Specifying business processes by means of rules*

S. Joosten[1,2] and R. Joosten[3]

May 18, 2005

[1] Ordina BusinessWorks
Postbus 7101
3430 JC Nieuwegein
The Netherlands

[2] Open University of the Netherlands
Postbus 2960
6401 DL Heerlen
The Netherlands

[3] TNO Informatie- en Communicatie Technologie
Postbus 15000
9700 CD Grongingen
The Netherlands

Abstract

Business rules are useful for various purposes. Usually, business rules are the means to communicate requirements with user communities (e.g. 'All tasks must be performed by staff that is both qualified and authorized for that task'). However, business rules can equally well be used as software specifications for controlling business processes. In this paper we propose to use business rules to specify both business processes and the software that supports them. This paper also proposes a repository for business rules (RAP) that not only stores and organizes rules, but serves as a process engine as well. This result is achieved by making use of smart mathematical notations, which bring a requirements engineer from his business interviews straight to a consistent design.

Keywords: Business Rules, Business Process Management, Formal Specifications.

ISRL Categories: CB0901.03 modeling languages, etc.

*Contribution to the European Business Rules Conference, Amsterdam, June 2005.

1 Introduction

Business rules can be used directly to manage and control business processes. This principle has proven itself in practice by defining ontologies, designing software, solving architectural issues in large projects, enforcing consistency in practical problems, and monitoring violations of architectures. Proofs of concept have been conducted both in the financial domain, i.e. securities and stocks, credits and loans, and insurances, and in the software domain, i.e. security, infrastructure, and methodology engineering.

Our approach yields cost reductions, better specifications, and increased certainty that business requirements are met. Cost is reduced by avoiding much of the conventional modeling activity. Specifications are improved because they are closer to the business, provably consistent and buildable. Increased certainty is obtained by a mechanism to detect violations of business rules systematically. Besides these demonstrated benefits, business rules promise increased flexibility in business process management, governance support, flawless linking of systems and processes, and support for application maintenance.

Business rules [22] are as close to the business as we can get. For example, a life insurance company might have a business rule saying that applications for a new pension plan are decided upon within three days. A business rule in the context of immigration might be that applications for green cards are put aside if the identity of applicants cannot be established legally. A medical benefits administrator might wish to prevent fraud by requiring that payments be authorized by two different staff, both of whom are authorized for the full amount paid (4-eyes principle). In fact, business rules do for an organization what legislature does for a nation. That is why organizations have scores of business rules, addressing employees and other stakeholders in their own language, and providing criteria for the assessment of their day-to-day work. In this paper we show that business rules can be used directly for controlling business processes, without reference to a business process model.

In order to make rules concrete, we have limited the idea of a business rule to the ones that can be violated. Rule based business process management (BPM) employs these violations to trigger business processes. The principle of signaling violations and subsequently resolving them constitutes the engine cycle of a rule based process engine. This yields a principle for BPM, in which business rules are used to control business processes directly.

Our work brings solutions to complexity issues of contemporary organisations. The number of employees, bureaucratic procedures, the number of products and their variations, the number of business processes, the amount

of relevant documents, the departmental politics within the organization, and everything else that is relevant for that particular business produce very complex situations in many organisations. Whether we work with business process management software, with regular audits, or with campaigns to communicate the rules, applying the rules systematically is often found difficult and time consuming. We observed that the complexity is not in the rules, but in the sheer numbers. A business rule is not complex by itself, but the number of places in the organization where a rule applies may be overwhelming. For example, the 4-eyes rule for payments is not difficult to understand or to live by. But how do you ensure in practice that all payments are authorized by two suitable employees, when a majority of all business processes involves payments from and to various accounts by many different employees in different departments with different ideas about what constitutes a 'suitable employee'? And once you have achieved that, what about enforcing all other business rules? The essence of the problem lies in the sheer amount of detail, which is not seldom too large for any human design effort. For this reason automated design support is imperative.

In our research we have explored a solution for this problem. The part of the solution that can be automated solves the complexity issue by using relational algebra [24], which has allowed us to generate specifications directly from business rules. The remaining work consists of drawing up business rules and managing them. This allows designers to stay close to the business. We have found that the dialogue between designers and business representatives is easier and more productive as they concentrate on business rules rather than on process models, use case scenarios, class diagrams and other (indirect) design artefacts. Our research has demonstrated that business rules alone can serve as functional business requirements.

