Technische Universität Berlin

Forschungsberichte
der Fakultät IV – Elektrotechnik und Informatik

# Conformance Analysis of Organizational Models in a new Enterprise Modeling Framework using Algebraic Graph Transformation - Extended Version

Christoph Brandt
Frank Hermann

Bericht-Nr. 2011 – 10
ISSN 1436-9915

# Conformance Analysis of Organizational Models in a new Enterprise Modeling Framework using Algebraic Graph Transformation - Extended Version

Christoph Brandt[1,2], Frank Hermann[1]

[1] cbrandt@cs.tu-berlin.de, frank@cs.tu-berlin.de, Institut für Softwaretechnik und Theoretische Informatik, Technische Universität Berlin, Germany

[2] christoph.brandt@tudor.lu, FNR Pearl Project *ASINE*, Centre de Recherche Public Henri Tudor, Luxembourg

**Abstract**

Organizational models play a key role in today's enterprise modeling. These models often show up as partial models produced by people with different conceptual understandings in a usually decentralized organization, where they are modeled in a distributed and non-synchronized fashion. For this reason, there is a first major need to organize partial organizational models within a suitable modeling framework, and there is a second major need to check their mutual conformance. This builds the basis to integrate the partial organizational models later on into one holistic model of the organization. Moreover, the partial models can be used for model checking certain security, risk, and compliance constraints. In order to satisfy the two major needs, this paper presents two mutually aligned contributions. The first one is a new enterprise modeling framework—the EM-Cube. The second contribution is a new approach for checking conformance of models that are developed based on the suggested formal modeling technique associated with the proposed framework. In addition to that, we evaluate our potential solution against concrete requirements derived from a real-world scenario coming out of the finance industry.

**Keywords**: partial models, conformance analysis, enterprise modeling framework

## 1 Introduction

At Credit Suisse we discovered that today's enterprise modelling is facing several challenges concerning the sound integration of heterogeneous and incomplete models that are often developed and maintained in a distributed and decentralized way. For this reason, this paper focuses on how (organizational) models should be organized in an enterprise modeling framework given these hard side constraints. In particular, given a decentralized organization, the existing partial models need to be kept mutually aligned and consistent to make it possible to integrate them in a sound way later on. In order to do that, we review the real-world situation at Credit Suisse as well as the (scientific) literature with regards to enterprise modeling and conformance checks to come up with requirements based on a qualitative analysis for a potential solution.

From a practical point of view we focus on the *toolability* of a potential solution. Here, hard side constraints like the implementability need to be taken into account. In addition to that, we expect a potential solution to live in a product eco-system, which means that it should be able to co-exist (in an integrated way) with other products at the same time.

From a theoretical point of view we focus on the *soundness* of a potential solution. Here, hard side constraints are formal properties like correctness and completeness concerning the used modeling, integration and analysis techniques. Further, we expect a potential solution to be applied in the context of other methods or best-practices, which means that it should be in harmony with other methods and best-practices.

Therefore, the research question consists of two parts. Firstly, we need to understand what kind of modeling framework can support the organization of partial models which is at the same time compatible with the real-world setting of a big and decentralized organization like an international bank. Secondly, we need to find a way to check consistency issues between partial models using implementable formal methods that work for any set of valid input models, such that the implemented tools are usable by the people in the field.

As a consequence, this paper presents two contributions that are mutually aligned. The first one is a new enterprise modeling framework, the EM-Cube, that supports the organization of local knowledge of an organization and that is at the same time compatible with requirements from a big and decentralized organization. The second one is an implementable approach that helps to analyze the conformance of partial models of an organizational model using formal techniques of algebraic graph transformation that come with formal guarantees like correctness, completeness, termination and efficiency as well as usability.

In detail, the paper is organized as follows: We first present as problem statement our findings from the real-world scenario at Credit Suisse as well as from the (scientific) literature, which is evaluated with regards to enterprise modeling and conformance analysis. Based on these findings, we present requirements derived from the real-world scenario and the related work that we use to ground a new enterprise modeling framework and its corresponding modeling technique. In a next step, this framework, the EM-Cube, is introduced and evaluated. Afterwards, an approach for conformance analysis of models based on the suggested modeling technique is developed and evaluated. Finally, we demonstrate the applicability of our approach by the help of a concrete example that is likewise evaluated and finish with some conclusions as well as potential future work.

## 2 Problem Statement

### 2.1 The Credit Suisse Scenario

The Credit Suisse scenario serves to motivate our potential solution and it provides the empirical source for real-world requirements that are used for evaluation later on. In detail, we will reflect how organizational models that are part of the set of the overall enterprise models at Credit Suisse are produced, organized and managed [71].

In the given case, a bank like Credit Suisse has a business and IT branch that are expected to operate in an aligned but independent way. In this context, the purpose of the used IT technology is to support financial business processes. In order to achieve this goal, the IT branch focuses on setting up an appropriate IT architecture [95]. However, there are different views on the IT at Credit Suisse. One view is about the way a software infrastructure is generated. This is done by the help of a model driven engineering [150] (MDE) process. Another view is about the resulting IT architecture, its management and development. In order to develop and manage the IT architecture, the IT branch of Credit Suisse selected TOGAF [132] as the most suitable best practice from their point of view. Both views are related towards each other. An illustration is presented in Fig. 1.

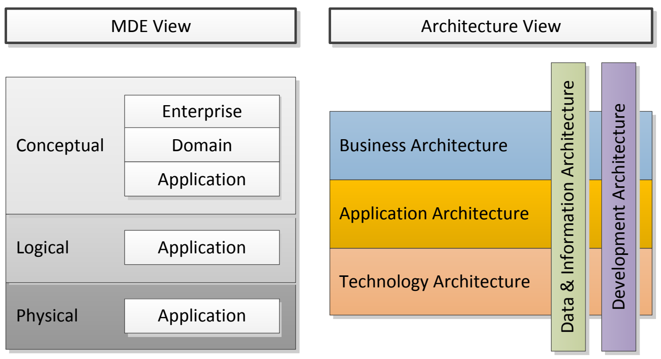


Figure 1: Credit Suisse Views

The view of the MDE process is driven by different conceptual models [63], which are formulated using UML [125] or specific UML profiles [70]. Based on the conceptual models the MDE process generates logical models, which are then translated into physical models representing concrete application programs that can be instantiated and executed.

The view of the IT architecture is driven by focussing on a business architecture, an application architecture and a technology architecture. Orthogonal to these architectures at Credit Suisse, there is a data and information architecture as well as a development architecture. The business architecture grasps business requirements as data types and class models that are used later on in the MDE process. The application architecture defines the application landscape as well as the interactions between different applications from a logical (and physical) point of view. The technology architecture targets the concrete hard- and software base. The data and information architecture is about scheme development and integration related to the big volumes of data the bank has to handle. The development architecture is about the architecture of a tool chain used to generate applications based on available models by the help of code generators.

The MDE process is put into relationship with the TOGAF process. In detail, the application architecture is mapped into a conceptual, logical and physical layer, whereas the business architecture can be found in the conceptual layer only. The technology architecture finally has its reference in the physical layer of the MDE process. In detail, the MDE development process at Credit Suisse operates at the conceptual layer with UML-based BOMs (business object models) [14]. It therefore abstracts away the management of terminologies and sticks with a pure data type approach. At the logical and physical MDE layer, logical and physical UML-based application models are used.

Therefore, Credit Suisse is putting priority on modeling an all encompassing IT architecture by the help of a customized TOGAF process for modeling, building and maintaining an IT architecture. From a methodological point of view, a classical software engineering approach is taken that fully builds on UML and specific UML profiles as well as automation support by the help of code generators.

Organizational security, risk and compliance issues are administrated by applying informal guidelines, checklists and proprietary standards with regard to the used technology as well as the running business processes, which are enforced on a case-by-case basis by specialized IT task forces in a handwork way.

## 2.2 Related Work

### 2.2.1 Enterprise Modeling

The presented state of the art of enterprise modeling motivates likewise our potential solution. It has extensively been reflected in [147] using the industrial standard IEEE 1471 [87], which provides a conceptual framework to describe and classify software intense systems. It is by itself no method, but a recommended best-practice and acts as a frame of reference, which focusses on the system of interest, its architecture and stakeholders, their concerns, the architectural descriptions, the viewpoints of the stakeholders and the views of the architectural description, the modeling language and the models. However, we do not develop the full spectrum here, but focus on those aspects, which will help to motivate real-world requirements for the evaluation of a new enterprise modeling framework encompassing its corresponding modeling technique.

In the following, we present an overview of some characteristic qualities of enterprise modeling approaches from the point of view of the Zachmann Framework, ARIS, GERAM, SOM, MEMO, TU Lisbon, SEAM, KTH Stockholm, TOGAF and Archimate, the EA Cube, Niemann and St. Gallen.

The Zachmann Framework [168, 148] is a framework for an information system architecture covering the whole enterprise. It is best characterized by its modeling layers *scope*, *business*, *logical systems*, *technical systems* and *detailed representations*. Each layer is sliced by the central questions of *what*, *how*, *where*, *who*, and *why* and comes with an increasing level of details.

ARIS (*Architektur integrierter Informationssystemmodellierung*) is a specialized architecture [140, 141, 142], which standardizes the way a business information system is modeled. It sticks with a business process perspective and comes with different, but integrated views, the ARIS house, on the enterprise using a relational database model, as there is an organizational view, a data view, a control view, a functional view, and an output view.

GERAM (Generalized Enterprise Reference Architecture and Methodology) [26, 88, 25, 123, 89, 88] is a reference framework for existing methodologies and modeling techniques that supports the comparison and evaluation of existing enterprise modeling approaches. It became an ISO standard in 2000 [89]. It can be best characterized by the nine modules, in which it decomposes. They are about the used reference architecture, the engineering methodology, the modelling languages, partial models, enterprise modeling concepts, tools, enterprise models and their implementations as well as an enterprise operational system.

The SOM approach (Semantic Object Model Approach) [68, 67, 66] is a special enterprise modeling framework with characteristic layers about the plan of an enterprise, the business objects and processes and its organizational, technical and physical structure. Central to the SOM approach is its emphasis of an business object and process system.

The MEMO (Multi-perspective Enterprise Modeling) approach [72, 74, 97, 101, 73] can best be characterized as modeling from different perspectives like a strategic and organizational perspective as well as a perspective of information systems. These perspectives are treated in an integrated way.

The TOGAF framework (The Open Group Architecture Framework) [153] started as a special technical architecture framework for information systems (TAFIM), which was published by the US Department of Defense. The current version builds on the ISO Standard 42010 [87]. It can best be characterized as a codified best-practice from industry supporting the management of enterprise architectures. As such, it distinguishes a business architecture, a data architecture, an application architecture and a technology architecture. It assumes a meta model, which organizes the content types for all architectural layers. In detail, the framework comes with six meta-model extensions that are about governance, services [151], business processes, data, infrastructure and motivation modeling.

An optional extension to the TOGAF framework is the Archimate language [94, 96, 109, 108, 93, 16, 110, 130, 111], which aims to support the description of different aspects like the structure and behavior of an enterprise as well as the processed information. It further assumes plurality with respect to different visualizations and viewpoints, and seeks to support the integration of existing documents of enterprise models. Further, it differentiates three levels, the business, the application and the technology level. For all aspects and layers it provides appropriate concepts and visualizations. In addition to that, it is assumed to be flexible with respect to different viewpoints and meta-models.

The management approach of enterprise architectures (EA) from TU Lisbon [156, 157, 158, 43, 158, 19, 159, 40, 21, 20, 18, 41, 115, 167] is about modeling information system architectures (ISA) and about the modeling method itself. The approach uses a high-level meta-model, the CEO framework, and refines this into five views as there is an organizational view, a business view, an information view, a system's application view and a system's technological view. It can best be characterized by the sub architectures it assumes like the informational architecture, the application architecture and the technological architecture. It further builds upon a conceptual understanding of certain business goals that are assigned to business processes, which run on IT systems.

The SEAM approach (Systemic Enterprise Architecture Methodology) [160, 112, 22, 113, 138, 139, 162, 163, 161, 133] supports interdisciplinary EA projects with methods and models. By doing this, SEAM seeks to complement existing EA approaches by methodological guidance. Key aspects of the SEAM approach are the configuration of modeling languages as well as the hierarchical understanding of enterprise systems.

The management approach of enterprise architectures by the KTH Stockholm [61, 91, 76, 92, 90, 106, 107, 119, 39, 60, 98, 131] provides decision support for the IT management in enterprises. It, therefore, focusses on techniques and models that help to analyze specific qualities of enterprise architectures. As consequence, it complements techniques that primarily deal with modeling enterprise architectures. The used information models correspond to different view points like an architectural viewpoint, a business process viewpoint and an application usage viewpoint.

The EA Cube [23, 48, 24, 49, 46, 47] codifies insights gained from industry and academia. It can best be described by its hierarchical levels as there is a level of goals and initiatives, a level of processes and services [151], a level of data and information, a level of systems and applications, and a level of networks and infrastructure. Orthogonal to these levels there are common threads in the area of security, standards and workforce. All this is organized into different segments, one for each line of business. It realizes at the same time a language and a method. The language defines what is being modeled, whereas the method defines how this happens.

The EA Management approach of Niemann [122] summarizes key lessons and guidelines derived from industrial experience. It can best be described by its three main modeling levels, which are the business architecture, the application architecture and the system architecture. Complementary to this structural aspect is the methodological aspect, which affects the different work phases like documentation, analysis, planning and acting.

