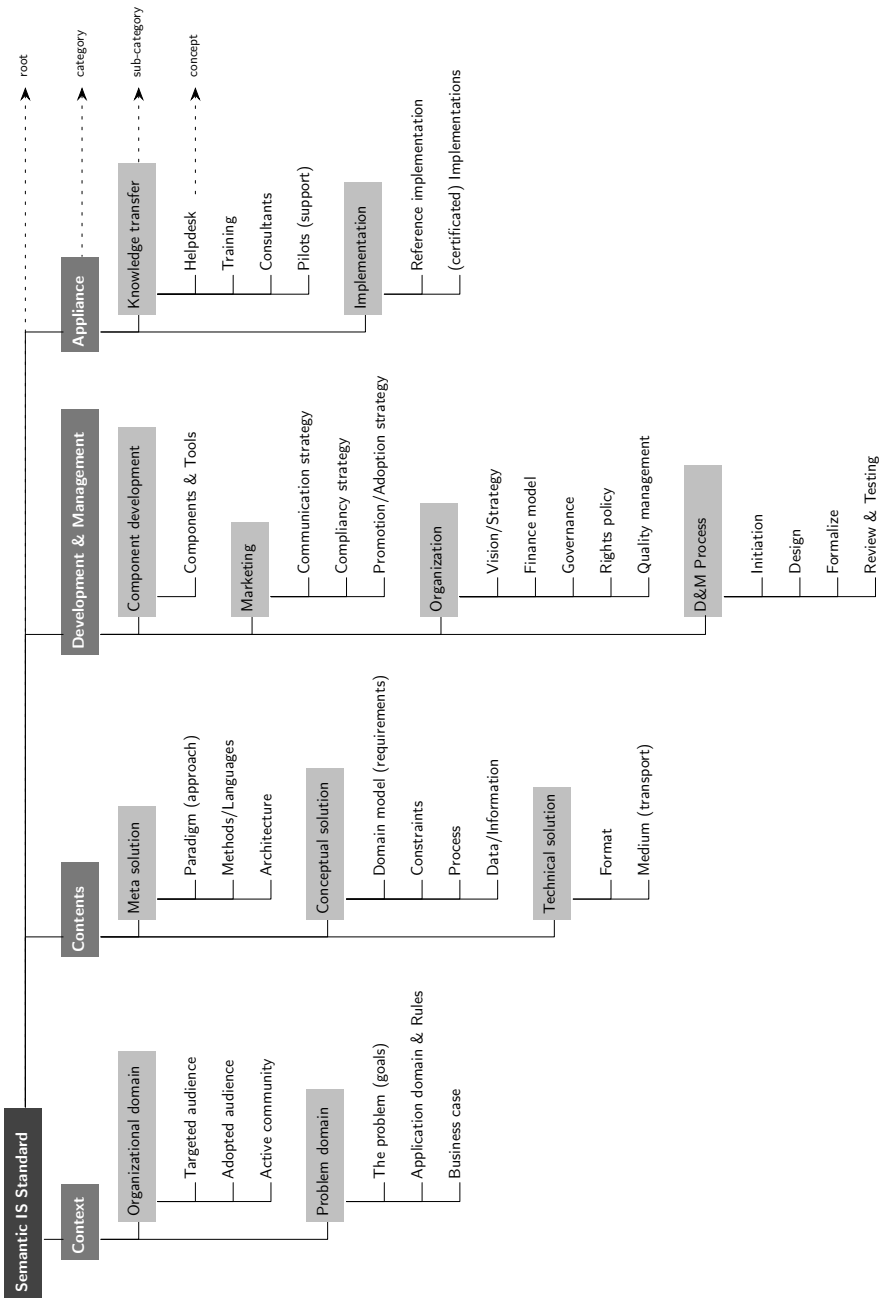


Figure 49. A conceptual model of semantic IS standards (adapted from Folmer et al., 2011).

Appendix B — The BRS rule classification scheme

Functional Category/ Sub-category	Common Name	Definition
1.0 Rejector	Constraint	Any rule that tends to disallow (that is, reject) an event if a violation of the rule would result. Rejectors shield the business from incorrect data (or incorrect state) — that is, from information that violates business rules. For example, a rejector might be specified to prevent a customer from placing an order on credit if the customer has a poor payment history.
2.0 Producer	—	Any rule that neither rejects nor projects events but simply computes or derives a value based on some mathematical function(s).
2.1 Computation rule	—	Any producer-type rule that computes a value following standard arithmetic operations (for example, sum, multiply, average, and so on) specified explicitly. A computation rule provides a precise formula for how a computed term is to be calculated. For example, a computation rule might be given to compute a customers' annual order volume.
2.2 Derivation rule	—	Any producer-type rule that derives a truth value (that is, true or false) based on logical operations (for example, AND, OR, NOT, EQUAL TO, and so on) specified explicitly. A derivation rule provides a precise definition for a derived term — that is, a truth-valued term whose value (true or false) is always established by the specified logical operations. For example, a derivation rule might be given to indicate whether a project is at risk depending on whether the project is over budget or understaffed.
3.0 Projector	Stimulus/ response rule	Any rule that tends to take some action (other than rejection) when a relevant event occurs. A projector never rejects events (as rejectors do); rather, it projects them — that is, causes some new event(s) to occur as a result. Projectors generally prescribe automatic system behavior, providing a productivity boost for workers. For example, a projector might be specified to reorder stock automatically if the quantity on hand drops below a certain point.
3.1 Enabler	Toggle	A projector that toggles something on or off.
3.1.1 Inference rule	—	An enabler that infers something to be true under appropriate circumstances. For example, an inference rule might be given to indicate that a person must be considered a woman if criteria for that person's age and gender are satisfied.
3.1.2 Rule toggle	Exception type rule	An enabler that turns another rule on or off under appropriate circumstances — that is, makes it capable or incapable of firing. For example, a rule toggle might be given to indicate that some normal operating rule is to be suspended under emergency circumstances.