Our research set out originally to operationalize business rules. We found we could represent business rules directly into relational algebra, a formalism well known for over a century [6, 21, 25] and familiar to computer professionals. This has resulted in a language for representing business rules called ADL (A Description Language). The quest to make it usable for practitioners has yielded the CC-method [8, 17], a way to draw up a consistent set of business rules, which can be validated by the business. We have also developed a tool that manipulates business rules, checks whether they form a coherent set of rules, computes violations of these rules and generates specifications for database design. It is used to signal violations throughout the scope in which business rules are valid. In this paper, we argue that these violations can be used as triggers to control business processes.

2 Control Principle

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, which is often attributed to Deming [26]. An example:

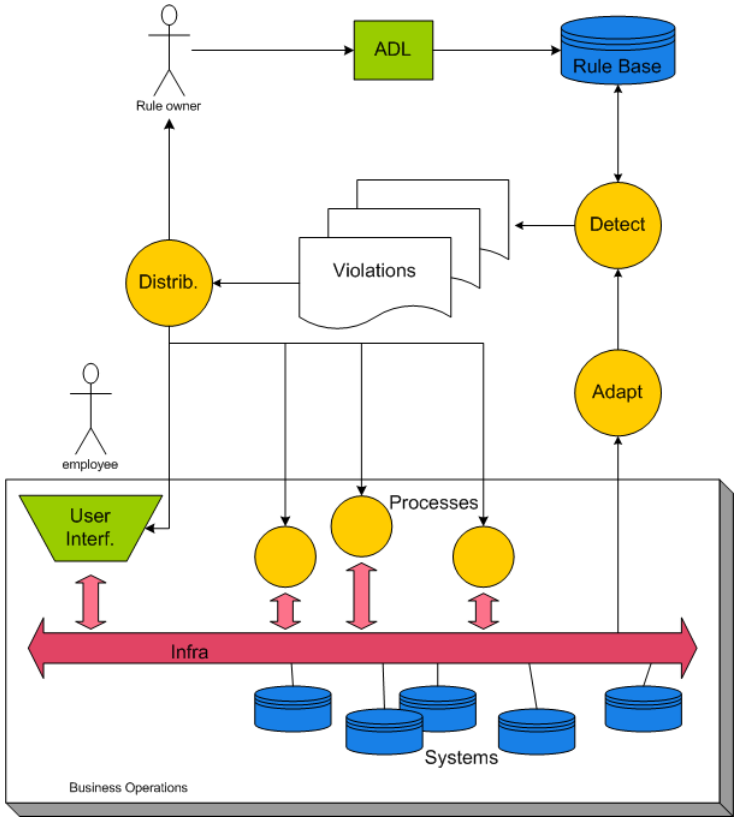

Figure 1: Principle of rule based process management

Suppose we have a rule that requires every invoice to be either paid or returned. The arrival of an invoice violates this rule, because we now have an invoice that is neither paid nor returned to sender. This violation is then used to trigger either a person or a computer to deal with the invoice. Either way, once the invoice is dealt with, the violation ceases to exist. So both human and automated actions are taken, all of which are directed towards resolving business rule violations.

Figure 1 illustrates the principle. Let the business operations consist of an arbitrary collection of people, machines, infrastructure, etcetera. An adapter observes the business by drawing information from it. That information can come from any available source, such as a data warehouse, interaction with

users, or interaction with information systems. The observations are fed to the violation detector, which checks them against business rules in a rule base. Resulting violations are fed to a process engine, which distributes appropriate triggers to people and computers.

This principle provides the maximum possible flexibility, because employees can do anything in any way and order, as long as they bring violations closer to being resolved. These actions can cause new violations, causing subsequent actions, and so on until the process is completed. For instance, once a payment is launched by one employee, a violation might occur that is recognized as a violation of the 4-eyes principle. The process engine can now trigger another employee who checks the payment and releases it definitively. Without that 4-eyes rule, the payment would have been released straight away.

You might object that the employee can simply return all invoices, and be done with all the work before dawn breaks. This would have been all right, if it weren't for a rule that requires all legitimate invoices to be paid. That rule forces to assess whether an invoice is legitimate and subsequently pay it. So whenever a process is not going the way it should, we need to add business rules or replace faulty ones. For this purpose, it is necessary to discuss and scrutinize business rules with their owners. In such cases, being able to conduct discussions with the business on the basis of business rules is highly valued. If the process works properly, there is no need to restrict employees and computers any further.