The EA Management approach of the University of St. Gallen [127, 164, 6, 38, 105, 143, 128, 9, 10, 69, 79, 165, 8, 166, 11, 103, 104, 134, 7, 80] focusses on an holistic approach to design organizations, which is, therefore, closely related to enterprise modeling and the management of enterprise architectures. It can best be characterized by its core business meta-model that serves to model a commercial organization as part of a business engineering activity. The core business meta-model is complemented by additional extensions that serve to specify the strategy and possible information systems. The modeling framework consists of the following layers: a business architecture layer, a process architecture layer, an integration architecture layer, a software architecture layer and a technology architecture layer. At the meta-level the modeling process comes as a special approach to meta-model engineering, which serves to customize and extent the core business meta-model that is used to specify the design of a commercial organization.

Based on this analysis we focus on aspects like modeling frameworks and modeling techniques, partial models, modeling domains and abstraction layers as well as model requirements. Therefore, we do not focus on very specific solutions assuming a certain organizational type, IT architecture, modeling methodology or modeling language.

### 2.2.2 Conformance Analysis

The presented state of the art of conformance analysis is intended to motivate likewise our potential solution in a similar fashion as in the previous cases. As before, we focus in our presentation on specific qualities that help to motivate the real-world requirements for the evaluation of a new enterprise modeling framework encompassing its corresponding modeling technique.

In [116] the author analyze whether a business process is executed according to the prescribed behavior in a process model in order to check for its conformance. In detail, the author looked at feedback aspects such as fitness, precision, generalization, structure, frequency, violation and location, which are discussed in the literature. Fitness defines how much of the behavior in an event log is captured by a process model. Precision indicates how much extra behavior has been contained in the model but no such behavior can be found in the provided event log. Generalization indicates if a model is not too precise (hence too static). Structure indicates if the a priori model is well structured and if it is readable. Frequency indicates how often a specific behavior has been performed in reality. Violation is the complement of the fitness aspect. It assumes that the provided process model only contains behavior that is allowed. Location indicates where the observed behavior has deviated from the a priori model. In his study he checked, which of those aspects are most relevant in a commercial environment. According to the findings of the author all techniques are of equal importance. However, the understandability of the techniques were judged differently by potential users. Fitness and violation aspects were easier to understand than other techniques. In addition to that, managers understood feedback techniques significantly better than supervisors.

Similar issues have been investigated in [135, 136, 137, 137, 3]. Here, the authors discuss conformance analysis between real (business) processes and underlying event logs as well as real business processes and idealized reference models of such processes. In the first case, they focus on fitness and appropriateness. In the second case, they investigate and quantify the deviations between business processes in order to check for regulatory compliance.

A fully different point of view is taken by the authors in [121, 120]. Here, conformance is discussed by looking at software engineering documents and their relationships. Such documents can, for example, represent raw data for (business) process specification. In detail, the authors exam under which conditions such documents are in semantic harmony by using a hyper text approach.

Based on this analysis we focus on aspects like conformance between models of different organizational sub-domains within a similar level of abstraction. Therefore, we do not focus on aspects related to the conformance of models within different levels of abstraction within one single domain.

## 2.3 Analysis

### 2.3.1 Requirements derived from the real-world Scenario

Subsequently, we like to present selected requirements that are intended to support and evaluate the development of a new enterprise modeling framework as well as a corresponding modeling technique based on insights gained from the empirical analysis of the real-world scenario at Credit Suisse. We conceptually introduce each requirement and show its underlying motivation with relation to the analyzed scenario. The selected requirements are summarized in the following table (see Fig. 2).

| Name | Description | Check |
|---|---|---|
| R1a | Separation of Organizational Models and IT Architectural Models | Must |
| R1b | Separation of IT Architectural Models from IT Technology Models | Must |
| R2 | Separation of Organizational Business and IT Models | Must |
| R3 | Mutual Alignment of Organizational Business and Organizational IT Models | Must |
| R4a | Mutual Alignment of Organizational IT and IT Architectural Models | Must |
| R4b | Mutual Alignment of IT Architectural Models and IT Technology Models | Must |
| R5 | Separation of Organizational Models from Organizational Constraints | Must |
| R6 | Endowing Organizational Models with Formal Semantics | Must |
| R7 | Endowing Organizational Constraints with Formal Semantics | Must |
| R8 | Realizing Formal Semantics with Formal Tools | Must |
| R9 | IT Architecture Model reflects a Service Oriented Architecture | Optional |
| R10 | Use of one Generic Formal Syntax Model for all Types of Languages | Must |
| R11 | Use of Implementable Formal Syntax Transformation Techniques | Must |
| R12 | Support of Conformance Checks between Models | Must |
| R13 | Support of Language Families of Small Domain Languages | Must |
| R14 | Support of bidirectional Model Transformation and Integration | Must |
| R15 | Support of Partial Models | Must |
| R16 | Support of a loose Coupling of Models | Must |

Figure 2: Selected Requirements based on the Credit Suisse Scenario

In the scenario, we found evidence that it is an advantage to separate organizational models from IT architectural models (R1a) because this helps to avoid modeling in two conceptual coordinate systems in an integrated way, which unavoidably is causing problems. The same is true when separating IT architectural models from IT technology models (R1b). In both cases, the underlying domain specific modeling languages evolve independently, the same may become true for their conceptual coordinate systems, which is going to break integrated modeling artifacts and, therefore, results in maintainability problems like lower flexibility and higher costs. A different situation can be found in the scenario between organizational business and organizational IT models (R2). Here, a different speed of modeling life cycles is the primary reason to keep the mutually depending modeling spaces separate. Assuming separate modeling spaces for organizational business and organizational IT models, a bidirectional alignment between these models in their prevailing modeling spaces become necessary (R3). Analogously, organizational IT models representing the data logistics in organizational terms need to be aligned with IT architectural models (R4a), and IT architectural models need to be aligned with IT technology models (R4b) representing the underlying hard- and software infrastructure in software and system engineering terms. The need to check organizational models regarding certain security, risk and compliance properties calls for organizational constraints, which leads to a natural separation of the modeling space of organizational models and the one for their corresponding constraints (R5). The objective to have the organizational constraints checked in a sound way leads to the requirement of using formal methods, which makes it necessary to have organizational models and their constraints underpinned by some kind of formal semantics (R6, R7). In addition to that, any such kind of formal semantics needs to be grounded in existing implementations of available formal tools in order to be able to automate possible model checking techniques (R8) easily. Depending on the current fashion, the IT architectural models may put a description layer on top of the existent hard- and software infrastructure, which encapsulates technological models. Here, architectural models may represent a service oriented architecture (R9). However, our understanding is that regarding the examined scenario this is a soft requirement that may be altered towards any other suitable type of IT architecture as long as the type of architecture does not remain unspecified. Because we need to discuss properties of languages and their artifacts a generic formal syntax technique is needed (R10), which can cope with all types of languages used in our scenario at the same time. Once, such language artifacts that represent models in our scenario are required to be transformed or synchronized in a automated and sound way, implementable formal syntax transformation techniques are needed (R11). Assuming decentralized and asynchronous modeling activities, automated conformance tests become mandatory (R12) to identify potential inconsistencies between models. We believe that such tests should be realized on top of the generic formal language model and its corresponding syntax transformation techniques. Further, our observations of the real-world modeling activities and processes at Credit Suisse led to the insight that we need support for integrated language families of small and evolving domain specific modeling languages (R13). In the ideal case, this support is realized by the help of the generic formal language model and its corresponding syntax transformation techniques we propose, which would facilitate the language integration and prepare the ground for the formal analysis of potential language and model properties. Regarding possible model transformation and integration operations we found evidence that such operations are required to work in a bidirectional way (R14) in order to support the management of horizontal relationships of models in addition to classical vertical relations as they can be found in today's generator driven approaches. We believe that organizational knowledge can best be represented by the help of partial models (R15). Finally, the change rate, we were able to observe, requires models to be loosely coupled (R16). The best-practice of tight data schema integration applied to the different domain models used at Credit Suisse fails here in the evaluated scenario to provide the needed flexibility.

### 2.3.2 Requirements derived from related work

As in the above case, we like to present selected requirements that are intended to support and evaluate the development of a new enterprise modeling framework as well as the corresponding modeling technique we suggest based on insights gained from the analysis of the related work, taking the view of enterprise modeling and conformance analysis as well as others. We conceptually introduce each requirement and show its underlying motivation with relation to the literature. The selected requirements are summarized in the following table (see Fig. 3).

| Name | Description | Check |
|---|---|---|
| R17 | Use of a Model Framework | Must |
| R18 | Use of Enterprise Engineering as Methodology | Must |
| R19 | Letting the concrete IT Architecture being a Parameter | Must |
| R20 | Support for Language-Critical Reconstruction Techniques | Must |
| R21 | Support of Agile Modeling Techniques | Must |
| R22 | Support of Model Transformation Techniques | Must |
| R23 | Support of Language Spaces | Must |
| R24 | Support of Scheme- and Instance-Models | Must |
| R25 | Mutual Alignment of Models and their Requirements | Must |
| R26 | Use of Formal Syntax Analysis Techniques | Must |
| R27 | Reuse of already Implemented Formal Semantics | Must |
| R28 | Support of Modeling and Programming Languages | Must |
| R29 | Support of Graphical and Textual Domain Languages | Must |
| R30 | Support of Formal Languages | Must |
| R31 | Support of Language Integration | Must |
| R32 | Use of Dimensional Descriptors | Must |

Figure 3: Selected Requirements based on the Related Work

As it is discussed in various references [89] there is an obvious need to organize enterprise models by the help of a modeling framework (R17). Such enterprise models are usually created using one possible modeling technique [154, 53], whereas the recommended modeling methodology [126] remains orthogonal to the chosen technique. In [126] enterprise engineering is recommended (R18) as methodology for the development of organizational models. Further, the wide variety of possible IT architectures discussed in the literature [10] makes it look reasonable to keep the type of the concrete IT architecture, which is part of an overall enterprise architecture, a given, but variable parameter (R19). Besides this and because the overwhelming number of enterprise documents are written in some natural language a kind of natural language processing is required before their content can be used to develop enterprise models. Here, language-critical re-construction techniques as introduced in [144] show a good potential to support this process (R20). Regarding possible modeling techniques, agile modeling [13] is maybe the best option when it comes to incomplete and inconsistent information in a distributive and asynchronous modeling process that need to be supported in order to set up organizational enterprise models (R21). Today, such enterprise models are heavily used to generate source code as it is discussed in [150]. So, it appears mandatory to support vertical transformation techniques (R22). For practical reasons, organizational enterprise models may be part of different, but loosely integrated sub-domains as there are processes and rules, for example [31]. Therefore, the support of handling different, but overlapping language spaces is needed (R23). In addition to that it is important to notice that enterprise models not only encompass schemes but also instances [30]. A scheme can be, for example, a business process model, whereas an instance, on the other hand, may represent a given IT landscape. As a consequence, scheme and instance models must be supported (R24) at the same time. Further, the need for a continues quality assurance of enterprise models [36] makes it necessary to ground them in corresponding model requirements (R25). This could be supported by linking both, models and requirements, in a bidirectional way. Besides this, quality assurance of models also benefits from formal syntax analysis techniques (R26), which help to assure, for example, that certain syntactical modeling constraints are respected [53]. The need to check semantic properties of enterprise models makes it looks promising to ground their specific semantics on the implemented semantics of already existing formal tools [28, 37, 36] (R27), because this enables software reuse and automatic evaluation of semantic properties at the same time. In the related work, modeling languages like UML [154] as well as programming languages like Java [12, 17] are used to build enterprise models and to generate executable code at the end (R28). Sometimes modeling languages are complemented by textual or graphical domain languages [30, 31] (R29). However, the underlying purpose remains the same, generating program code. In specific scenarios, formal specification languages are used (R30), for example, to check certain semantic properties of domain models [36]. Because enterprise models often are build up on top of different types of languages [32], language integration becomes a central issue [155] (R31). Finally, in order to keep a future solution of enterprise modeling manageable, orthogonal descriptors should help to keep the overall complexity low [88, 89] (R32).

# 3 Enterprise Modeling

## 3.1 The new EM-Cube

The first main contribution is the new EM-Cube, which is presented next, an evaluation is given afterwards. It is a special enterprise modeling framework that supports the organization of partial organizational models in a way, which is compatible with a *decentralized* and *asynchronous* modeling process executed people having different conceptual understandings observed in the Credit Suisse scenario. It is further aligned with formal techniques of algebraic graph transformation that we suggest as corresponding modeling technique. Algebraic graph transformation helps to automate a wide range of modeling operations needed in the context of this cube.