3.1.3 Process toggle	—	An enabler that turns an operation, process, or procedure on or off under appropriate circumstances — that is, makes it capable or incapable of executing. For example, a process toggle might be given to indicate that a sensitive process cannot be executed while a security breach is suspected.
3.1.4 Data toggle	—	An enabler that creates or deletes instances of actual data under appropriate circumstances. For example, a data toggle might be given to indicate that a juvenile's criminal record must be erased when he or she reaches 18 years of age.
3.2 Copier	—	A projector that replicates (copies) actual values.
3.2.1 Imprint rule	—	A copier that sets the value of something that persists (for example, something in a database). For example, an imprint rule might be used to initialize the tuition owed by a student in a given semester to the base tuition for that semester when the student enrolls.
3.2.2 Presentation rule	—	A copier that establishes a value or parameter related to how data is to be presented (for example, on a screen, in a report, and so on). For example, a presentation rule might be given to indicate that an order is to be displayed on the screen in red if the order is overdue.
3.3 Executive	Trigger	A projector that causes an operation, process, or procedure to execute or a rule to fire.
3.3.1 Process trigger	—	A projector that causes an operation, process, or procedure to execute. For example, when an order is shipped, a process trigger might be given to execute a process that automatically sends the intended recipient a notification.
3.3.2 Rule trigger	—	A projector that causes a rule to fire. For example, when data about a shipment is displayed to the screen, a rule trigger might be given that fires another rule to predict the shipment's arrival date.

Table 12. The BRS Rule Classification Scheme (adapted from Ross, 2003: p.146-148)

Appendix C — RuleSpeak templates

Based on the basic sentence templates in RuleSpeak, this appendix provides a quick reference for rule-based template for semantic IS standards. Each rule category has one or more rule keywords, which appear in all capital letters. The simple syntactical conventions used in the table are explained in the following list.

- The symbols `< >` indicate the syntactical item inside is mandatory.
- The symbols `[ ]` indicate the syntactical item inside is optional.
- The symbol `/` indicates that only one syntactical item of the two or more listed need be selected.
- *Condition* always involves a logical expression. A condition is always based on one or more terms and facts (or data items) and may include logical operators such as `AND`, `OR`, and `NOT`.
- *Fact* inside brackets refers to the rest of the fact statement after the subject. For example, the `<fact>` for the statement "customer places order" is *places order*. Also, in all cases where `<fact>` appears, an embedded condition is permitted, e.g. "places more than ten orders" embeds the condition *more than ten*.
- Use of the keyword `should` in a rule statement indicates that the rule is a suggestor (i.e. a guideline, heuristic, or suggestion).

Rejector	
Keywords:	`MUST`, `ONLY`
Description:	A constraint for maintaining correctness (consistency) by preventing violations
Subject must be:	`term` or `fact` (`data item` allowed as well)

Template:	`<subject> MUST/should [not] <fact> [if/while <condition>]`
	`<subject> may/should [not] <fact> ONLY if/while <condition>`
	`<subject> may/should <fact> ONLY <preposition> <condition>`

Example:	An order must indicate the date it was received. A student must not take more than four courses while on probation. A salaried employee may work only in a budgeted department.

Table 13. RuleSpeak "rejector" template (adapted from Ross, 2003).

Permission statement	
Keywords:	`MAY`, `NEED NOT`
Description:	A policy or clarification permitting a business practice.
Subject must be:	`term`, `fact`, or `process` (`data item` allowed as well)
Template:	`<subject> MAY <fact/rule keyword> [if/while <condition>]` `<subject> NEED NOT <fact/rule keyword> [if/while <condition>]`
Example:	An order on credit totaling $1,000 or under may be accepted from a customer even if the customers credit has not been checked. A customer need not place any orders.

Table 14. RuleSpeak "permission statement" template (adapted from Ross, 2003).

Computation rule	
Keywords:	`BE COMPUTED`
Description:	A statement or arithmetic formula indicating how to calculate a numeric value
Subject must be:	`computed term` (`data item` allowed as well)
Template:	`<subject> must/should [not] BE COMPUTED as <mathematical formula> [if/while <condition>]`
Short:	`<subject> = <mathematical formula> [if/while <condition>]`
Example:	The amount paid for an order must be computed as the sum of all payment amounts applied to the order.
	The amount paid for an order = the sum of all payment amounts applied to the order.

Table 15. RuleSpeak "computation rule" template (adapted from Ross, 2003).

Derivation rule	
Keywords:	`BE TAKEN TO MEAN`, `MEANS`
Description:	A statement or logical expression indicating how to determine a yes/no (true/false) result.
Subject must be:	`derived term` (`data item` allowed as well)
Template:	`<Subject> must/should [not] BE TAKEN TO MEAN <logical expression> [if/while <condition>]`
Short:	`<Subject> MEANS [not] <logical expression> [if/while <condition>]`
Example:	Big-ticket item must be taken to mean the items cost exceeds $500.
	Big-ticket item means the items cost exceeds $500.

Table 16. RuleSpeak "derivation rule" template (adapted from Ross, 2003).

Inference rule	
Keywords:	`BE CONSIDERED`
Description:	A rule that infers a conclusion from a particular set of circumstances.
Subject must be:	`term` (`data item` allowed as well)
Template:	`<Subject> must/should [not] BE CONSIDERED [a] <term> if/while <condition>`
Short:	`<Subject> is [not] [a] <term> if/while <condition>`
Example:	A person must be considered a woman if the person is female and the persons age is 21 or over.
	A person is a woman if the person is female and the persons age is 21 or over.

Table 17. RuleSpeak "inference rule" template (adapted from Ross, 2003).

Rule toggle	
Keywords:	`UNLESS`, `EXCEPT`, `BE ENFORCED`
Description:	A rule that turns another rule on or off in a particular set of circumstances, especially for making exceptions.
Subject must be:	`rule`
Template:	`<Rule statement>, UNLESS/EXCEPT <condition>`
Formal:	`<Rule name> must/should [not] BE ENFORCED if/while <condition>`
Example:	A library card may be held by at most one borrower unless one of the borrowers who hold the library card is Bill Gates.
	The one-borrower-per-library-card rule must not be enforced if one of the borrowers who hold the library card is Bill Gates.

Table 18. RuleSpeak "rule toggle" template (adapted from Ross, 2003).

Process toggle	
Keywords:	`BE ENABLED`, `BE DISABLED`
Description:	A rule that turns a process on or off in a particular set of circumstances.
Subject must be:	`process` or `procedure`
Template:	`<Subject> must/should [not] BE ENABLED/DISABLED if/while <condition>`
Example:	Send-appointment-notice must be disabled if the client's address is unknown.