But now, you might add, what if there is no rule to distinguish between legitimate and illegitimate invoices? In that case, we might ask the user. If the user doesn't know either, the process is genuinely stuck. In that case, someone in the organization will have to tell what to do. That person is the process owner, and he is responsible for having the right rules in place. A set of rules is considered complete if every case that might occur can be dealt with, either by human action, by computer activity or by a combination of both. So rules do not have to describe every possible situation, but may rely on human judgment to fill in the blanks. The set is nevertheless considered complete if all cases can be dealt with without getting stuck.

Having discussed the principle of controlling a business process with rules, let us take a more detailed look into the heart of business processes: the process engine.

3 Process Engine

A process engine that supports rule based BPM differs from a conventional workflow engine, because it refers directly to business rules for knowing

which actions to take. Figure 2 shows how it works.

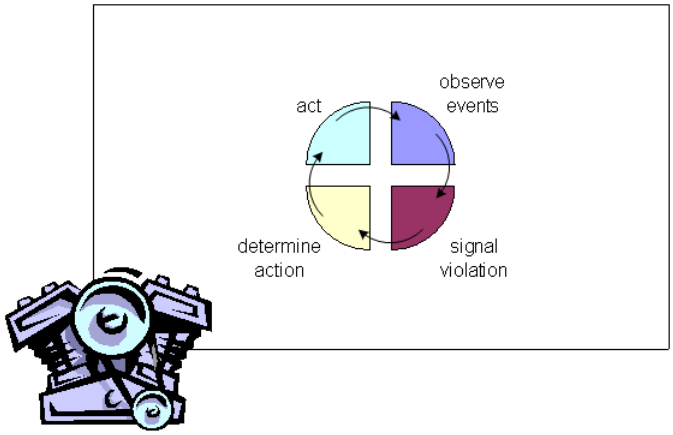

Figure 2: Engine cycle for a rule based process engine

Let us discuss the engine cycle from the point where violations are signalled. All events that are made available to the engine are analyzed to see whether violations have occurred. The logic to detect violations is derived from the business rules. This is possible because the business rules are represented in a suitable mathematical form. In order to obtain systematic and perpetual detection of violations, the analysis is performed dynamically by a computer. Every time the cycle is traversed the engine may detect new violations.

Each violation is traced back to a specific business rule. This is necessary to determine an action to resolve the violation. The action can be either automated or it might involve activity from human actors. The design tool may point out several alternatives for resolving a violation. In that case an intervention by the designer is needed to determine, which alternatives can already be discarded at design time, which alternatives can be selected during the design, and which alternatives can be passed on to the user to select from. In the last case, the user is given a choice the moment the violation is detected. Routing activities to an appropriate user is also done in this step. Needless to say that the choice of a user (i.e. the decision where a step is routed to) is also made based on rules.

Performing the action boils down to either calling a service (which is done by the engine) or invoking a dialogue with the user. The user experiences a dialogue in which there is no predefined order of activities. Activities are invoked only when necessary, in the order which is most appropriate for resolving violations.

Once the action is finished, the entire system will be in a new state. The

events, produced by the actions taken, may lead to effects in various parts of the infrastructure. These events are then detected by the signaling functionality and that is where the cycle is closed. By traversing this cycle perpetually, all activities can be triggered in time.

The power of the process engine can be measured as its capacity to deal with a certain number of events per unit of time. If the number of events peaks beyond the power of the process engine, it may miss events, delaying them to subsequent cycles. So peak loads lead to small delays in the treatment of events, very much like transactions do in TP-monitors on mainframe computers. If this occurs too often, the engine needs to be replaced by one that has more power.

4 Design Process

Rule based BPM fosters a much simplified design process, which is depicted in figure 3. At the centre of the action you will find the dialogue between