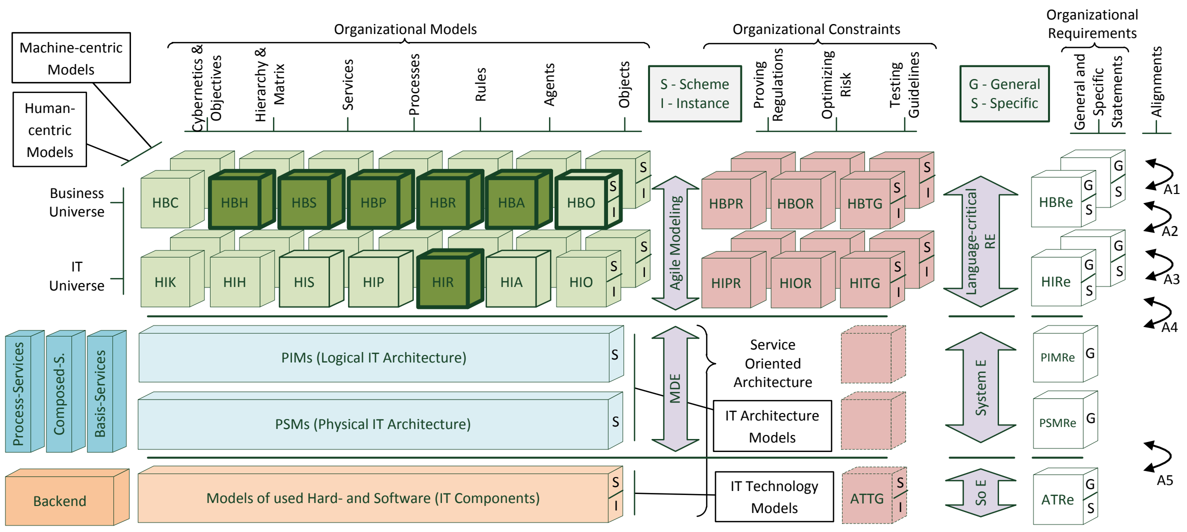


Figure 4: The EM-Cube (Version 2)

In detail, the purpose of the EM-Cube[1] is to provide a modeling framework for the organizational business model, the organizational IT model, the (logical and physical) IT architecture model and the IT technology model of a commercial organization. Because a holistic organizational model is usually not available as such, it is assumed to be build up based on partial organizational models. These partial organizational models fall into different organizational sub-domains as shown in Fig. 4. They target cybernetic models, models of the organizational structure, organizational service models, organizational process models, organizational rule models, organizational agent models and organizational object models. The corresponding families of domain specific modeling languages for the different organizational sub-domains are assumed to be mutually integrated in a loose way.

Further on, the EM-Cube requires to model organizational constrains separately from organizational models in order to enable model checking of organizational models later on. The suggested domains for organizational constraints fall into different champs depending on the underlying verification or validation techniques used for the checks as there are proving or simulation techniques. In the special case of testing constraints that relate to real-world organizational behavior of a socio-technical system, such constraints can not be model checked and therefore need to be propagated into the IT hardware- and software infrastructure and validated during normal operations.

[1]A first version of the EM-Cube was published in [35, 36].

Requirements, finally, constitute a modeling space of their own. Organizational requirements are assumed to provide the reconstructed knowledge base used to ground (partial) organizational models. They may be derived out of organizational handbooks, or appear out of interviews with organizational domain experts. IT architectural requirements are assumed to provide the architectural specifications for the IT architecture model. Technological requirements specify soft- and hardware properties of the back-end. Organizational, IT architectural and technological requirements are assumed to be loosely coupled with their corresponding models and constraints.

The separation of the different modeling domains of the organizational business and IT model, of the organizational IT model and the IT architecture model, and of the IT architecture model and the IT infrastructure model is cured by mutual alignments (A2, A4, A5), respectively. As a consequence, the EM-Cube helps to organize an organizational business model that is aligned with an organizational IT model, which is mapped towards a logical IT architecture that leads to a physical IT architecture, which is aligned with an existing hard- and software infrastructure represented by the help of an IT infrastructure model.

The organizational models in the EM-Cube are either human-centric or machine-centric models. Human-centric models are models build by the help of domain specific modeling languages, which support incomplete, inconsistent and partial models. Machine-centric models are models created using machine languages, which are expected to be well-formed. Human-centic models are intended to grasp the modeling input from humans, whereas machine-centric models are intended to provide (a part of) the codified formal semantics for such human-centric models. In order to equip human-centric models with a formal semantics in a flexible way they are aligned by the help of declarative correspondance rules with machine-centric models (A1, A3). Either type of model shows up as syntactical artifact, which can be processed automatically by implemented formal syntax techniques.

Further on, models can be scheme or instance models. In the case of organizational models, scheme models describe, for example, the type of a business process like a loan granting process. An organizational instance model, in contrast, could be used to represent a given IT landscape. In the case of technological models, scheme models may end up as programs coming out of a generation process, whereas instance models may be used to reference running software services (A5). Analogously, requirements decompose into general and specific statements, either grounding parts of a scheme or instance model.

In order to be as close as possible to the given real-world scenario at Credit Suisse the EM-Cube assumes an IT architecture that is service oriented. However, other IT architectures are possible, too. Here, this IT architecture model is described by platform independent and platform dependent models, which can be decomposed into basis services, composed services and process services models. Platform dependent models are thought to be aligned with a given hard- and software infrastructure, the so called back-end.

Given the different types of models, we like to suggest to apply different modeling processes. Human-centric models may be best created using an agile modeling process [13], whereas IT architecture models fit better with a process that uses a model driven engineering approach (MDE) [150]. Organizational requirements, which need to be reconstructed from organizational handbooks could be produced using a modeling process based on language-critical re-construction techniques ("Language-critical RE") [144], whereas architectural requirements could be supported by the help of system engineering techniques ("System E") [77], and technological requirements could be realized using software engineering techniques ("So E") [27].

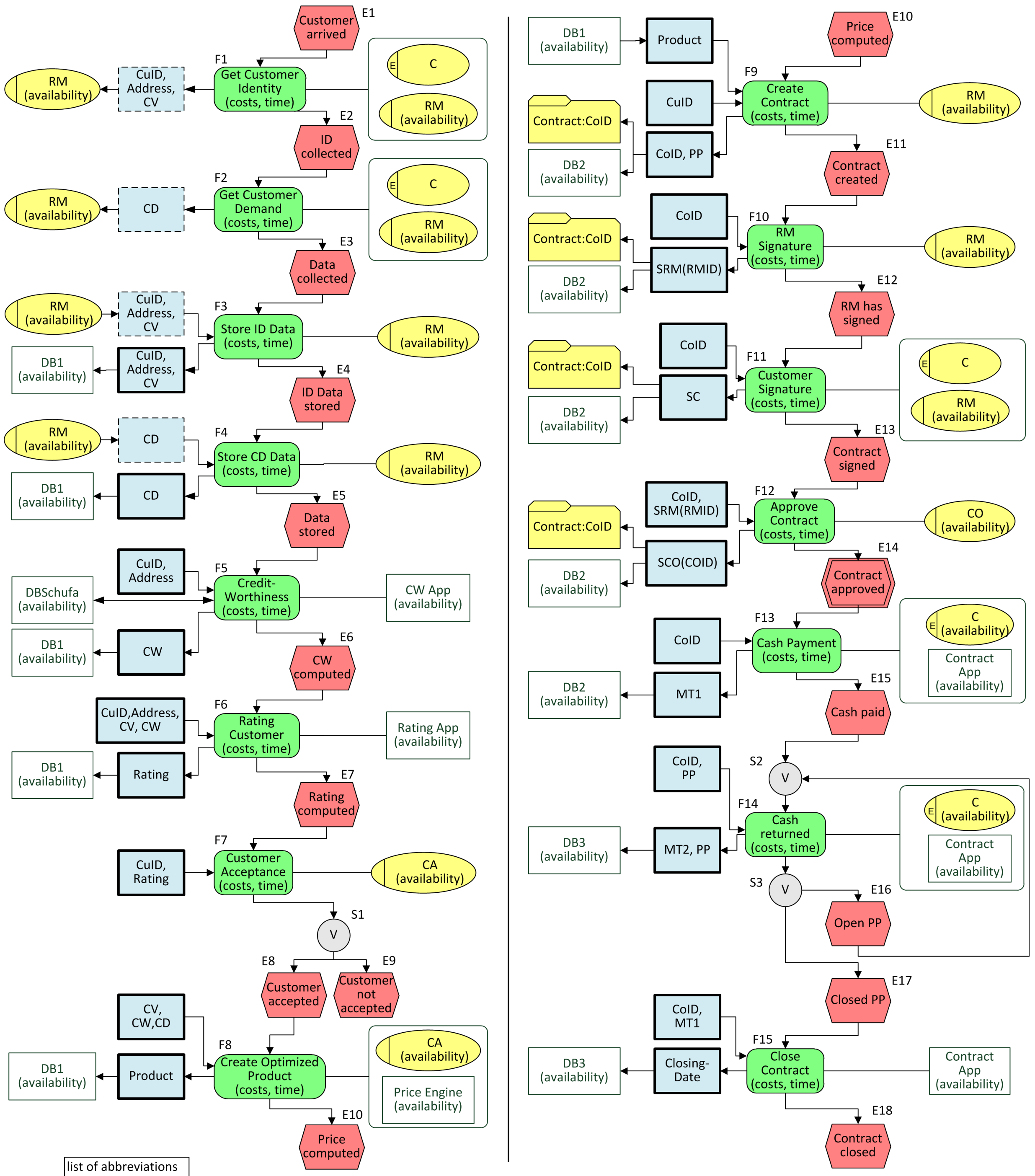
## 3.2 Example: Concrete Organizational Enterprise Models

For the purpose of demonstration, we use an already presented human-centric business process model [36] that we like to check if it conforms with other human-centric organizational models. Here, the used business process model is build upon a simplified loan granting process in the finance industry that is complemented by continuity snippets, which encode possible back-up solutions in case parts of the original process fail [36].

In the following, however, we abstract from business continuity and put the focus on conformance problems between such a concrete loan granting process as presented in Fig. 5 and other organizational models that may be related to it. In our case, those other organizational models are models of the organizational hierarchy, models of the business and IT service landscape, organizational models of access rights and type structures for organizational agents as presented in Fig. 6. Based on what we have learned from the Credit Suisse scenario such organizational models are typically provided by different people at different places and at different times in the organization. Beyond that we learned that these people are organizational domain experts who are codifying their local knowledge into such models using small but focused ad-hoc domain languages using their own concepts while describing the organization.

In detail, we now like to introduce the different organizational models that constitute our example. In the Credit Suisse scenario domain models come as special UML models. However, here, we use for the sake of simplicity an extended event-driven process chain (EPC) model for the critical business process in Fig. 5 as well as some domain models expressed in ad-hoc DSMLs for the related organizational models in Fig. 6.

The human-centric business process (BP) model in Fig. 5 represents a simplified loan granting process. The overall picture is that in this process a relationship manager (RM) gets (F1, F2) data from a potential customer (C) that is stored (F3, F4) into some databases and used to calculate (F5, F6) the credit worthiness (CW) and the overall rating of a customer to enable (F7) a customer acceptance decision for the requested loan. Once, such a customer is accepted, a personalized loan offer and a corresponding contract is created (F8, F9). This contract is signed (F10, F11) in the following by the relationship manager and the customer. A credit officer (CO) needs to approve (F12) this contract afterwards. Finally, the loan is payed out (F13) and the repayment of the loan starts (F14). When the last installment is payed, the contract is closed (F15). Besides this high-level view, we like to point to some technical details of the model. In comparison with a standard EPC, this model is extended in a way that it has organizational objects in two columns. On the right side they define which organizational entities are executing a certain business or IT function (F). On the left side they depict the sources and sinks



list of abbreviations

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| App | Application | CO | Credit Officer | MT | Money Transfer | SRM | Signature of Relationship Manager |
| C | Customer | CuID | Customer ID | PP | Payment Plan | RMID | Relationship Manager ID |
| CA | Credit Advisor | CV | Customer Value | RM | Relationship Manager | | |
| CD | Customer Demand | CW | Credit-Worthiness | SC | Signature of Customer | | |
| CoID | Contract ID | DB | Data Base | SCO | Signature of Credit Officer | | |

Figure 5: Given human-centric business process model (BP model $M_1$)

of the ongoing data flow between organizational entities and business or IT functions. Other candidates for such sources and sinks are IT applications and real-world documents like contracts. The blue rectangles that fit between the business and IT functions and the sources and sinks of the data flow represent the view on the cache of a potential workflow engine while executing this workflow. Rectangles with bold edges stand for permanent memory cells while rectangles with dashed edges represent volatile memory cells. Therefore, a workflow engine can temporarily cache data coming from business or IT functions as well as applications or organizational entities. The volatile elements help to have manual and automated parts of a workflow in one integrated model. In the middle of the diagram, business and IT functions are listed as green, and possible events are listed as red diagram elements. Composed elements, as it is the case for some organizational entities and applications, are thought to act as one element.