Table 19. RuleSpeak "process toggle" template (adapted from Ross, 2003).

Data toggle	
Keywords:	`BE CREATED`, `BE DELETED`
Description:	A rule that deletes data (or creates it randomly) in a particular set of circumstances.
Subject must be:	`data item`
Template:	`<Data item> must/should [not] BE CREATED/DELETED if/while <condition>`
Example:	Each outstanding case issue must be deleted when the case is closed.

Table 20. RuleSpeak "data toggle" template (adapted from Ross, 2003).

Imprint rule	
Keywords:	`BE SET`
Description:	A rule that sets a stored data item to a particular value.
Subject must be:	`term` or `fact` (`data item` allowed as well)
Template:	`<Term> must/should [not] BE SET to <term/value> [when/if <condition>]`
Example:	A students-semester-fees-owed must be set to $3,065 when the student registers for a semester.

Table 21. RuleSpeak "imprint rule" template (adapted from Ross, 2003).

Presentation rule	
Keywords:	`BE DISPLAYED`
Description:	A rule that sets a stored data item to a particular value.
Subject must be:	`term` or `fact` (`data item` allowed as well)
Template:	`<Subject> must/should [not] BE DISPLAYED [to/on/in <media>] <display manner> [if/while <condition>]`
Example:	An order must be displayed to the screen in red if the order is overdue.

Table 22. RuleSpeak "presentation rule" template (adapted from Ross, 2003).

Process trigger	
Keywords:	`BE EXECUTED`
Description:	A rule that automatically executes a process or procedure in a given set of circumstances.
Subject must be:	`process` or `procedure`
Template:	`<Subject> must/should BE EXECUTED when <condition>`
Example:	Send-advance-notice must be executed for an order when the order is shipped

Table 23. RuleSpeak "process trigger" template (adapted from Ross, 2003).

Rule trigger	
Keywords:	`BE FIRED`
Description:	A rule that automatically fires another rule in a given set of circumstances.
Subject must be:	`rule`
Template:	`<Rule name> must/should BE FIRED when <condition>`
Example:	The projected-shipment- date-rule must be fired when a shipment is displayed to the screen.

Table 24. RuleSpeak "rule trigger" template (adapted from Ross, 2003).

Appendix D — XMI files for the M2M example

Syntax (simplified):

```
<?xml version="1.0" encoding="ASCII"?>
<syntax:Root xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:syntax="syntax" xsi:schemaLocation="syntax D:/Docs/dev/SBVR-P/eclipse-km3-workspace/Syntax-SBVR-UML-usecase/Syntax/Syntax.ecore#/0">
  <elements xsi:type="syntax:Text" objectName="text" sentences="//@elements.1"/>
  <elements xsi:type="syntax:Sentence" objectName="EachCompanySellsAtLeastOneProduct" sentenceCat="//@elements.14" words="//@elements.2 //@elements.4 //@elements.6 //@elements.8 //@elements.10 //@elements.12"/>
  <elements xsi:type="syntax:Word" objectName="Each" sentence="//@elements.1" wordCat="//@elements.16" baseDesignation="//@elements.3"/>
  <elements xsi:type="syntax:Designation" objectName="Each" meaning="//@elements.24"/>
  <elements xsi:type="syntax:Word" objectName="Company" sentence="//@elements.1" wordCat="//@elements.17" baseDesignation="//@elements.5"/>
  <elements xsi:type="syntax:Designation" objectName="Company" meaning="//@elements.25"/>
  <elements xsi:type="syntax:Word" objectName="Sells" sentence="//@elements.1" wordCat="//@elements.19" baseDesignation="//@elements.7"/>
  <elements xsi:type="syntax:Designation" objectName="ToSell" meaning="//@elements.26"/>
  <elements xsi:type="syntax:Word" objectName="AtLeast" sentence="//@elements.1" wordCat="//@elements.21" baseDesignation="//@elements.9"/>
  <elements xsi:type="syntax:Designation" objectName="AtLeast" meaning="//@elements.27"/>
  <elements xsi:type="syntax:Word" objectName="One" sentence="//@elements.1" wordCat="//@elements.22" baseDesignation="//@elements.11"/>
  <elements xsi:type="syntax:Designation" objectName="1" meaning="//@elements.28"/>
  <elements xsi:type="syntax:Word" objectName="Product" sentence="//@elements.1" wordCat="//@elements.23" baseDesignation="//@elements.13"/>
  <elements xsi:type="syntax:Designation" objectName="Product" meaning="//@elements.29"/>
  <elements xsi:type="syntax:SentenceCat" objectName="EachCompanySellsAtLeastOneProduct" terminal="false" composedOf="//@elements.18 //@elements.15" sentence="//@elements.1"/>
  <elements xsi:type="syntax:NPCat" objectName="EachCompany" terminal="false" composedOf="//@elements.16 //@elements.17" composes="//@elements.14" sentence="//@elements.1" head="//@elements.17" determiner="//@elements.16" isSubjectOf="//@elements.19"/>
  <elements xsi:type="syntax:QUnvaluedCat" objectName="Each" terminal="true" composes="//@elements.15" sentence="//@elements.1" word="//@elements.2" expresses="//@elements.24" np="//@elements.15" object="//@elements.17"/>
  <elements xsi:type="syntax:NCat" objectName="Company" terminal="true" composes="//@elements.15" sentence="//@elements.1" word="//@elements.4" expresses="//@elements.25" np="//@elements.15"/>
  <elements xsi:type="syntax:VPCat" objectName="SellsAtLeastOneProduct" terminal="false" composedOf="//@elements.19 //@elements.20" composes="//@elements.14" sentence="//@elements.1" head="//@elements.19"/>
  <elements xsi:type="syntax:TVCat" objectName="Sells" terminal="true" composes="//@elements.18" sentence="//@elements.1" word="//@elements.6" expresses="//@elements.26" passive="false" subject="//@elements.15" vp="//@elements.18" directObject="//@elements.20"/>
  <elements xsi:type="syntax:NPCat" objectName="AtLeastOneProduct" terminal="false" composedOf="//@elements.21 //@elements.22 //@elements.23" composes="//@elements.18" sentence="//@elements.1" head="//@elements.23" determiner="//@elements.21" isDirectObjectOf="//@elements.19"/>
  <elements xsi:type="syntax:QValuedCat" objectName="AtLeast" terminal="true" composes="//@elements.20" sentence="//@elements.1" word="//@elements.8" expresses="//@elements.27" np="//@elements.20" object="//@elements.23" value="//@elements.22"/>
  <elements xsi:type="syntax:NumeralCat" objectName="One" terminal="true" composes="//@elements.20" sentence="//@elements.1" word="//@elements.10" expresses="//@elements.28"/>
  <elements xsi:type="syntax:NCat" objectName="Product" terminal="true" composes="//@elements.20" sentence="//@elements.1" word="//@elements.12" expresses="//@elements.29" np="//@elements.20"/>
  <elements xsi:type="syntax:UniversalQuantification" objectName="Each" expressedBy="//@elements.16"/>
  <elements xsi:type="syntax:ObjectType" objectName="Company" expressedBy="//@elements.17"/>
  <elements xsi:type="syntax:AssociativeFactType" objectName="ToSell" expressedBy="//@elements.19"/>
  <elements xsi:type="syntax:AtLeastNQuantification" objectName="AtLeast" expressedBy="//@elements.21"/>
  <elements xsi:type="syntax:NonNegativeInteger" objectName="One" expressedBy="//@elements.22"/>
  <elements xsi:type="syntax:ObjectType" objectName="Product" expressedBy="//@elements.23"/>
</syntax:Root>
```