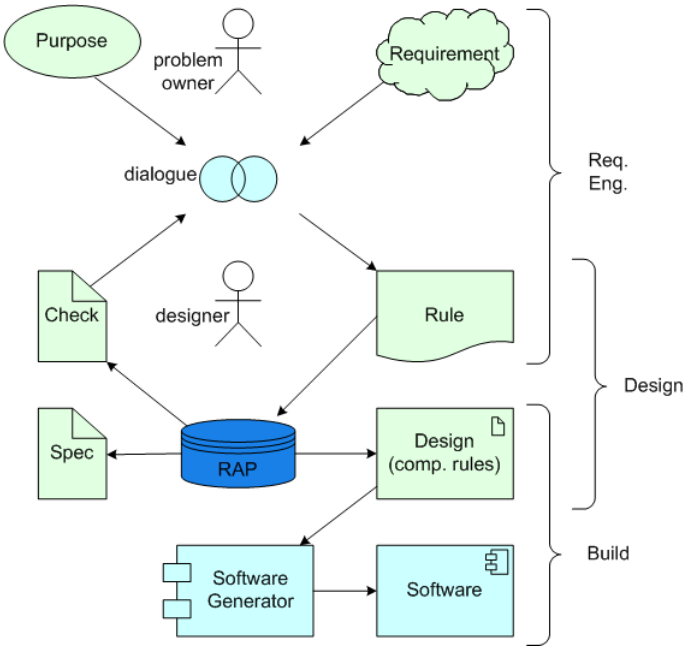

Figure 3: Design process for rule based process management

a problem owner and a designer. The former decides which requirements he wants and the latter makes sure they are captured accurately and completely. If the problem owner were an insurance broker, wishing to organize

automated sales at the back-end of his internet-shop, he might want to accept orders only from individuals whose identity is authenticated. That is a business rule.

The rule-based process designer may help the problem owner to make requirements explicit, but ownership of the requirements remains in the business. The designer can tell with the help of his tools whether the requirements are sufficiently complete and concrete to make a buildable specification. The designer sticks to the principle of one-requirement-one-rule, enabling him to explain the correspondence of the specification to the business.

A rule base (RAP) helps the designer by storing, managing and checking rules. Applications on top of the rule base help the designer to make specifications, analyze rule violations, and validate the design. An ADL-compiler analyzes rules and puts them in the rule base. The machinery helps designers by storing and checking rules, assisting in the assembly of a specification and a design. The specification is a document that visualizes and explains how software will support the business in the language of the business. It explains the business requirements to the business. So, each business rule is described as a business principle, to be maintained in the operations. Although each rule can be discussed (with the business) in isolation, in the technology they produce intricate and sometimes unforeseen interdependencies.

5 Evidence

Various research projects and projects in business have supplied a significant amount of evidence that supports the power of business rules convincingly. These projects have been conducted in various locations. Research at the Open University of the Netherlands has focused on two things: developing the CC-method and building a violation detector. Research at Ordina and TNO Informatie- en Communicatie Technologie has been conducted to establish the usability of ADL-rules for representing genuine business rules in genuine enterprises. Collaboration with the university of Eindhoven has produced a first prototype of a rule base. Experiments conducted at TNO, KPN, Bank MeesPierson, ING-Bank, Rabobank and Delta Lloyd have provided experimental corroboration of the method and insight in the limitations and practicalities involved. This section discusses some of these projects, pointing out which evidence has been acquired by each one of them. Our research has not yet resulted in a process engine as described in section 3. We expect that commercially available engines will come a long way in supporting rule based BPM, possibly with some adaptations.

The CC-method was conceived in 1996, resulting in a conceptual model called WorkPAD [16]. The method used relational rules to analyze complex problems in a conceptual way. WorkPAD was used as the foundation of process architecture as conducted in a company called Anaxagoras, which was founded in 1997. Conceptual analyses, which frequently drew on the WorkPAD heritage, applied in practical situations resulted in the PAM method for business process management [19], which is now frequently used by process architects at Ordina and within her customer organizations. Another early result of the CC-method is an interesting study of van Beek [27], who used CC to provide formal evidence for tool integration problems; work that led to a major policy shift in IT-projects. The early work on CC has provided evidence that an important architectural tool had been found: using business rules to solve architectural issues is large projects.

For lack of funding, the CC-method has long been restricted to be used in conceptual analysis and metamodeling [18, 9], although its potential for violation detection became clear as early as 1998. Metamodeling in CC was first used in a large scale, user-oriented investigation into the state of the art of BPM tools [10], which was performed for 12 governmental departments. In 2000, a violation detector was written for the ING-Sharing project, which proved the technology to be effective and even efficient on the scale of such a large software development project. After that, the approach started to take off.

In the meantime, TNO Informatie- en Communicatie Technologie (at that time still known as KPN Research) has used CC modeling for various other purposes. For example, CC modeling has been used to create consensus between groups of people with different ideas on various topics related to information security, leading to a security architecture for residential gateways [15]. That such CC models constitute buildable specifications has been shown by work done at the University of Twente [5].