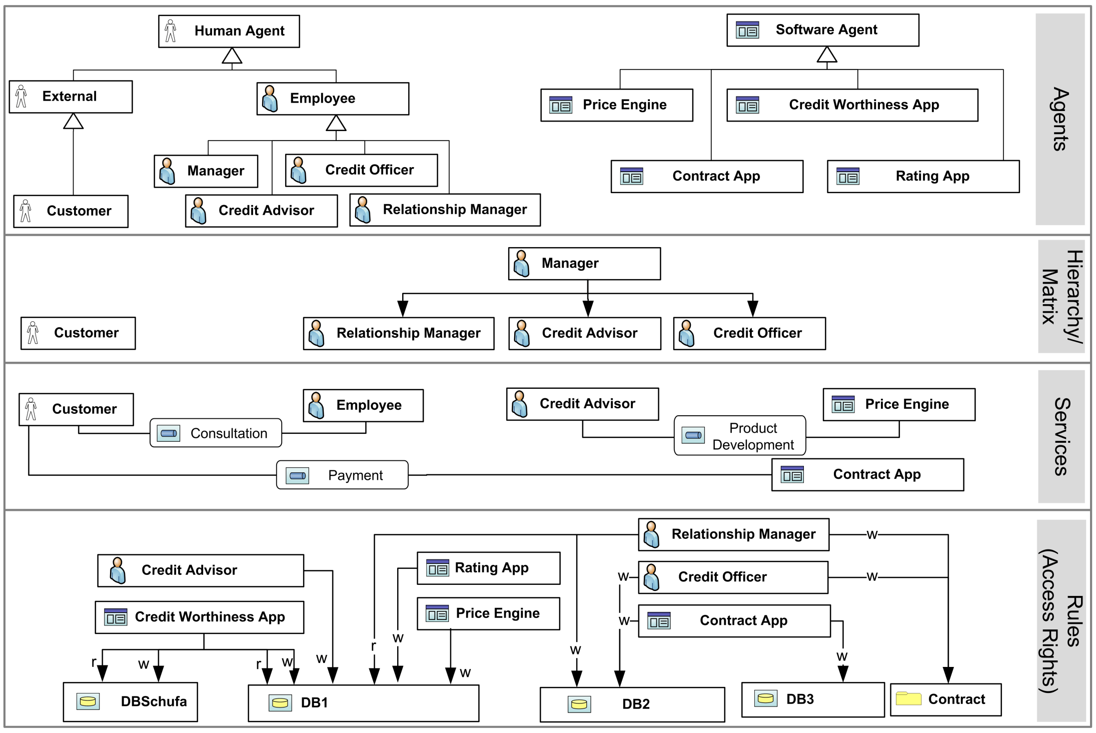


Figure 6: Given and (partially) integrated human-centric organizational models (PIO model $M_2$)

The human-centric organizational model in Fig. 6 is a composition of actually 4 organizational models that are presented in an partially integrated way in order to simplify our example on conformance analysis. However, the conformance analysis can be performed as well separately for the single models, i.e., via 4 separate conformance checks.

The first component of the partially integrated organizational (PIO) model in Fig. 6 specifies a human-centric agent model (see top fragment of the figure), which comes as a special type hierarchy. It differentiates between human and software agents. Human agents are further specialized into external and internal agents. Here, external agents are customers and internal agents are managers, credit advisors, credit officers and relationship managers. Software agents subsume price engines, contract, credit worthiness and rating applications.

The human-centirc hierarchy model in Fig. 6 defines that a manager is supervising a relationship manager, a credit advisor and a credit officer. Customers stand for their own.

The human-centric service landscape model in Fig. 6 documents that there are different business service nodes like a credit advisor, an employee and a customer as well as IT service nodes like a price engine and a contract application. The different communication channels between those service nodes are named according to their underlying purpose.

The human-centric rule model in Fig. 6 is representing access rights that define which agent has a read or write access right in relation to a certain database application or regarding a contractual paper document.

## 3.3 Illustrative Remarks about Details of the EM-Cube

Both, organizational models and constraints as well as the corresponding organizational requirements, are explained in detail in the following tables.

| Business / IT Universe | Symbol | Domain: Illustrative remarks |
|---|---|---|
| Human-centric Models | $M_{HBC/HIC}$ | *Cybernetics and Objectives*: Cybernetic IT models are discussed in [4]. One possible approach is to use multi-domain models build on Modelica [75]. Objectives can be realized as target functions in a control circuit. Cybernetic models could add control elements to a business process model in order to manage its performance helping to keep the original process model lean. In the context of the EM-Cube we like to recommend to use a graphical domain language for control models that may be enriched with a textual domain language. |
| | $M_{HBH/HIH}$ | *Hierachy and Matrix*: An attempt of modeling organizational structures can be found in [2]. In this case, the meta-language UML is used. In the context of the EM-Cube we would like to recommend to use a graphical domain language in order to model organizational structures. |

Figure 7: Human-centric Organizational Models (Part I)

| Business / IT Universe | Symbol | Domain: Illustrative remarks |
|---|---|---|
| | $M_{HBS/HIS}$ | *Services*: There are different approaches to model service landscapes. One, for example, is focussing on events as it was presented in [118], another one is looking at message streams between agents as discussed in [15]. Potential models can be build using the approach of UML profiles or formal methods like reo [15]. Here, we suggest to focus on graphical domain models that represent service nodes and their connectors and that allow to specify the different types of communication channels, which may be best addressed by a graphical domain language that is integrated with a textual domain language that serves the specification of certain parameters. |
| | $M_{HBP/HIP}$ | *Processes*: Business and IT processes are usually expressed using graphical domain languages like it is the case for event driven process chains [44]. However, there is a pleantropa of further languages available. Our recommendation in the context of the EM-Cube are graphical domain languages that soundly integrate with textual domain languages for additional specifications. |
| | $M_{HBR/HIR}$ | *Rules*: Business and IT rules can be governed by meta-models as presented in [129], or they can be processed by the help of emerging web standards like [114]. Our recommendation for modeling rules would be to go for textual domain languages. |
| | $M_{HBA/HIA}$ | *Agents*: In the case of agent modeling, we recommend to use a textual domain language. |
| | $M_{HBO/HIO}$ | *Objects*: Object in the context of the Credit Suisse scenario show up as business object models (BOMs), which are encoded using UML or UML profiles. We would like to suggest to use ontologies for the different domains instead [152, 149]. |

Figure 8: Human-centric Organizational Models (Part II)

| Business / IT Universe | Symbol | Domain: Illustrative remarks |
|---|---|---|
| Machine-centric Models | $M_{MBC/MIC}$ | *Cybernetics and Objectives*: A candidate for a textual machine language for cybernetic models is Modelica. Modelica supports hybrid simulation models, which are build on automatas and differential equations that are realized as implemented formal semantics by tools like OpenModelica or Dymola [75]. |
| | $M_{MBH/MIH}$ | *Hierachy and Matrix*: In the context of the Credit Suisse scenario organizational structures are modeled using UML or UML profiles. However, we suggest to ground domain models of organizational structures on an implemented formal semantics like f-logic [99]. Details of possible textual machine languages are left for future work. |
| | $M_{MBS/MIS}$ | *Services*: A potential candidate for a graphical machine language for service landscapes is reo [15], which has been successfullly mapped to mcrl2 [102] as an implemented formal semantics [78]. |
| | $M_{MBP/MIP}$ | *Processes*: Process models have been equipped with a pleantropa of formal semantics. In the context of the analysis of the Credit Suisse scenario we found evidence that the mcrl2 language is useful for the purpose of model checking of business processes [36]. It implements the formal semantics of an advanced process algebra encompassing a modal logic which can be used to specify and check complex system behavior [78]. |
| | $M_{MBR/MIR}$ | *Rules*: Possible candidates for a textual machine languages are rule languages that are build on first-order logic [50, 37] as an implemented formal semantics. Available implementations are, for example, Prolog systems [42] or specialized commercial products. |
| | $M_{MBA/MIA}$ | *Agents*: Agent domain models can be, for example, grounded on a formally specified speech-act semantics [29]. Possible implementations and corresponding textual machine languages are left for future work. |
| | $M_{MBO/MIO}$ | *Objects*: Possible candidates for a textual machine language are ontology specification languages as they can be found in commercial products like Ontobroker from Ontoprise. In this case, the implemented formal semantics realizes the f-logic as introduced in [99]. |

Figure 9: Machine-centric Organizational Models

| Business / IT Universe | Symbol | Domain: Illustrative remarks |
|---|---|---|
| Human-centric Models | $M_{HBPR/HIPR}$ | *Proving Regulations*: In the case of the discussed loan granting process [36] it can be, for example, imagined that a textual domain language is used to express organizational security constraints of a business process. |
| | $M_{HBOR/HIOR}$ | *Optimizing Risks*: The modeling of operational risks could be realized by using predefined control elements developed in a graphical domain language for control models [4]. Details are currently under investigation. |
| | $M_{HBTG/HITG}$ | *Testing Guidelines*: The modeling of hypotheses could be realized by the help of textual domain languages, in which properties of socio-technical systems are described. The development of such domain languages is part of the potential future work. |

Figure 10: Human-centric Organizational Constraints

| Business / IT Universe | Symbol | Domain: Illustrative remarks |
|---|---|---|
| Machine-centric Models | $M_{MBPR/MIPR}$ | *Proving Regulations*: In the case of the discussed loan granting process the organizational security constraint of the 4-eye-principle regarding the business process at stake is semantically encoded by the fully implemented modal logic, which comes with mcrl2 as a possible machine language [36]. |
| | $M_{MBOR/MIOR}$ | *Optimizing Risks*: Possible candidates for an implemented formal semantics are simulation systems that can handle hybrid simulation models by interpreting automata specifications and systems of differential equations [4]. Modelica could be considered to be a candidate of a possible machine language in this case. |
| | $M_{MBTG/MITG}$ | *Testing Guidelines*: Here, it could be promising to have a detailed look at different machine languages of computer algebra systems implementing a variety of formal semantics. However, a detailed investigation remains part of the potential future work. |

Figure 11: Machine-centric Organizational Constraints

| Business / IT Universe | Symbol | Domain: Illustrative remarks |
|---|---|---|
| Human-centric Models | $M_{HBR_e/HIR_e}$ | *Statements*: General and specific organizational statements could be expressed in a reconstructed organizational domain language [144]. |
| Machine-centric Models | $M_{MBR_e/MIR_e}$ | *Statements*: Our impression is that first-order logic could be used as a formal semantics to ground a reconstructed organizational domain language. In this case Prolog could be used [42]. |

Figure 12: Human-centric and machine-centric Organizational Requirements

## 3.4 Evaluation

In a next step, we evaluate our new modeling framework against requirements that came out of the analysis of the real-world scenario at Credit Suisse as well as from the analysis of the related work. We abstract from requirements that are only supported by the corresponding modeling technique, we suggest to use in combination with the EM-Cube. The selected requirements are summarized in the following table (see Fig. 13).

| Name | Description | Check |
|---|---|---|
| R1a | Separation of Organizational Models and IT Architecture Models | OK |
| R1b | Separation of IT Architecture Models from IT Technology Models | OK |
| R2 | Separation of Organizational Business and IT Models | OK |
| R3 | Mutual Alignment of Organizational Business and Organizational IT Models | OK |
| R4a | Mutual Alignment of Organizational IT and IT Architectural Models | OK |
| R4b | Mutual Alignment of IT Architecture Models and IT Technology Models | OK |
| R5 | Separation of Organizational Models from Organizational Constraints | OK |
| R6 | Endowing Organizational Models with Formal Semantics | OK |
| R7 | Endowing Organizational Constraints with Formal Semantics | OK |
| R8 | Endowing Formal Semantics with Formal Tools | OK |
| R9 | Use of a Service Oriented Architecture | OK |
| R15 | Support of Partial Organizational Models | OK |
| R17 | Use of a Model Framework | OK |
| R19 | Letting the concrete IT Architecture being a Parameter | OK |
| R23 | Support of Language Spaces | OK |
| R24 | Support of Scheme- and Instance-Models | OK |
| R25 | Mutual Alignment of Models plus their Constraints and Model Requirements | OK |
| R32 | Use of orthogonal Dimensional Descriptors | OK |

Figure 13: Selected Requirements for the Specification of the EM-Cube

In the EM-Cube organizational business and IT models are separated (R1a) from IT architecture models by a mutual alignment (R4a, A4). The same is true (R1b) for the separation of IT architecture models and IT technology models describing the hard- and software back-end, linked likewise by the help of alignments (R4b, A5). Organizational business models and organizational IT models are equally separated (R2) by a mutual alignment (R3, A2). Besides this, organizational models are separated from their corresponding organizational constraints (R5). Organizational business and IT models as well as their corresponding constraints are assumed to be entered as human-centric models and aligned with machine-centric models that codify aspects of their corresponding formal semantics (R6, R7). In [37, 36] we were able to show that such a formal semantics can be provided by the help of a formal tool (R8). This example was realized in the context of an earlier version of the EM-Cube. In the second version of the EM-Cube presented in this paper the architecture models decompose into platform independent and platform specific models that can be differentiated further into process services, composed services and basis services. Therefore, we assume that organizational models are aligned towards and executed by a service oriented IT architecture (R9). However, because of the explicit alignment, organizational models can easily be aligned towards and executed by any other type of IT architecture that might be appropriate in a given context. Therefore, the concrete architectural type can remain a given but flexible parameter (R19). The use of partial organizational models (R15) is supported by the concept of human-centric models, which can be incomplete and maybe even inconsistent, whereas machine-centric models are required to be well-formed. Both worlds are combined by the help of flexible and declarative correspondence rules as part of the corresponding modeling technique. In addition to that, there is an assumed bidirectional alignment between (partial) models and their constraints with corresponding model requirements (R25). All alignments are assumed to be realized by the help of declarative correspondence rules. As can be seen in Fig. 4, the new EM-Cube is a special modeling framework (R17) that soundly supports the organization of models in line with the different sub-domains identified in the Credit Suisse scenario, each representing its own language space (R23). It also supports the handling of organizational scheme and instance models (R24). Its orthogonal dimensional descriptors (R32) as discussed in [88, 89] can easily be formulated as there is a dimension for the purpose of models like organization, architecture or technology, and as there is a dimension for the nature of models like human-centric, machine-centric or software-centric.