SBVR (simplified):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:simplesbvr="simplesbvr">
  <simplesbvr:ObjectType objectName="Company" representations="/5"/>
  <simplesbvr:NonNegativeInteger objectName="One" representations="/11" value="1"/>
  <simplesbvr:ObjectType objectName="Product" representations="/13"/>
  <simplesbvr:Designation objectName="Each" meaning="/18" text="/4"/>
  <simplesbvr:Text objectName="Each" value="Each"/>
  <simplesbvr:Designation objectName="Company" meaning="/0" text="/6"/>
  <simplesbvr:Text objectName="Company" value="Company"/>
  <simplesbvr:Designation objectName="ToSell" meaning="/15" text="/8"/>
  <simplesbvr:Text objectName="ToSell" value="ToSell"/>
  <simplesbvr:Designation objectName="AtLeast" meaning="/19" text="/10"/>
  <simplesbvr:Text objectName="AtLeast" value="AtLeast"/>
  <simplesbvr:Designation objectName="1" meaning="/1" text="/12"/>
  <simplesbvr:Text objectName="1" value="1"/>
  <simplesbvr:Designation objectName="Product" meaning="/2" text="/14"/>
  <simplesbvr:Text objectName="Product" value="Product"/>
  <simplesbvr:AssociativeFactType objectName="ToSell" representations="/7" role1="/16" role2="/17"/>
  <simplesbvr:FactTypeRole nounConcept="/0"/>
  <simplesbvr:FactTypeRole nounConcept="/2"/>
  <simplesbvr:UniversalQuantification representations="/3" introducedVariable="/20" scopesOver="/21"/>
  <simplesbvr:AtLeastNQuantification representations="/9" introducedVariable="/24" scopesOver="/21"
minCardinality="/1"/>
  <simplesbvr:Variable rangesOver="/0"/>
  <simplesbvr:BinaryAtomicFormulation isBasedOn="/15" roleBinding1="/22" roleBinding2="/23"/>
  <simplesbvr:RoleBinding occursIn="/21" isOf="/16" bindsTo="/20"/>
  <simplesbvr:RoleBinding occursIn="/21" isOf="/17" bindsTo="/24"/>
  <simplesbvr:Variable rangesOver="/2"/>
</xmi:XMI>
```

UML (simplified):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:simpleuml="simpleuml">
  <simpleuml:Class name="Company" ownedAttribute="/5"/>
  <simpleuml:Class name="Product" ownedAttribute="/4"/>
  <simpleuml:PrimitiveType name="Integer"/>
  <simpleuml:Association name="ToSell" memberEnd="/5 /4"/>
  <simpleuml:Property name="Company" type="/0" classifier="/1" association="/3"/>
  <simpleuml:Property name="Product" type="/1" lowerValue="1" classifier="/0" association="/3"/>
</xmi:XMI>
```

Appendix E — XMI example from the SBVR standard

Consider the following example, which includes a small portion of a vocabulary and a rule statement.

```
company
```

```
officer
```

```
company appoints officer
```

```
EU-Rent
  General Concept:          company
```

```
EU-Rent must appoint at least 3 officers
```

The following figure is a UML instance diagram showing a MOF-based SBVR model of the example. For simplicity, only facts expressible in terms of the Meaning And Representation Vocabulary and the Logical Formulation of Semantics Vocabulary are shown. Some end names are elided where they are obvious from the class names or for '**thing1 is thing2**' (where it makes no difference). For elements of the vocabulary, the three layers of expression, representation, and meaning are apparent in the diagram. The rule, shown at the bottom, connects to the meanings of the elements of the vocabulary though its logical formulation.