Another purpose that TNO used CC modeling for was the study of international standardizations efforts such as RBAC (Role Based Access Control) in 2003 and architecture (IEEE 1471-2000) [12] in 2004. Several inconsistencies have been found in the last (draft) RBAC standard [2]. Also, it was noticed that for conformance, the draft standard does not require to enforce protection of objects, which one would expect to be the very purpose of any RBAC system.

The analysis of the IEEE 1471-2000 recommendation has uncovered ambiguities in some of the crucial notions defined therein. Fixing these ambiguities and making the rules governing architectural descriptions explicit has resulted in a small and elegant procedure for creating (parts of) such architectural descriptions in an efficient and effective way [14]. Additional

research at TNO [29] currently investigates the possibilities for creating context dependent 'cookbooks' for creating architectural descriptions in a given context, based on CC-modelling.

The efforts at TNO have provided the evidence that the CC-method works for its intended purpose, which is to accellerate discussions about complex subjects and produce concrete results from them in practical situations.

In 2002 research at the Open University was launched to further this work in the direction of an educative tool. This resulted in the ADL language, which was first used in practice by Bank MeesPierson [4]. The researcher described rules that govern the trade in securities at MeesPierson. He found that business rules can very well be used to do perpetual audit, solving many problems where control over the business and tolerance in daily operations are in conflict. At Rabobank, in a large project for designing a credit management service center, debates over terminology were settled on the basis of metamodels built in CC. These metamodels resulted in a noticable simplification of business processes and showed how system designs built in Rational Rose should be linked to process models [3]. The entire design of the process architecture [20] was validated in CC. At the same time, CC was used to define the notion of skill based distribution. This study led to the design by Ordina of a skill based insurance back-office for Interpolis. The CC-models developed for this purpose were reused in 2004 by Ordina to design an insurance back office for Delta Lloyd. This work provided an explanation why previous attempts to make design knowledge (represented in UML [23, 13]) had failed. It also demonstrated that a collection of business rules may be used as a design pattern [11] for the purpose of reusing design knowledge. In 2005 Ordina started a project to make knowledge reuse in the style demonstrated at Delta Lloyd into a repeatable effort.

6 Consequences

Business rules have a high potential for bringing a radical change in the way we design business processes. Since business rules can be communicated so much easier and can be used to control processes as well, rule based BPM carries the promise of bringing BPM closer to the business.

Currently, process modeling techniques force an ordering of activities upon the organization, which can sometimes be too confining. Where case management [28, 7] provides much more flexibility, it still requires process modeling and still requires a significant design effort. Rule based BPM does not have these limitations. It enforces nothing but business rules that are required by the business itself and the design can be automated further.

Generating process designs promises a further decrease in design times and a further increase in consistency and correctness of designs.

Where decreases in design times and better specifications are proven results, there are potential benefits to be obtained as well. Governance, as required e.g. by Sarbanes-Oxley [1] and other legislature, may be enhanced by the perpetual signaling of violations, an area explored by Barends [4]. The increased quality of specifications will make offshoring much easier. Offshoring might even become obsolete once specifications can be fed directly into software generators.

It has also become clear that the power of business rules, when represented in relational algebra, goes well beyond business process management alone. Rules have also been used in architectural studies, reusing knowledge in design patterns. Also, rules can be used to describe the modeling techniques and methods themselves; this is referred to as Metamodeling. The Open University of the Netherlands is currently teaching a metamodeling course, which employs the tools described in this paper.

Further research is required to bring this work from principle to production. A demonstration in which a currently available process engines employs the ADL rule engine in real enterprises is something that is required in the short term.

References

[1] Sarbanes-Oxley Act of 2002.

[2] ANSI/INCITS 359: Information Technology. *Role Based Access Control Document Number: ANSI/INCITS 359-2004.* InterNational Committee for Information Technology Standards (formerly NCITS), Feb. 2004.

[3] Baardman, E., and Joosten, S. Procesgericht systeemontwerp. *Informatie 47*, 1 (Jan. 2005), 50–55.

[4] Barends, R. Activeren van de administratieve organisatie. Research report, Bank MeesPierson and Open University of the Netherlands, November 26, 2003.

[5] Chong, C. N., Etalle, S., Hartel, P., Joosten, R., and Kleinhuis, G. Service brokerage with prolog. In *7th International Conference on Enterprise Information Systems (ICEIS), Miami, Florida, USA* (2005), I. Seruca and J. Cordeiro, Eds., INSTICC Press.