# 4 Conformance Analysis

## 4.1 Algebraic Graph Transformation

The second main contribution is a formal solution that helps to check the mutual conformance of organizational models that are located in different parts of the EM-Cube. This solution uses formal techniques of algebraic graph transformation (AGT). It is hold compatible with requirements for the new EM-Cube.

AGT as a formal technique operates on the abstract syntax of textual and graphical (visual) languages. It is used to define implementable operations for the construction, integration, transformation and synchronization of models [53, 56, 84]. In addition to that, the formal analysis and modification of models are supported. Its powerful and automated formal analysis techniques are a major advantage compared to other approaches like meta modelling [124]. In this paper, we benefit from this special capability by using several available formal results for our implementable formal solution for conformance checks between models.

While meta modelling is purely declarative, AGT is used as a constructive but also as a declarative specification technique and even allows the combination of both [53, 64]. In all cases, the type graph specifies the main structural requirements of the instance graphs by defining the possible node, attribute and edge types and, moreover, the inheritance structures. This concept directly corresponds to the definition of the meta model in meta modelling, which describes the types of possible objects and links. The type graph can be extended by graph constraints [81, 53] that have to be satisfied by all instance graphs of the domain language, which correspond to the definition of constraints in meta modelling. Moreover, a type graph with constraints can be extended with transformation rules leading to a graph grammar. These transformation rules form the constructive part, because the rules are used to construct the possible instance graphs that additionally satisfy the constraints and this way, they generate the specified language. Thus, the modeller of a domain language can choose which domain restrictions to put in the graph constraints and which to ensure by the transformation rules automatically.

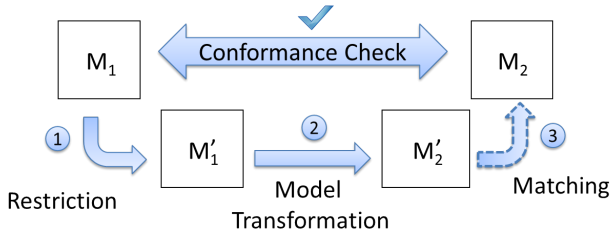


Figure 14: Conformance Check

The main concept of a conformance check using AGT is visualized in Fig. 14. Given two models $M_1$ and $M_2$, we perform three steps. At first, model $M_1$ is reduced to sub model $M_1'$ by removing those elements that are not relevant for the conformance check. In step two, model $M_1'$ is transformed into model $M_2'$, which belongs to the domain language of the given model $M_2$. The last step checks whether the generated model $M_2'$ of this transformation matches model $M_2$ in a suitable way, i.e., fulfilling some conformance conditions. In particular, model $M_2'$ does only contain information that can be computed from the information available in $M_1$. For this reason, model $M_2$ may contain additional information compared to model $M_2'$. Note that this conformance checking process is a purely syntactical operation.

This conceptual idea can be realized by AGT. In order to do this, we are required to provide a formal operation for model transformations that is correct and complete on the syntactic level and we are further required to provide a formal operation for pattern matching of models. As a consequence, a concrete conformance check can be realized in a formal way that is fully implementable.

Before giving all necessary definitions, we like to start with an illustrative extract of our compact example that should help to absorb the formalisms more easily later on. In detail, we take the given human-centric business process (BP) model in Fig. 5, which we call model $M_1$. We check the conformance of model $M_1$ with model $M_2$, which is the given (partially) integrated human-centric organizational (PIO) model in Fig 6. For this purpose, we take the type graph for the domain language of business process models and restrict it to those types that we need for our conformance check. This means that we abstract away those types that do not contribute anything in this case. This allows us to automatically reduce BP model $M_1$ to a reduced BP model $M_1'$ (RBP model), which does not contain any element typed over a removed type. In the second step, we execute a model transformation on the RBP model $M_1'$ and derive the resulting model PIO model $M_2'$, which is depicted in Fig. 24. In the last step, we perform automated pattern matching in order to check whether the generated model $M_2'$ (Fig. 24) matches the elements and structures of model $M_2$ in Fig. 6. If a valid match is found, we know that model $M_1$ conforms to model $M_2$ and if no valid match is found, we know that model $M_1$ does not conform to model $M_2$.

In order to perform the conformance check, we first have to define the relationship of the two involved domain languages and provide a fully automatic model transformation. For this purpose, we use the concept of triple graph grammars (TGGs, see [145]). A triple graph $G = (G^S \leftarrow G^C \rightarrow G^T)$ consists of three graphs, which are called source, correspondence and target graphs. The source graph $G^S$ specifies the abstract syntax of a model in the first (source) domain and the target graph $G^T$ specifies the abstract syntax of a model in the second (target) domain. The correspondence graph $G^C$ together with additional morphisms $s\colon G^C \rightarrow G^S$ and $t\colon G^C \rightarrow G^T$ specifies the correspondences between elements of $G^S$ and elements of $G^T$. This general concept allows for arbitrary correspondences between source and target elements, i.e., an element of a source model can be related to an arbitrary amount of elements of the corresponding target model.

A triple graph grammar $TGG = (TG, SG, TR)$ consists of a triple type graph $TG$, a triple start graph $SG$ and a set of triple rules $TR$. The triple type graph for the model transformation is depicted in Fig. 15 and the start graph in our example is the empty triple graph $SG = \varnothing$. The triple type graph (see Fig. 15) specifies the general structure of the concrete domain models and, moreover, the possible correspondences between the nodes of these models. It consists of a source type graph $TG^S$, a target type graph $TG^T$ and a correspondence type graph $TG^C$. The grey arrows specify the mappings between the correspondence type graph and the domain type graphs $TG^S$ and $TG^T$. Intuitively, these grey arrows specify how elements of the source domain (domain of RBP model $M_1'$) can be aligned via correspondence links with elements of the target domain (domain of PIO model $M_2$). The actual mutual correspondence links of two given models are created by the triple rules and have to respect the structure as defined in the triple type graph. For example, actors and resources of the source domain (RBP) are directly aligned with their corresponding actors and resources in the target domain (PIO). Nodes of type "Function" in the source domain correspond to nodes of type "Channel" in the target domain. Note that a type graph may contain inheritance edges (white arrow head). In the present example, this

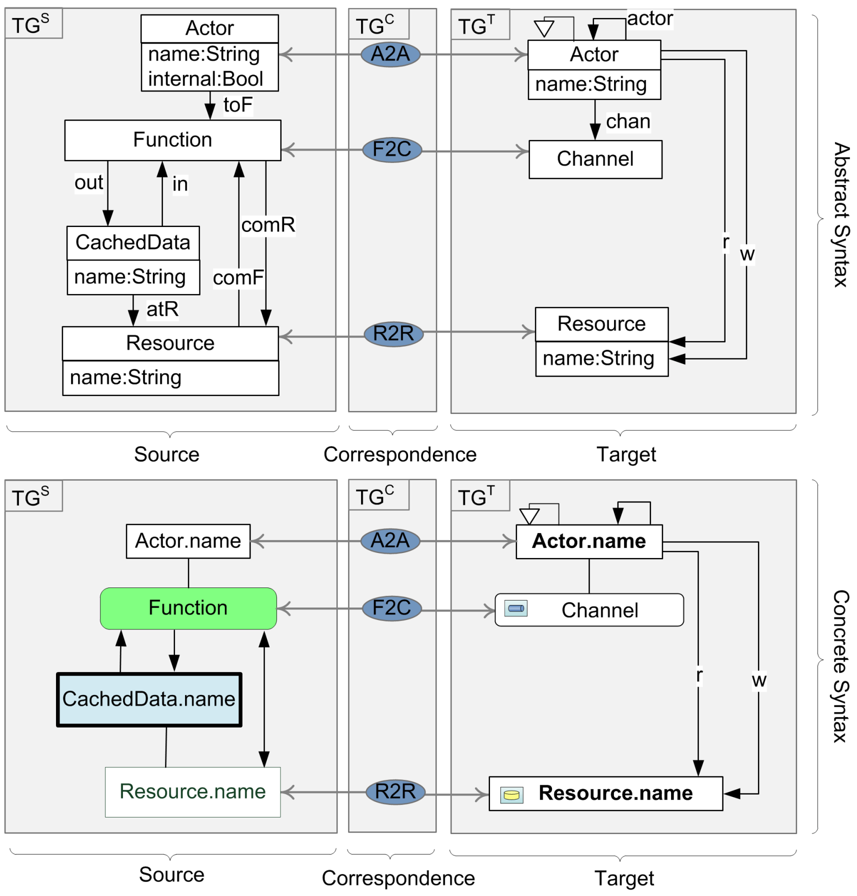


Figure 15: Triple Type Graph in abstract and concrete Syntax

is the case for the node type “Actor” in the target domain. Thus, we can define generalized roles of actors and the concrete roles inherit all features of the more general one. The role “Employee”, e.g., is used to specify that all actors of an enterprise may be in contact with a customer - see Fig. 6.

A triple rule $tr: L \to R$ consists of a triple graph $L$ (left hand side), a triple graph $R$ (right hand side) and a triple graph morphism $tr$. Triple rules specify how two models of interrelated domains can be extended synchronously. For this reason, triple rules do not delete elements and we can assume that the triple graph morphism $tr$ is given by an inclusion. Intuitively, the rules define in which way relevant concepts of the domain of model $M_1'$ are mapped into concepts of the domain of model $M_2$ and vice versa. Thus, a TGG specifies the structure of consistently interrelated models of the source and target domains.

Some of the triple rules for our example are presented in Fig. 16 and described in Ex. 1 below, while the remaining rules are presented in Sec. 5. The rules are depicted in compact notation, i.e., the left and right hand sides $L$ and $R$ are depicted as one triple graph and those elements that occur only in $R$ are marked with the label “++”, because they are created by the rule.

Given an existing triple graph, a triple rule can be applied if the left hand side $L$ of the rule, i.e., the non-marked nodes and edges, match the elements in the given triple graph. The result of a rule application is a transformation step $G_1 \Rightarrow G_2$, where those nodes and edges that are marked with "++" are added to the given triple graph $G_1$ leading to the resulting triple graph $G_2$. A triple rule may be extended by negative application conditions (NACs), which restrict the application of the rules [56]. A NAC is a triple graph that extends the left hand side of the rule by forbidden patterns. This means that a rule with NACs is not applicable, if these forbidden patterns are present in the current triple graph $G_1$. Within the figures, we mark NACs by a frame with label "NAC".

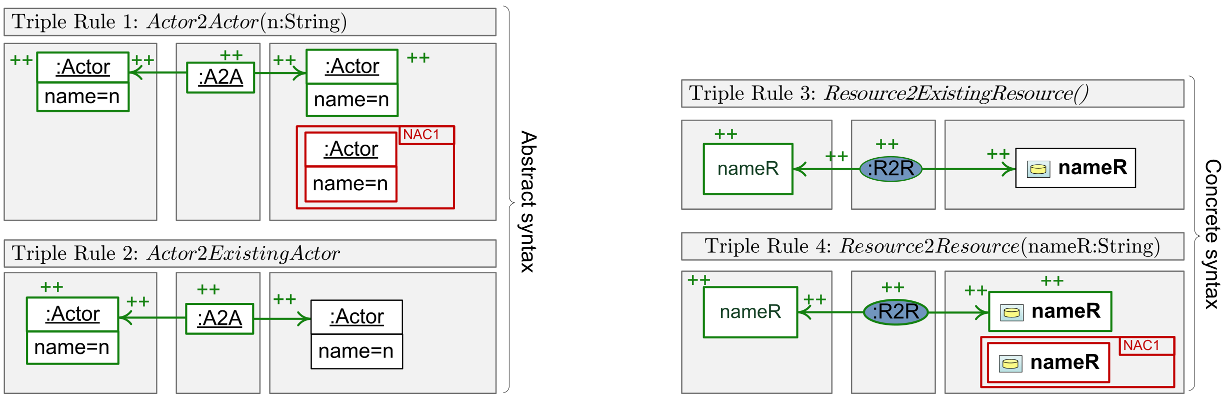

Triple Rule 1: *Actor2Actor*(n:String)
:Actor
name=n
:A2A
:Actor
name=n
NAC1
:Actor
name=n
Triple Rule 2: *Actor2ExistingActor*
:Actor
name=n
:A2A
:Actor
name=n
Abstract syntax
Triple Rule 3: *Resource2ExistingResource()*
nameR
:R2R
nameR
Triple Rule 4: *Resource2Resource*(nameR:String)
nameR
:R2R
nameR
NAC1
nameR
Concrete syntax
++

Figure 16: Triple rules in compact notation - part I

**Example 1** (Triple Rules). *The first two triple rules in Fig. 16 are depicted in abstract syntax. They specify that actors of an RBP model (business process model domain) can be related to actors of a PIO model (organizational model domain), if their names are equal. The first rule creates a new actor in the PIO component together with a corresponding new actor in the RBP component. The NAC ensures that the PIO component does not already contain an actor with the same name. The reason is that our domain language of PIO models requires that names of actors and resources are unique in the abstract syntax of PIO models. The second rule takes an existing actor in the PIO component and relates it with an new created actor in the WDEPC component of a triple graph. This means that the first rule has an empty left hand side, while the second one contains the actor node in the PIO domain. The third and forth rules are depicted in the more intuitive visual notation, i.e., in concrete syntax that is derived from the abstract syntax.*

Note that from an intuitive point of view, the triple rules of a TGG specify the consistency requirements that have to be satisfied by two interrelated models. Therefore, the design of the triple rules is essential before performing the conformance check.