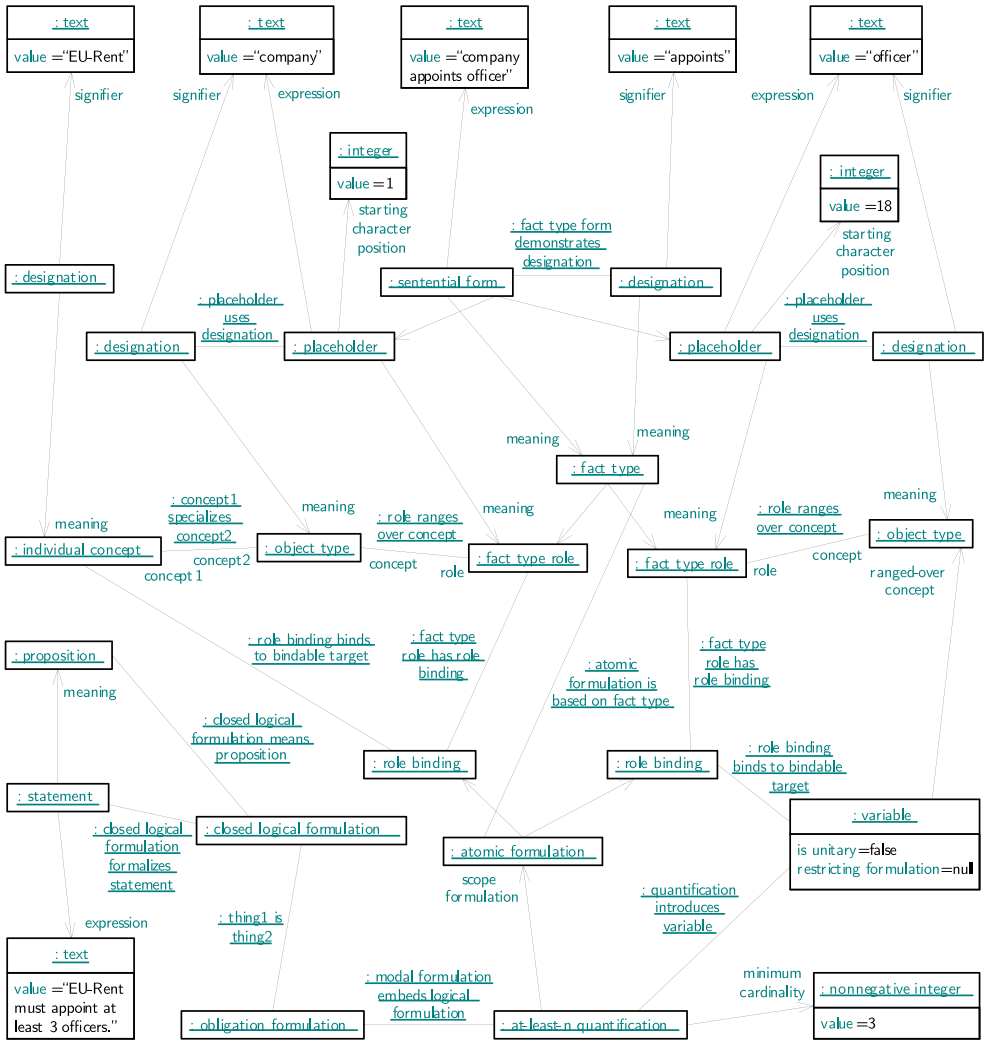


Figure 45. SBVR meta-model example Object Management Group, 2008).

The example MOF-based SBVR model is expressed below in XML based on the SBVR XML Schema. The xmi:id values are arbitrary and have no special meaning, but they build on the related signifiers to help readability. The XML tags, which include the namespace prefix '`sbvr`', are the XMI names for model elements of the SBVR Metamodel.

XMI header:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xmi:XMI xmi:version="2.1" xmlns:xmi="http://schema.omg.org/spec/XMI/2.1" xmlns:sbvr="http://www.omg.org/spec/SBVR/20070901/SBVR.xml">
```

XMI for company:

```
<sbvr:designation xmi:id="company" signifier="company-t" meaning="company-c"/>
<sbvr:objectType xmi:id="company-c"/>
<sbvr:text xmi:id="company-t" value="company"/>
```

XMI for officer:

```
<sbvr:designation xmi:id="officer" signifier="officer-t" meaning="officer-c"/>
<sbvr:objectType xmi:id="officer-c"/>
<sbvr:text xmi:id="officer-t" value="officer"/>
```

XMI for company *appoints* officer:

```
<sbvr:sententialForm xmi:id="companyAppointsOfficer" expression="cao-t" meaning="cao-c" placeholder="cao-p1 cao-p2"/>
<sbvr:factType xmi:id="cao-c" role="cao-r1 cao-r2"/>
<sbvr:factTypeFormDemonstratesDesignation factTypeForm="companyAppointsOfficer" designation="appoints"/>
<sbvr:designation xmi:id="appoints" signifier="appoints-t" meaning="cao-c"/>
<sbvr:text xmi:id="cao-t" value="company appoints officer"/>
<sbvr:text xmi:id="appoints-t" value="appoints"/>
<sbvr:placeholder xmi:id="cao-p1" expression="company-t" startingCharacterPosition="i1" meaning="cao-r1"/>
<sbvr:placeholderUsesDesignation placeholder="cao-p1" designation="company"/>
<sbvr:concept1SpecializesConcept2 concept1="cao-r1" concept2="company-c"/>
<sbvr:factTypeRole xmi:id="cao-r1"/>
<sbvr:positiveInteger xmi:id="i1" value="1"/>
<sbvr:placeholder xmi:id="cao-p2" expression="officer-t" startingCharacterPosition="i18" meaning="cao-r2"/>
<sbvr:placeholderUsesDesignation placeholder="cao-p2" designation="officer"/>
<sbvr:concept1SpecializesConcept2 concept1="cao-r2" concept2="officer-c"/>
<sbvr:factTypeRole xmi:id="cao-r2"/>
<sbvr:positiveInteger xmi:id="i18" value="18"/>
```

XMI for EU-RENT with `General Concept:` company:

```
<sbvr:designation xmi:id="EU-Rent" signifier="EU-Rent-t" meaning="EU-Rent-c"/>
<sbvr:individualConcept xmi:id="EU-Rent-c"/>
<sbvr:text xmi:id="EU-Rent-t" value="EU-Rent"/>
<sbvr:concept1SpecializesConcept2 concept1="EU-Rent-c" concept2="company-c"/>
```

XMI for EU-RENT must *appoint* at least 3 officers:

```
<sbvr:statement xmi:id="stmt" expression="stmt-t" meaning="stmt-p"/>
<sbvr:text xmi:id="stmt-t" value="EU-Rent must appoint at least 3 officers."/>
<sbvr:proposition xmi:id="stmt-p"/>
<sbvr:closedLogicalFormulationFormalizesStatement closedLogicalFormulation="ob2" statement="stmt"/>
<sbvr:closedLogicalFormulationMeansProposition closedLogicalFormulation="ob2" proposition="stmt-p"/>
<sbvr:obligationFormulation xmi:id="ob"/>
<sbvr:closedLogicalFormulation xmi:id="ob2"/>
<sbvr:thing1IsThing2 thing1="ob" thing2="ob2"/>
<sbvr:modalFormulationEmbedsLogicalFormulation modalFormulation="ob" logicalFormulation="am3"/> <sbvr:at-
least-nQuantification xmi:id="am3" scopeFormulation="atom" minimumCardinality="i3"/>
<sbvr:quantificationIntroducesVariable quantification="am3" variable="v"/>
<sbvr:variable xmi:id="v" ranged-overConcept="officer-c" restrictingFormulation="" isUnitary="false"/>
<sbvr:atomicFormulation xmi:id="atom" roleBinding="bind1 bind2"/>
<sbvr:atomicFormulationIsBasedOnFactType atomicFormulation="atom" factType="cao-c"/> <sbvr:roleBinding
xmi:id="bind1"/>
<sbvr:roleBindingBindsToBindableTarget roleBinding="bind1" bindableTarget="EU-Rent-c"/>
<sbvr:factTypeRoleHasRoleBinding factTypeRole="cao-r1" roleBinding="bind1"/>
<sbvr:roleBinding xmi:id="bind2"/>
<sbvr:roleBindingBindsToBindableTarget roleBinding="bind2" bindableTarget="v"/>
<sbvr:factTypeRoleHasRoleBinding factTypeRole="cao-r2" roleBinding="bind2"/>
<sbvr:positiveInteger xmi:id="i3" value="3"/>
</xmi:XMI>
```

Appendix F — SETU business vocabulary

assignment

Concept Type:	object type

correctionTimecard

Definition:	a timecard that *has* a status *of* timecard that *is* corrected
Concept Type:	timecard
Synonymous Form:	corrected timecard

expenseAllowance

Definition:	a container that can *have* expense information and allowance information for a specific period
Concept Type:	object type
Synonyms:	expenses, allowances

human resource

Definition:	a party that *works* for another staffing company
Concept Type:	object type
General Concept:	party
Note:	Original definition: *The person performing the activities that are reported on the Timecard.*

invoice

Concept Type:	object type

party

Definition:	a person or group *participating* in a transaction
Concept Type:	object type
Dictionary Basis:	a person or group participating in an action or affair — *a mountain-climbing party, a party to the transaction.*

receiver

Definition:	a party that can *receive* a timecard
Concept Type:	noun concept
General Concept:	party
Note:	Original definition: *The person, organization or business entity responsible for receiving the Timecard.*

recorder

Definition:	a party that can *record* time information
Concept Type:	noun concept
General Concept:	party

reportedTime

Definition:	a container that *has* time information for a specific period
Concept Type:	object type
Synonyms:	reported time

shop floor recording system

Definition:	a party that *records* time information at the staffing consumer location
Concept Type:	object type
General Concept:	party

staffing company

Definition:	a party that *matches* human resources to staffing customers
Concept Type:	object type
General Concept:	party
Dictionary basis:	An employment agency is an organization which matches employers to employees — *an advertising agency, an employment agency.*

staffing customer

Definition:	a party that *hires* human resources from staffing companies
Concept Type:	object type
General Concept:	party

submitter

Definition:	a party that can *send* a timecard
Concept Type:	noun concept
General Concept:	party
Note:	Original definition: *The person, organization or business entity responsible for submitting the Timecard.*

timecard

Definition:	an object that *contains* timecardLines
Concept Type:	object type

timecardLine

Definition:	a container that can *have* reportedTime or expenseAllowance for a human resource for a specific period
Concept Type:	object type
Synonymous Form:	timecard line

Appendix G — SETU structure rules

Timecard — Attributes (the same for correctionTimecard)

identifier Concept Type: attribute	timecard has identifier Concept Type: is-property-of fact type Synonymous Form: identifier of timecard Necessity: timecard must have one identifier
type Concept Type: attribute	timecard has type Concept Type: is-property-of fact type Synonymous Form: type of timecard Necessity: timecard can have at most one type
status Concept Type: attribute	timecard has status Concept Type: is-property-of fact type Synonymous Form: status of timecard Necessity timecard can have at most one status
reference Concept Type: attribute	timecard has reference Concept Type: is-property-of fact type Synonymous Form: reference of timecard Necessity: timecard must have reference
actionCode Concept Type: attribute	timecard has actionCode Concept Type: is-property-of fact type Synonymous Form: actionCode of timecard Necessity: timecard can have at most one actionCode
resourceInfo Concept Type: attribute	timecard has resourceInfo Concept Type: is-property-of fact type Synonymous Form: resourceInfo of timecard Necessity: timecard must have one resourceInfo
approvalInfo Concept Type: attribute	timecard has approvalInfo Concept Type: is-property-of fact type Synonymous Form: approvalInfo of timecard Necessity: timecard can have approvalInfo
submitterInfo Concept Type: attribute	timecard has submitterInfo Concept Type: is-property-of fact type Synonymous Form: submitterInfo of timecard Necessity: timecard can have at most one submitterInfo
reportElement Concept Type: attribute	timecard has reportElement Concept Type: is-property-of fact type Synonymous Form: reportElement of timecard Necessity: timecard can have reportElement
comment Concept Type: attribute	timecard has comment Concept Type: is-property-of fact type Synonymous Form: comment of timecard

Timecard — Structural rules

timecard must have one human resource

timecard must have one submitter

```
timecard must have at least one receiver
timecard can have assignment
timecard has at least one timecardLine
timecard can have invoice
```

Timecard — Characteristics

```
identifier is unique
  Definition:                an identifier of timecard that is ...
  Concept Type:              characteristic
```

TimecardLine — Attributes

```
reportedActivity
  Concept Type:              attribute

timecardLine has reportedActivity
  Concept Type:              is-property-of fact type
  Synonymous Form:           reportedActivity of timecardLine
```

```
reportedElement
  Concept Type:              attribute

timecardLine has reportedElement
  Concept Type:              is-property-of fact type
  Synonymous Form:           reportedElement of timecardLine
```

ReportedTime — Attributes

```
identifier
  Concept Type:              attribute

reportedTime has identifier
  Concept Type:              is-property-of fact type
  Synonymous Form:           identifier of reportedTime
  Necessity:                 reportedTime can have at most one
                             identifier
```

```
type
  Concept Type:              attribute

reportedTime has type
  Concept Type:              is-property-of fact type
  Synonymous Form:           type of reportedTime
  Necessity:                 reportedTime must have one type
```

```
actionCode
  Concept Type:              attribute

reportedTime has actionCode
  Concept Type:              is-property-of fact type
  Synonymous Form:           actionCode of reportedTime
  Necessity:                 reportedTime can have at most one
                             actionCode
```