[6] De Morgan, A. On the syllogism: Iv, and on the logic of relations. *Transactions of the Cambridge Philosophical Society 10*, read in 1860, reprinted in 1966 (1883), 331–358.

[7] Deen, R. Het nut van case-handling - flexibel afhandelen. *Workflow magazine 6*, 4 (2000), 10–12.

[8] Dijkman, R. M., Ferreira Pires, L., and Joosten, S. M. Calculating with concepts: a technique for the development of business process support. In *Proceedings of the UML 2001 Workshop on Practical UML - Basesd Rigorous Development Methods Countering or Integrating the eXstremists* (2001), A. Evans, Ed.

[9] Dijkman, R. M., and Joosten, S. M. An algorithm to derive use case diagrams from business process models. In *Proceedings IASTED-SEA 2002* (2002).

[10] Dommelen, W. v., and Joosten, S. Vergelijkend onderzoek hulpmiddelen beheersing bedrijfsprocessen. Tech. rep., Anaxagoras and EDP Audit Pool, 1999.

[11] Fowler, M. *Analysis Patterns - Reusable Object Models*. Addison-Wesley, Menlo Park, 1997.

[12] IEEE: Architecture Working Group of the Software Engineering Committee. *Standard 1471-2000: Recommended Practice for Architectural Description of Software Intensive Systems*. IEEE Standards Department, 2000.

[13] Jakobson, I., Booch, G., and Rumbaugh, J. *The unified software development process*. Addison-Wesley, Reading, 1999.

[14] Joosten, R., and Beute, B. Requirements for personal network security architecture specifications. Tech. Rep. Freeband/PNP2008/D2.4, TNO, The Netherlands, 2005.

[15] Joosten, R., Knobbe, J.-W., Lenoir, P., Schaafsma, H., and Kleinhuis, G. Specifications for the RGE security architecture. Tech. Rep. Deliverable D5.2 Project TSIT 1021, TNO Telecom and Philips Research, The Netherlands, Aug. 2003.

[16] Joosten, S. Workpad - a conceptual framework for workflow process analysis and design. Unpublished, 1996.

[17] Joosten, S. Rekenen met taal. *Informatie 42* (juni 2000), 26–32.

[18] Joosten, S., and Purao, S. R. A rigorous approach for mapping workflows to object-oriented is models. *Journal of Database Management 13* (October-December 2002), 1–19.

[19] Joosten et.al., S. *Praktijkboek voor Procesarchitecten*, 1st ed. Kon. van Gorcum, Assen, September 2002.

[20] Ordina, and Rabobank. Procesarchitectuur van het servicecentrum financieren. Tech. rep., Ordina and Rabobank, Oct. 2003. presented at NK-architectuur 2004, www.cibit.nl.

[21] Peirce, C. S. Note b: the logic of relatives. In *Studies in Logic by Members of the Johns Hopkins University (Boston)* (1883), C. Peirce, Ed., Little, Brown & Co.

[22] Ross, R. G. *Principles of the Business Rules Approach*, 1 ed. Addison-Wesley, Reading, Massachussetts, Feb. 2003.

[23] Rumbaugh, J., Jakobson, I., and Booch, G. *The Unified Modeling Language Reference Manual*. Addison-Wesley, Reading, Mass., 1999.

[24] Schmidt, G., Hattensperger, C., and Winter, M. *Heterogeneous Relation Algebra*. in: Relational Methods in Computer Science, Advances in Computing Science. Springer-Verlag, 1997, ch. 3, pp. 39–53.

[25] Schröder, F. W. K. E. Algebra und logik der relative. In *Vorlesungen über die Algebra der Logik (exacte Logik)* (first published in Leipzig, 1895), Chelsea.

[26] Shewhart, W. A. *Statistical Method From the Viewpoint of Quality Control*. Dover Publications, New York, 1988 (originally published in 1939).

[27] van Beek, J. Generation workflow - how staffware workflow models can be generated from protos business models. Master's thesis, University of Twente, 2000.

[28] van der Tol, R. Workflow-systemen zijn niet flexibel genoeg. *AutomatiseringGids*, 11 (2000), 21.

[29] van Essen, E., Beute, B., and Joosten, R. Service domain design. Tech. Rep. Freeband/PNP2008/D3.2, TNO, The Netherlands, 2005.