In the following, we briefly review the main formal concepts of TGGs [145, 57]. A triple graph $G = (G^S \xleftarrow{s_G} G^C \xrightarrow{t_G} G^T)$ defines a source model by graph $G^S$, a target model by graph $G^T$ and their correspondences given by graph $G^C$ linked to the source and target models by graph morphisms $s_G$ and $t_G$.

$$\begin{tikzcd}
G = (G^S \arrow[d, "m"'] & G^C \arrow[l, "s_G"'] \arrow[r, "t_G"] \arrow[d, "m^C"'] & G^T) \arrow[d, "m^T"] \\
H = (H^S & H^C \arrow[l, "s_H"] \arrow[r, "t_H"'] & H^T)
\end{tikzcd}$$

$m^S \downarrow$ (=) (=)

A triple graph morphism $m = (m^S, m^C, m^T) : G \to H$ defines a mapping between two triple graphs and consists of three graph morphisms - one for each component. Moreover, a triple graph morphism is required to be compatible with the structure of $G$ and $H$, i.e. with the internal morphisms $s_G, t_G, s_H$, and $t_H$: $(m^S \circ s_G = s_H \circ m^C) \wedge (m^T \circ t_G = t_H \circ m^C)$.

A triple rule $tr : L \to R$ is given by an injective triple graph morphism $tr$. This means that triple rules are non-deleting, because the left hand side $L$ is completely included in the right hand side $R$. This is sufficient, because triple rules are used for the specification of consistent integrated models and not for the editing of source and target models.

$$\begin{tikzcd}
L \arrow[r, hook, "tr"] \arrow[d, "m"'] \arrow[dr, phantom, "(PO)"] & R \arrow[d, "n"] \\
G \arrow[r, hook, "t"'] & H
\end{tikzcd}$$

In order to apply a triple rule $tr$ to a triple graph $G$ we have to find a triple graph morphism $m : L \to G$ from the left hand side $L$ into $G$. The match specifies the location at which the rule is applied. A rule application is given by a triple graph transformation step $G \xRightarrow{tr,m} H$ from $G$ to $H$ via triple rule $tr$ and match $m$. Roughly spoken, the image $m(L)$ of the left hand side $L$ in $G$ is replaced by $R$ leading to the resulting triple graph $H$. The formal categorical construction is a pushout in the category of triple graphs (see [57] and [56, 83] concerning rules with NACs).

Finally, a TGG generates the language of integrated models $VL$: $VL = \{G \mid \varnothing \xRightarrow{tr^*} G$ via $TR\}$. Therefore, TGGs provide a constructive and declarative pattern based specification formalism for the language of consistent integrated models. Moreover, we derive the language of consistent source models $VL_S$ and consistent target model $VL_T$ according to the TGG by restricting the integrated models in $VL$ to either the source domain ($VL_S$) or the target domain ($VL_T$).

## 4.2 Model Transformation

The operational rules for the forward and backward directions of bidirectional model transformations based on TGGs are derived automatically from the TGG itself [56]. In the present scenario, we use consider the forward case in order to transform BP models into corresponding PIO models.

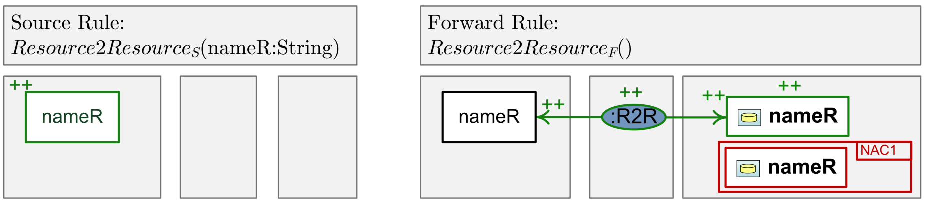


Figure 17: Derived source and forward rules for "*Resource2Resource*"

Figure 17 shows the derived source and forward rules for the triple rule "Resource2Resource()" from Fig. 16. The source rule is obtained by restricting original the triple rule to the source component, i.e., by removing all elements from the correspondence and target component. Thus, source rule "Resource2Resource()$_S$" creates a new resource in the BP domain without changing the PIO domain. The forward rule is intuitively obtained by removing all "++"-signs from the source component. This means that the forward rule reads the source component and adds the missing elements in the correspondence and target domain. The depicted forward rule "Resource2Resource()$_F$" takes an existing resource of the BP domain and adds the corresponding resource in the PIO domain, where the NAC ensures that this resource does not exist already. The formal construction is based on the redefinition of the morphisms of the triple rule according to Def. 1 below.

**Definition 1** (Derived Triple Rules)**.** *Given a triple rule $tr = (tr^S, tr^C, tr^T) : L \to R$, the source rule $tr_S : L_S \to R_S$ and the forward rule (FW-rule) $tr_F : L_F \to R_F$ are derived according to the diagrams below.*

$$\begin{tikzcd}
L = ( \arrow[d, "tr"'] & L^S \arrow[d, "tr^S"'] & L^C \arrow[l, "s_L"'] \arrow[r, "t_L"] \arrow[d, "tr^C"'] & L^T) \arrow[d, "tr^T"] \\
R = ( & R^S & R^C \arrow[l, "s_R"] \arrow[r, "t_R"'] & R^T)
\end{tikzcd}$$

*triple rule $tr$*

$$\begin{tikzcd}
L_S = ( \arrow[d, "tr_S"'] & L^S \arrow[d, "tr^S"'] & \varnothing \arrow[l] \arrow[r] \arrow[d] & \varnothing) \arrow[d] \\
R_S = ( & R^S & \varnothing \arrow[l] \arrow[r] & \varnothing)
\end{tikzcd}$$

*source rule $tr_S$*

$$\begin{tikzcd}
L_F = ( \arrow[d, "tr_F"'] & R^S \arrow[d, "id"'] & L^C \arrow[l, "tr^S \circ s_L"'] \arrow[r, "t_L"] \arrow[d, "tr^C"'] & L^T) \arrow[d, "tr^T"] \\
R_F = ( & R^S & R^C \arrow[l, "s_R"] \arrow[r, "t_R"'] & R^T)
\end{tikzcd}$$

*forward rule $tr_F$*

The beauty of these definitions can be directly grasped by looking at how easily the morphisms can be shifted around by going from the definition of triple rules to the definition of forward rules ($s_{L_F} = tr^S \circ s_L$). Likewise, the source rules are the result of a simple restriction of the triple rules to the source component.

From a conceptual point of view the model transformation based on forward rules assumes a source graph that is transformed into a target graph by applying forward transformation rules. In detail, the given source graph is extended by an empty part for the correspondence nodes and an empty part for the target graph. Then, the forward rules operate on the integrated graph until the target graph is fully created. Finally, the target graph is obtained by removing the source and correspondence graphs from the integrated triple graph. The formal definition is given next, where the execution of the forward transformation rules is controlled by a special control condition called "source consistency" (see [57]). Intuitively, this condition ensures that forward rules are not applied twice at the same structure and each element of the source model is translated exactly once, such that the resulting integrated model is a consistent triple graph according to the TGG. Note that the application of forward rules without an equivalent control condition does not ensure these properties. An efficient execution algorithm respecting source consistency is presented in [83].

**Definition 2** (Model Transformation based on FW-Rules)**.** *A* model transformation sequence *$(G^S,\ G_0 \xRightarrow{tr_F^*} G_n, G^T)$ based on FW-rules consists of a source graph $G^S$, a target graph $G^T$, and a source consistent forward sequence $G_0 \xRightarrow{tr_F^*} G_n$ with $G^S = G_0^S$ and $G^T = G_n^T$.*
*A* model transformation *$MT : VL(TG^S) \Rrightarrow VL(TG^T)$ is defined by all model transformation sequences $(G^S, G_0 \xRightarrow{tr_F^*} G_n, G^T)$ based on FW-rules with $G^S \in VL(TG^S)$ and $G^T \in VL(TG^T)$.*

In order to be sure that the resulting integrated model of a model transformation is consistent for a given valid source model we have to require that our model transformation based on forward rules is *correct.* Moreover, we require *completeness* in the sense that the forward model transformation leads to a corresponding valid target model for any given valid source model.

**Definition 3** (Syntactical Correctness and Completeness)**.** *A model transformation* $MT : VL(TG^S) \Rrightarrow VL(TG^T)$ *based on forward rules is*

- syntactically correct*, if for each model transformation sequence* $(G^S, G_0 \xRightarrow{tr_F^*} G_n, G^T)$ *via* $MT$ *there is* $G \in VL$ *with* $G = (G^S \leftarrow G^C \rightarrow G^T)$ *implying further that* $G^S \in VL_S$ *and* $G^T \in VL_T$*, and it is*
- complete*, if for each* $G^S \in VL_S$ *there is a model transformation sequence* $(G^S, G_0 \xRightarrow{tr_F^*} G_n, G^T)$ *via* $MT$*.*

Definition 3 above means that every model transformation based on forward rules (Def. 2) is syntactically correct if it leads to valid triple graphs. It is complete if for each integrated model the contained source model can be transformed into the target model of the integrated model. As stated by Thm. 1 below, both properties hold for all model transformations based on forward rules according to Def. 2 using the control condition “source consistency”. The formal proof is given by the proof for Thm. 2 in [56] based on the provided on-the-fly execution algorithm. We can therefore rely on both properties for the conformance checks.

**Theorem 1** (Syntactical Correctness and Completeness)**.** *Each model transformation* $MT : VL(TG^S) \Rrightarrow VL(TG^T)$ *based on forward rules is syntactically correct and complete.*

However, a model transformation based on forward rules $MT\colon VL(TG^S) \Rrightarrow VL(TG^T)$ is in general not a function from $VL(TG^S)$ to $VL(TG^T)$. But there are sufficient criteria for ensuring functional behaviour and efficient executions as presented in [83]. The analysis of these criteria can be performed automatically using the critical pair analysis engine of AGG [5] and additional automated optimization techniques [83].

## 4.3 Conformance Analysis Based on TGGs

In order to perform conformance checks between different human-centric models of the EM-Cube, we now introduce the formal notions and results for conformance analysis based on TGGs and apply them to our case study in Sec. 5.

From a conceptual point of view, a source model conforms to a target model if both models can be integrated according to the given TGG, i.e., if there is a consistent integrated model that aligns both given models.

**Definition 4** (Conformance)**.** *Let* $TGG = (TG, \emptyset, TR)$ *be a triple graph grammar. A model* $M_1 \in VL_S$ *in the source language* conforms *to a model* $M_2 \in VL_T$ *in the target* *language if and only if there is an integrated model $(M_1 \leftarrow M^C \rightarrow M_2) \in VL$, i.e. $(M_1 \leftarrow M^C \rightarrow M_2)$ is consistent according to the TGG.*

In the case that two heterogeneous domains are completely interrelated according to a TGG, conformance can be analysed directly via Def. 4 using the derived model integration techniques for TGGs [51] and existing TGG tools [100, 146]. Given a model $M_1$ of the source language and a model $M_2$ of the target language, then $M_1$ conforms to $M_2$ if they can be integrated to a consistent triple graph.

However, in the general case, like in our case study, the situation is more complex. The given source model (the BP model in Fig. 5) does not contain all necessary information for deriving the target model (the PIO model in Fig. 6). In particular, our BP model in Fig. 5 does not provide information about the inheritance relation within the PIO model in Fig. 6. For this reason, we consider a more general notion of conformance called *forward conformance*. Model $M_2$ (the PIO model in Fig. 6) is not required to be a model of the target language $VL_T$. It is only required to be typed over the type graph $TG^T$ of the target domain. $M_1$ forward conforms to $M_2$, if there is a model $M_2'$ that corresponds to $M_1$ and its elements can be mapped to the elements of $M_2$ in a consistent way. The conditions for this mapping are defined separately by a set $\mathcal{M}_C$ of triple morphisms, called conformance morphisms.

**Definition 5** (Forward Conformance)**.** *Let $TGG = (TG, \emptyset, TR)$ be a triple graph grammar and $\mathcal{M}_C$ be the class of conformance morphisms. A model $M_1 \in VL_S$ in the source language* forward conforms *to a model $M_2$ with respect to TGG and $\mathcal{M}_C$ iff $M_2$ is typed over $TG^T$ and there is a model $M_2' \in VL_T$, such that $M_1$ conforms to $M_2'$ and there is a* conformance morphism *$m \in \mathcal{M}_C$ with $m : M_2' \rightarrow M_2$.*

In our case study, the class $\mathcal{M}_C$ of conformance morphisms consists of all morphisms that are identities on attribute values, i.e. attribute assignments are preserved. This allows in particular that PIO' models may specify permitted communication and access rights on a more abstract level using the inheritance information.

By Theorem 2 below, we can conclude that conformance checks can be fully automated. A source model $M_1$ forward conforms to the target model $M_2$ in those cases in which there is a model transformation sequence from a source model $M_1$ to an intermediate target model $M_2'$ and a conformance morphism between the intermediate target model $M_2'$ and the target model $M_2$. The conformance morphisms are obtained by generating all matches via standard pattern matching of graph transformation tools and eliminating those that violate the specified conformance criteria. This criteria can be specific for the concrete application domain and we present a quite general and often appropriate one in our example in Sec. 5.