```
dateTimeInfo
  Concept Type:              attribute

reportedTime has dateTimeInfo
  Concept Type:              is-property-of fact type
  Synonymous Form:           dateTimeInfo of reportedTime
  Necessity:                 reportedTime can have at most one
                             dateTimeInfo
```

```
rate
  Concept Type:              attribute

reportedTime has rate
  Concept Type:              fact type
  Synonymous Form:           rate of reportedTime
  Necessity:                 reportedTime can have rate
```

```
comment
  Concept Type:              attribute

reportedTime has comment
  Concept Type:              fact type
  Synonymous Form:           comment of reportedTime
  Necessity:                 reportedTime can have comment
```

ReportedTime — Structural rules

reportedTime must *have* one timecardLine

ExpenseAllowance — Attributes

identifier Concept Type: attribute	expenseAllowance *has* identifier Concept Type: is-property-of fact type Synonymous Form: identifier *of* expenseAllowance Necessity: expenseAllowance can *have* at most one identifier
type Concept Type: attribute	expenseAllowance *has* type Concept Type: is-property-of fact type Synonymous Form: type *of* expenseAllowance Necessity: expenseAllowance must *have* one type
expenseOrAllowance Concept Type: attribute	expenseAllowance *has* expenseOrAllowance Concept Type: is-property-of fact type Synonymous Form: expenseOrAllowance *of* expenseAllowance Necessity: expenseAllowance must *have* one expenseOrAllowance
expenseOrAllowance Concept Type: attribute	expenseAllowance *has* expenseOrAllowance Concept Type: is-property-of fact type Synonymous Form: expenseOrAllowance *of* expenseAllowance Necessity: expenseAllowance can *have* at most one actionCode
actionCode Concept Type: attribute	expenseAllowance *has* actionCode Concept Type: is-property-of fact type Synonymous Form: actionCode *of* expenseAllowance Necessity: expenseAllowance can *have* at most one actionCode
dateTimeInfo Concept Type: attribute	expenseAllowance *has* dateTimeInfo Concept Type: is-property-of fact type Synonymous Form: dateTimeInfo *of* expenseAllowance Necessity: expenseAllowance can *have* at most one dateTimeInfo
amount Concept Type: attribute	expenseAllowance *has* amount Concept Type: is-property-of fact type Synonymous Form: amount *of* expenseAllowance Necessity: expenseAllowance must *have* one or two amount
quantity Concept Type: attribute	expenseAllowance *has* quantity Concept Type: is-property-of fact type Synonymous Form: quantity *of* expenseAllowance Necessity: expenseAllowance can *have* at most one quantity
comment Concept Type: attribute	comment *has* comment Concept Type: is-property-of fact type Synonymous Form: comment *of* expenseAllowance Necessity: expenseAllowance can *have* comment

ExpenseAllowance — Structural rules

```
expenseAllowance must have one timecardLine
```

‘Parties & Roles’ — Structural rules

```
All of the following are always true for a submitter:
  • It can record time information
  • It can validate timecard
  • It can send timecard
```

```
All of the following are always true for a receiver:
  • It can validate timecard
  • It can receive timecard
```

```
All of the following are always true for a submitter:
  • It can record time information
```

```
All of the following are always true for a shop floor recording system:
  • It can be a recorder
  • It can be a submitter
```

```
All of the following are always true for a staffing customer:
  • It can be a submitter
  • It can be a receiver
  • It can be a recorder
```

```
All of the following are always true for a staffing company:
  • It can be a submitter
  • It can be a receiver
  • It can be a recorder
```

Appendix H — SETU power rules

Please note that these rules are based on Stornebrink's framework, which has been proven to work, but there is no commercial or open source software to implement RBPM.

collect-information	
Pre-condition:	-
Boundary-condition:	-
Post-condition:	reportedTime *is available* expenseAllowance *is available*

create-timecard	
Pre-condition:	create-timecard may be executed when any of the following is true: • reportedTime *is available* • expenseAllowance *is available*
Boundary-condition:	-
Post-condition:	timecard must *be created* status *of* timecard must be set to unconfirmed

correct-timecard	
Pre-condition:	correct-timecard may be executed when result *is available* and when identifier *of* timecard *is known*
Boundary-condition:	-
Post-condition:	correct-timecard may be disabled when all of the following are true: • correctionTimecard must be created • identifier *of* correctionTimecard must be set to identifier *of* timecard • status *of* correctionTimecard must be set to unconfirmed

validate-timecard	
Pre-condition:	validate-timecard may be executed when timecard *is created*
Boundary-condition:	-
Post-condition:	
Note:	It is not clear when data is valid, therefore there is no postcondition defined.

exchange-timecard	
Pre-condition:	exchange-timecard may be executed when timecard *is available*
Boundary-condition:	-
Post-condition:	timecard *is sent* to receiver
Note:	This process is not modeled in the SETU standard, it is part of the validate process.

accept-timecard	
Pre-condition:	accept-timecard may be executed when timecard *is available*
Boundary-condition:	-
Post-condition:	status *of* timecard must be set to accepted
Note:	This process is not modeled in the SETU standard, it is part of the validate process.

reject-timecard	
Pre-condition:	reject-timecard may be executed when timecard *is available*
Boundary-condition:	-
Post-condition:	status *of* timecard must be set to rejected
Note:	This process is not modeled in the SETU standard, it is part of the validate process.

create-result	
Pre-condition:	create-result may be executed when status *of* timecard *is* rejected
Boundary-condition:	-
Post-condition:	result must be created
Note:	Result should be defined in the vocabulary, maybe in less ambiguous terms e.g. 'validation result'. There one can define that a validation result must have a 'reason for rejection'.

exchange-result	
Pre-condition:	exchange-result may be executed when result *is available*
Boundary-condition:	-
Post-condition:	result *is sent* to receiver
Note:	This process is not modeled in the SETU standard, it is part of the create result process.

Appendix I — SETU control rules

Below we find the four business rules from the SETU standard.

```
identifier of timecard is unique
```

```
identifier is unique when identifier of timecard = customer number
```

```
elaborate information should not be in resourceInfo of timecard
```

```
dateTimeInfo
  Definition:        a period
  Note:              Period is defined as: "A time interval measured from a start date/time to an end date/
                     time" (Object Management Group, 2008).
```

The following business rules are there to create artificial flow.