**Theorem 2** (Forward Conformance)**.** *Given a triple graph grammar $TGG = (TG, \emptyset, TR)$ with derived model transformation $MT : VL_S \rightarrow VL_T$ based on forward rules, a class of conformance morphisms $\mathcal{M}_C$, a source model $M_1 \in VL_S$ and a target model $M_2$ typed over $TG^T$. Then, $M_1$ forward conforms to $M_2$ with respect to TGG and $\mathcal{M}_C$ iff there is a model transformation sequence $(M_1, G_0 \xRightarrow{tr_F^*} G_n, M_2')$ via MT and a conformance morphism $m \in \mathcal{M}_C$ with $m : M_2' \rightarrow M_2$.*

As already mentioned in the previous section, there are some elements in the BP model in Fig. 5 that are not relevant for checking conformance with respect to the PIO model in Fig. 6. Therefore, we slightly generalize the notion of forward conformance. The difference is that we first reduce the BP model in Fig. 5 by a restriction to the relevant types resulting into the RBP model and perform the conformance check for the reduced model. This idea leads to the general notion of generalized forward conformance. The model $M_1$ typed over a type graph $TG_1$ is restricted to a model $M_1|_{TG^S}$ typed over the source type graph $TG^S$ of the TGG by removing all elements in $M_1$ that are not typed over $TG^S$. Formally, a type restriction $r_1 : TG^S \hookrightarrow TG_1$ specifies that $TG^S$ is a subgraph of $TG_1$ and therefore, $TG^S$ may contain less types than $TG_1$ [58].

**Definition 6** (Generalized Forward Conformance)**.** *Let $TGG = (TG, \emptyset, TR)$ be a triple graph grammar and let $\mathcal{M}_C$ be a class of conformance morphisms, $r_1 : TG^S \hookrightarrow TG_1$ be a type restriction and let $M_1$ be a model typed over $TG_1$. Then, $M_1$* forward conforms *to $M_2 \in VL_T$ with respect to $r_1$, if the restricted model $M_1|_{TG^S}$ of $M_1$ is a model in $VL_S$ and $M_1|_{TG^S}$ forward conforms to $M_2$ with respect to $TGG$ and $\mathcal{M}_C$.*

We now end up with the following procedure according to the visualization in Fig 14:

1. The given model $M_1$ (in our case the BP model in Fig. 5) is restricted to $M_1'$ (the RBP model), i.e., elements of non-relevant type are removed.

2. The model transformation is executed with input $M_1'$ (the RBP model) and in the case of success we know by Thm. 1 that $M_1'$ is a model in $VL_S$.

3. Finally, we can apply Thm. 2 and check for (complete) conformance of $M_2'$ to $M_2$, in our case the PIO model in Fig. 24, where existence of a conformance morphism is performed by a match search available in graph transformation engines [5].

In the next section, we discuss that the presented domain model BP satisfies the notion of generalized forward conformance concerning the given PIO model and for full technical details of the case study we refer to [82].

Summing up, the notion of generalized forward conformance is general enough for analysing conformance of heterogeneous models with loose coupling, i.e., each of the modelling domains may contain exclusive information not relevant for the other domains and the analysis is performed using verified formal techniques of algebraic graph transformation that can be fully implemented.

## 4.4 Evaluation

In a next step, we evaluate the presented modeling technique against the requirements that came out of the analysis of the real-world scenario at Credit Suisse as well as out of the analysis of the related work. The requirements concerning only the new EM-Cube were evaluated in the previous section.

The types of graphs that are able to be transformed by the help of algebraic graph transformation (AGT) are equally well suited to represent the abstract syntax of textual

| Name | Description | Check |
|---|---|---|
| R10 | Use of a Generic Formal Syntax Model for all Types of Languages | OK |
| R11 | Use of Implementable Formal Syntax Transformation Techniques | OK |
| R12 | Support for Conformance Checks between Models | OK |
| R13 | Support for Language Families | OK |
| R14 | Support for bidirectional Model Transformation and Integration | OK |
| R15 | Support of Partial Models | OK |
| R16 | Support for loose Coupling of Models | OK |
| R20 | Support for Language-Critical Reconstruction Techniques | OK |
| R21 | Support for Agile Modeling Techniques | OK |
| R22 | Support for Model Transformation Techniques | OK |
| R25 | Mutual Alignment of Models and their Requirements | OK |
| R26 | Use of Formal Syntax Analysis Techniques | OK |
| R27 | Reuse of already Implemented Formal Semantics | OK |
| R28 | Support for Modeling and Programming Languages | OK |
| R29 | Support for Graphical and Textual Domain Languages | OK |
| R30 | Support for Formal Languages | OK |
| R31 | Support for Language Integration | OK |

Figure 18: Selected Requirements for Algebraic Graph Transformation

as well as graphical (visual) languages (R10) [36]. Available implementations of AGT like MOFLON [117], AGG [5] and AGG-M [33] show that the formal syntax transformation technique can be implemented effectively (R11). The modeling technique supports further formal conformance checks between models as it was developed in the scope of this article (R12). These checks were executed using the tool AGG. Since AGT equally well supports textual and graphical (visual) languages, they can be easily used in an integrated fashion as discussed in [36], which supports the management of language families build upon textual and graphical languages (R13). In addition to that, the modeling technique is able to support bidirectional model transformation [83] as well as model integration [51, 100] using the same set of declarative triple rules (R14, R22). We claim that these features enable rule reuse and, therefore, lead to the opportunity of lower costs, higher flexibility and better consistency compared with generator driven approaches. Because transformation and integration techniques also work on partial graphs as well as partial type graphs, partial models are supported by AGT (R15) [59]. The loose coupling of models (R16) is realized by the mathematical relationship between source and target models in the context of model transformation and integration [33]. In addition to that, we believe that language-critical reconstruction techniques as discussed in [34] show a good potential to be soundly supported by formal graph transformation techniques on the level of the abstract syntax graphs of textual and graphical (visual) languages (R20). We further believe that this property holds also for agile modeling techniques [13]. In particular, agile modeling benefits from the rule-based and declarative nature of transformation systems, which can also work on partial models and partial type graphs (R21). Models and their constraints can be mutual aligned with corresponding requirements (R25) using declarative triple rules [34] on different levels of abstraction. The suggested modeling technique of algebraic graph transformation supports formal syntax analysis techniques (R26) concerning, e.g., functional behavior [85] and information preservation [57]. In particular, the analysis of functional behaviour is used to optimize the efficiency of the execution of model transformations [83], while the analysis of information preservation is used to ensure the quality and precision of a model transformation [57]. An already implemented formal semantics of a formal tool can be interfaced by defining a graph grammar for the formal specification language used by such a tool in order to create machine-centric models in such a language that represents a certain semantic specification (R27). A related example regarding the formal semantics of business processes is discussed in [36]. Both, modeling languages and programming languages, can be textual or graphical (visual). Their corresponding language specification can therefore be encoded by the help of graph grammars and type graphs, which are part of the suggested modeling technique (R28) [65]. The same holds for graphical and textual domain languages (R29), which may even evolve [30], and which could be supported by applying the same formal graph transformation techniques as meta-transformations on the graph grammar [55]. This applies likewise (R30) when defining formal languages that may be either textual or graphical [59]. We believe that language integration can be supported by integrating graph grammars and related triple graph grammars (R31).

## 5 Concrete Example

This section presents the further details of our running example, from which we used some parts to illustrate the main concepts in the previous sections. The example demonstrates how the EM-Cube can be instantiated using formal techniques of algebraic graph transformation in order to check for the mutual conformance of organizational models.

At first, we complete the triple graph grammar that is used to specify how RBP models (source domain) are consistently aligned with PIO models (target domain). For this purpose, we present the complete set of triple rules and depict again the triple type graph in Fig. 19 (also shown in Fig. 15).

Figure 20 recalls the first four triple rules, which we introduced in Sec. 4 before. They are used for aligning actors and resources in the source domain (RBP models) with their corresponding counterpart in the target domain (PIO models). In a similar way, rule "Function2Channel" in Fig. 21 is used to synchronously create a function in an RBP model and its corresponding channel in the PIO model. The rule "execute" does not create an explicit correspondence node, but is based on already existing correspondences. The effect of the rule "execute" is that an actor in the RBP model is connected to a function he is executing and at the same time, the corresponding actor in the PIO component is connected with the corresponding channel. Note that a channel is intended to specify the possible communication of an actor with those actors that are involved in the execution of the same function.

The next rules already coincide with one kind of their operational rules (either source

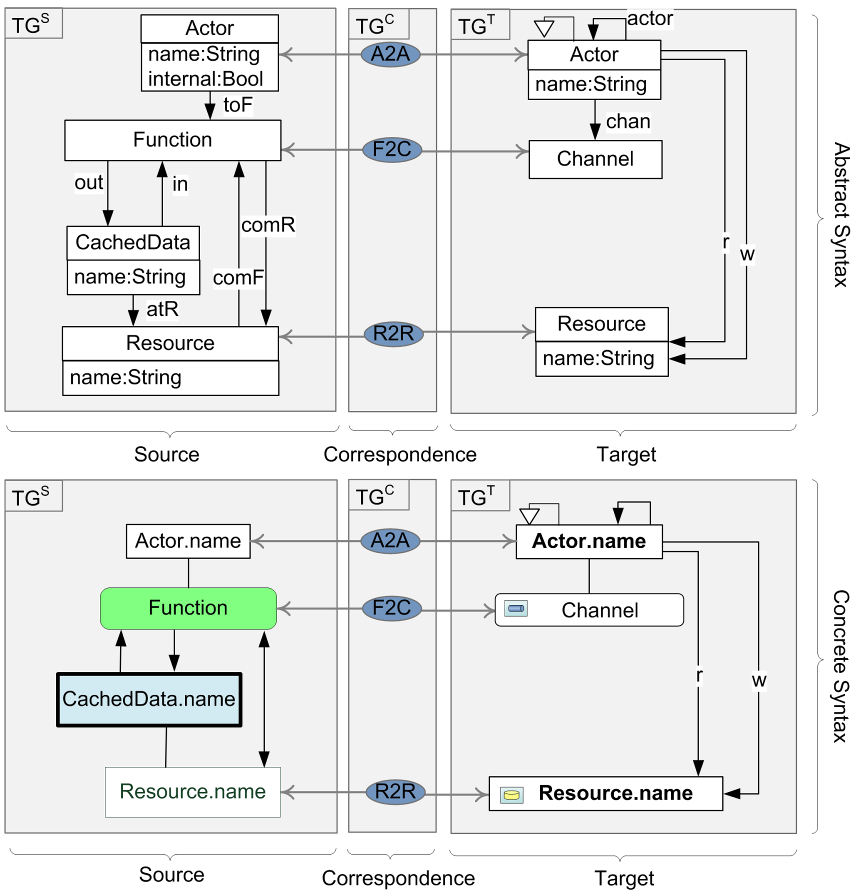
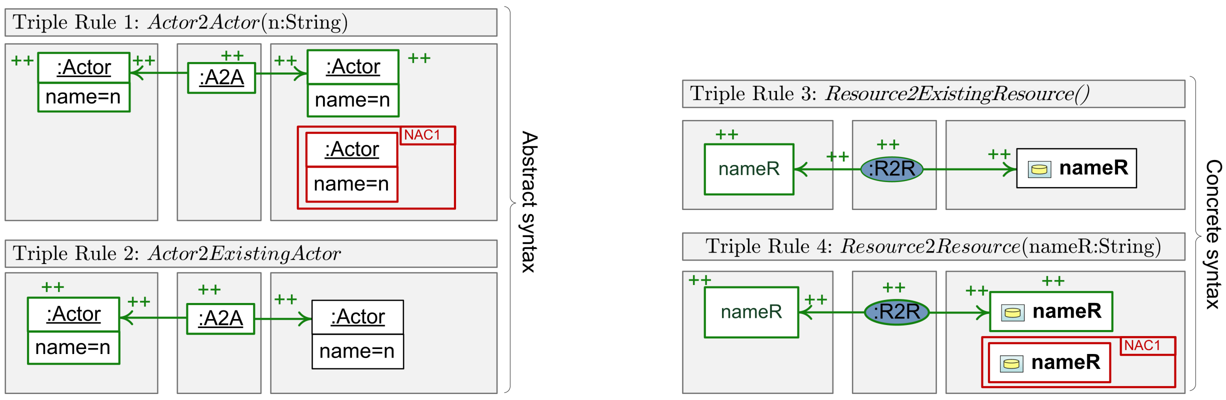


Figure 19: Triple Type Graph in abstract and concrete Syntax



Triple Rule 1: Actor2Actor(n:String)
++
:Actor
name=n
:A2A
NAC1
Triple Rule 2: Actor2ExistingActor
Abstract syntax
Triple Rule 3: Resource2ExistingResource()
nameR
:R2R
Triple Rule 4: Resource2Resource(nameR:String)
Concrete syntax

Figure 20: Triple rules in compact notation and abstract/concrete syntax - part I

or forward rule). These rules are used in order to reduce the amount of necessary triple rules. We could equivalently use triple rules that are possible combinations of rules 7-10 with rules 11-13. However, the rules would be quite complex and the difference between them would be rather small.