```
an unconfirmed timecard must be either accepted or rejected
```

```
a validated timecard must be sent to a receiver
```

```
a timecard must be validated
```

```
a rejected timecard must have a result
```

```
a rejected timecard must be sent to a receiver
```

Appendix J — SETU standard attribute descriptions

Attribute	Description
Identifier	*Identifier is mandatory. A single instance is required.* The identifier of the timecard. This identifier should be unique within the scope of each organization that specifies an identifier.
Type	*Type is optional. A single instance might exist.* The type of timecard.
Status	*Status is optional. A single instance might exist.* The status of the Timecard as a whole.
Reference	*Reference is mandatory. Multiple instances might exist.* References to other documents or parties. This may be a reference to an Assignment, another Timecard or a Staffing customer.
ActionCode	*ActionCode is optional. A single instance might exist.* A code specifying the action to be performed on a the TimeCard.
ResourceInfo	*ResourceInfo is mandatory. A single instance is required.* Information on the resource performing the reported activities. It includes a limited amount of details and an optional reference to a record with a more detailed description.
SubmitterInfo	*SubmitterInfo is optional. One single instance might exist.* The details of the party responsible for submitting the Timecard and the timestamp of submission.
ApprovalInfo	*ApprovalInfo is optional. Multiple instances might exist.* The details of the party responsible for approving the Timecard and the timestamp of approval.
Reportelement	*Reportelement is optional. Multiple instances might exist.* A container for a variety of reporting elements.

Table 25. Attributes of Timecard (adapted from Folmer et al., 2009).

Attribute	Description
ReportedActivity	*ReportedActivity is optional. A single instance might exist.* A brief description of the activities conducted during the period registered. It may include a reference to a more detailed description.
Reportelement	*Reportelement is optional. Multiple instances might exist.* A container for a variety of reporting elements.

Table 26. Attributes of TimecardLine (adapted from Folmer et al., 2009).

Attribute	Description
Identifier	*Identifier is optional. A single instance might exist.* The identifier of the reported time entry. This identifier should be unique in combination with the timecard identifier.
Type	*Type is mandatory. A single instance is required.* The type of timeframe or time event that is registered.
ActionCode	*ActionCode is optional. A single instance might exist.* A code specifying the action to be performed on a ReportedTime item.
DateTimeInfo	*DateTimeInfo is optional. A single instance might exist.* The details of the timeframe or time that is registered.
Rate	*Rate is optional. Multiple instances might exist.* Description of the (agreed) rate against which activities are conducted. Rates may include those for billing or for payrolling.
Comment	*Comment is optional. Multiple instances might exist.* General (textual) comments on a reported time entry item.

Table 27. Attributes of ReportedTime (adapted from Folmer et al., 2009).

Attribute	Description
Identifier	*Identifier is optional. A single instance might exist.* The identifier of the expense/allowance entry. This identifier should be unique in combination with the timecard identifier.
Type	*Type is mandatory.* A single instance is required. The type of expense or allowance that is registered.
ExpenseOrAllowance	*ExpenseOrAllowance is mandatory. A single instance is required.* Indication of whether the item is an expense or an allowance.
ActionCode	*ActionCode is optional. A single instance might exist.* A code specifying the action to be performed on an ExpenseAllowance item.
DateTimeInfo	*DateTimeInfo is optional. A single instance might exist.* The details of the timeframe or time the expense or allowance is incurred.
Amount	*Amount is mandatory. Up to two instances might exist.* The value of the expense or allowance. Amounts may be specified for the same expense/ allowance both for invoicing and/or for payrolling.
Quantity	*Quantity is optional. A single instance might exist.* A numerical quantity that is assigned or determined by calculation or measurement.
Comment	*Comment is optional. Multiple instances might exist.* General (textual) comments on a particular expense item.

Table 28. Attributes of ExpenseAllowance (adapted from Folmer et al., 2009).

Appendix K — Survey results

The survey results have been summarized per question. The respondents will stay anonymous.

Do you think that a business rules approach to semantic IS standards development, could/will lead to:

1. Accuracy increase?

Result: 50% agrees

Two our of four respondents have answered this question and motivated their choice. Both respondents disagreed that business rules don't per se increases the *accuracy* of semantic IS standards. However any method (based on a business rule approach) can have a positive influence on accuracy.

2. Completeness increase?

Result: 75% agrees

All respondents have answered this question and motivated their choice. Most respondents agreed that business rules force some form of completeness for defining structure. Due to the format of SBVR documents (XMI), checks and validations can be done fairly easy to check for completeness. However one respondent argued that the way of modeling (i.e. using a business rule approach) does not influence completeness.

3. Consistency increase?

Result: 100% agrees

All respondents have answered this question and motivated their choice. All respondents agree that consistency is increased by using a business rule approach to semantic IS standards development. A business rule approach forces the developer to think about the definition of objects. Formal expressions (like 'Structured English' and RuleSpeak) make consistency checks also possible. One respondent noted that for a business rule approach to be effective, an automated environment needs to be used to define rules and relationships.

4. Compliance increase?

Result: 100% agrees

Three of four respondents have answered this question and motivated their choice, one did not provide an answer nor motivation. They agreed that compliancy would work for legislation if the same terminology is used. The same as the third question, an automated environment is necessary to increase compliancy.

5. Precision increase?

Result: 100% agrees

Three of four respondents have answered this question and motivated their choice, one did not provide an answer nor motivation and one did not provide an answer. Two respondents agreed that a business rule approach forces the developer to think about the definition of objects. One respondent thought that it relates too much to accuracy, so he couldn't answer this question.

6. Traceability increase?

Result: 50% agrees

All respondents have answered this question and motivated their choice. Respondents were divided on this quality attribute. Some though it was not a relevant attribute, while other stated that with proper tools, the source of a rule would become clear. However this is not a property that only business rules have.

7. Understandability increase?

Result: 25% agrees

All respondents have answered this question and motivated their choice. Even though a rule modeling language like 'Structured English' will be understandable to (business) people, even more so than UML, it will total decrease overview, which makes models necessary.

8. Portability increase?

Result: 0% agrees

Three of four respondents have answered this question and motivated their choice, one did not provide an answer but only commented. All respondents agreed that portability is not relevant for business rules or that a business rule approach does not increase portability.