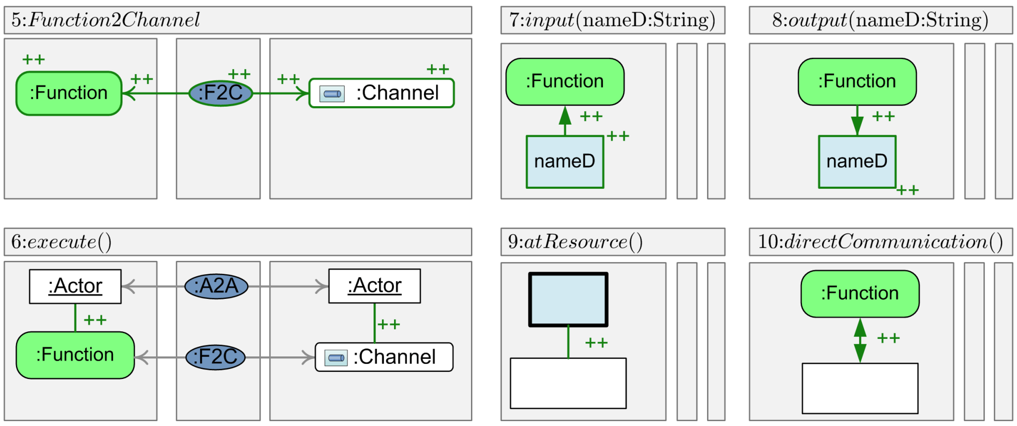


Figure 21: Triple rules in compact notation and concrete syntax - Part II

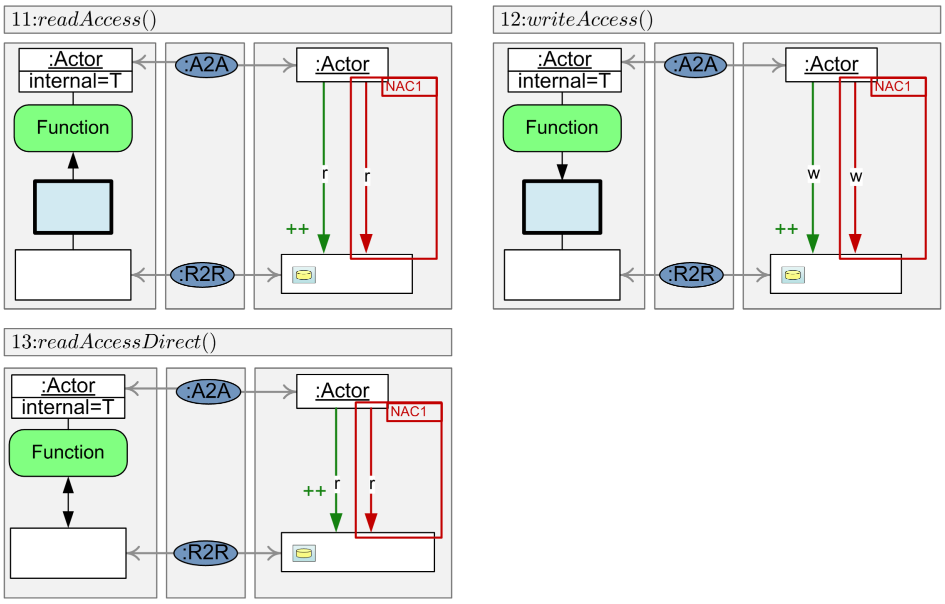


Figure 22: Triple rules in compact notation and concrete syntax - Part III

Each possible data flow path between a function and a resource shall correspond to a corresponding write and read access right in the PIO model for each actor that is attached to the function in the RBP model. At first, the triple rules in Fig. 21 are used to create the data elements and their links in the RBP model. Rule “input” requires a node of type “Function” and creates a new data element and the connecting input link. This means that the data element is read by the function. In a similar way, rule “output” concerns data elements that are written by a function in an RBP model. Rule “atResource” takes a given data element and a resource and connects them. Finally, rule “directCommunication” establishes a direct link between a function and a resource, i.e., the actual data elements are not specified.

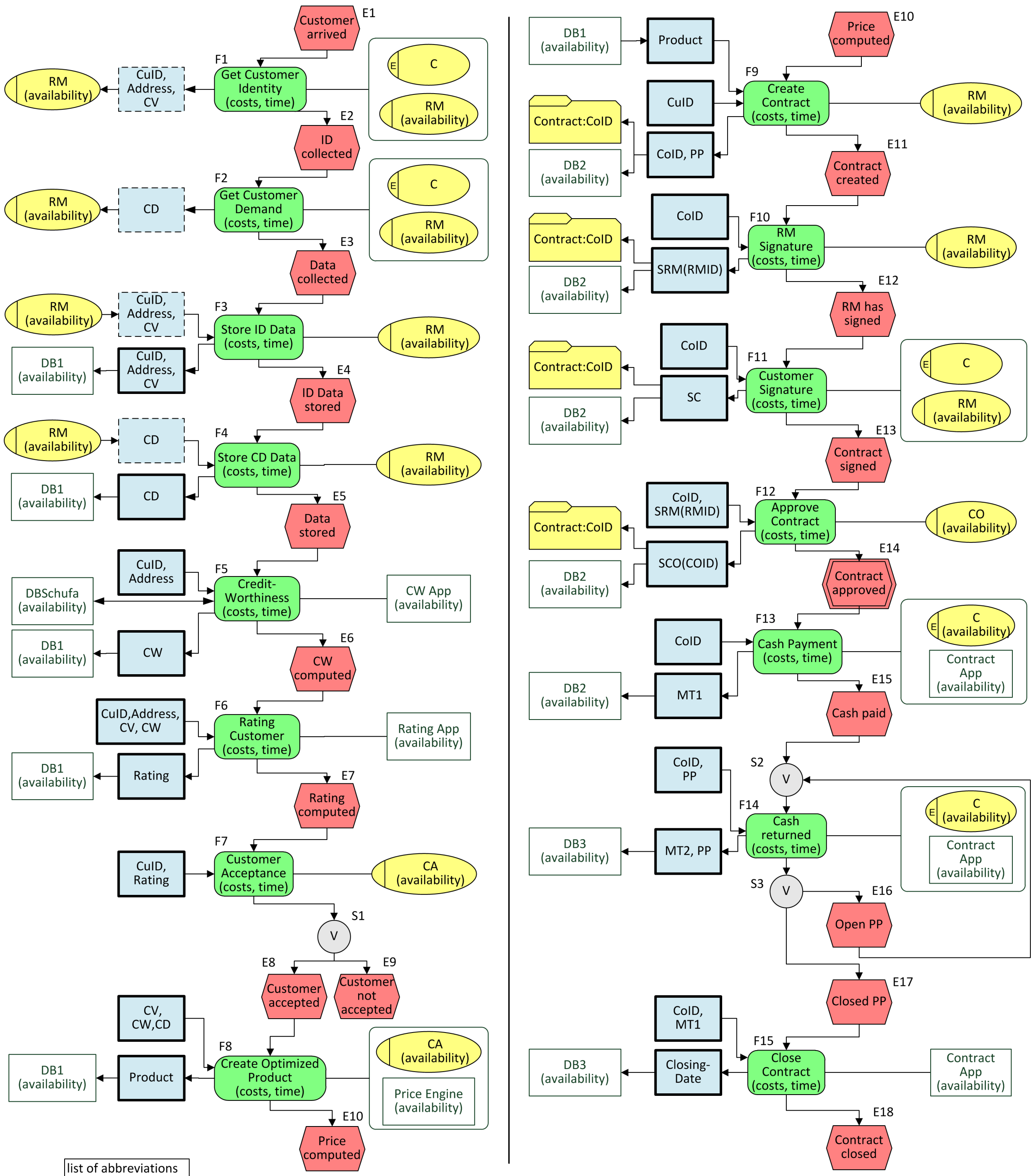
Based on the established data flow structures that are created by rules 7-11, the three rules in Fig. 22 create the corresponding access rights of actors to resources. Rules "readAccess" and "readAccessDirect" require a data flow into a function node and create a link of type “r” that specifies that the actor has read access to this resource. Similarily, rule “wirteAccess” looks for a data element that is written into a resource and establishes a link of type “w” that specifies that the actor has write access to this resource. All these three rules are equipped with negative application conditions (NACs, [56]) that are indicated by a frame with label “NAC1”. A NAC specifies a pattern that is forbidden when applying the rule. This ensures that the rules are not applied again, i.e., when a corresponding access right does already exist for the particular actor.

Summing up, the triple rules specify in a compact and visual way, how RBP models shall be aligned to PIO models. Based on the triple rules, the operational rules for executing the model transformation are derived as described before in Sec. 4. These operational rules build the basis for performing conformance checks between RBP and PIO models, which we focus on next. For the full technical details and results concerning the execution of models transformations based on TGGs and in particular concerning the TGG of this case study, we refer to Chapter 6 of [82].

**Example 2** (Conformance of Business Process Models)**.** *We analysed conformance of the BP model in Fig. 23 (also shown in Fig. 5) with the PIO model in Fig. 25 (also shown in Fig. 6) using the tool AGG. For the class $\mathcal{M}_C$ of conformance morphisms we use the set of all morphisms $m$, such that $m$ is an identity on attribute values and $m$ respects the inheritance relation in the PIO model. This means that nodes can be mapped to nodes whose type is more general. This kind of morphism respecting the inheritance relation is formalized by the notion of clan morphisms in [53]. The class of conformance morphisms is a quite natural choice, because the morphisms preserve the names while allowing for flexibility concerning models with inheritance structures [52].*

*In the first step of the analysis, we performed the type restriction on the business process (BP) model $M_1$. The corresponding type morphism $r_1 : TG^S \to TG_1$ is given by an inclusion. The restricted type graph $TG^S$ is the source component of the TGG type graph shown in Fig. 15. The type graph $TG_1$ for BP models extends $TG^S$ by events and additional adjacent edge types. Therefore, all events and adjacent edges in $M_1$ are removed leading to $M_1'$. In the next step of the conformance check, the model transformation is executed using the tool AGG [5] according to the execution approach in [83] for TGGs. The resulting partially integrated organizational (PIO) model $M_2'$ is shown in Fig. 24. Note that the channel nodes in $M_2'$ do not have name attributes, because this information is not available in RBP models. Nodes in a PIO model with the same name may occur multiple times in the concrete syntax but they refer to the same element in the abstract syntax as it is common praxis in visual modelling techniques like, e.g., UML [154]. Moreover, the private communication channels of actors (loops) are not depicted in the visual notation of PIO models in order to improve readability.*

*For the final step–the matching–we used AGG with a slight extension for computing the required conformance morphism $m : M_2' \to M_2$ from the generated PIO $M_2'$ model into the given PIO model $M_2$ in Fig. 6. Model $M_2'$ does only contain information that can be*



list of abbreviations

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| App | Application | CO | Credit Officer | MT | Money Transfer | SRM | Signature of Relationship Manager |
| C | Customer | CuID | Customer ID | PP | Payment Plan | RMID | Relationship Manager ID |
| CA | Credit Advisor | CV | Customer Value | RM | Relationship Manager | | |
| CD | Customer Demand | CW | Credit-Worthiness | SC | Signature of Customer | | |
| CoID | Contract ID | DB | Data Base | SCO | Signature of Credit Officer | | |

Figure 23: Given human-centric business process model (BP model $M_1$)

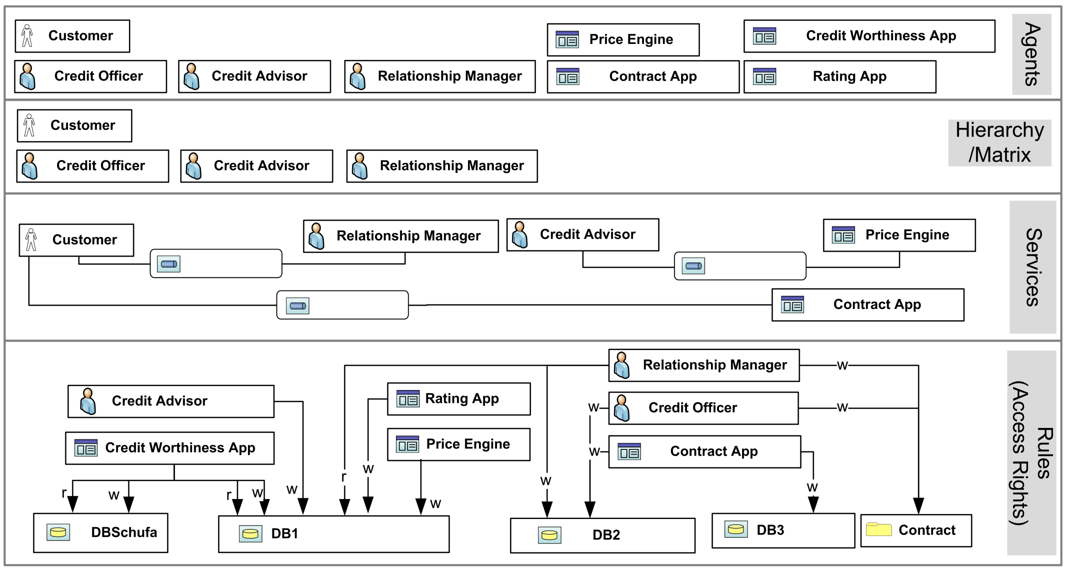


Figure 24: (Partially) integrated human-centric organizational models (PIO model $M_2'$) generated from BP model $M_1$

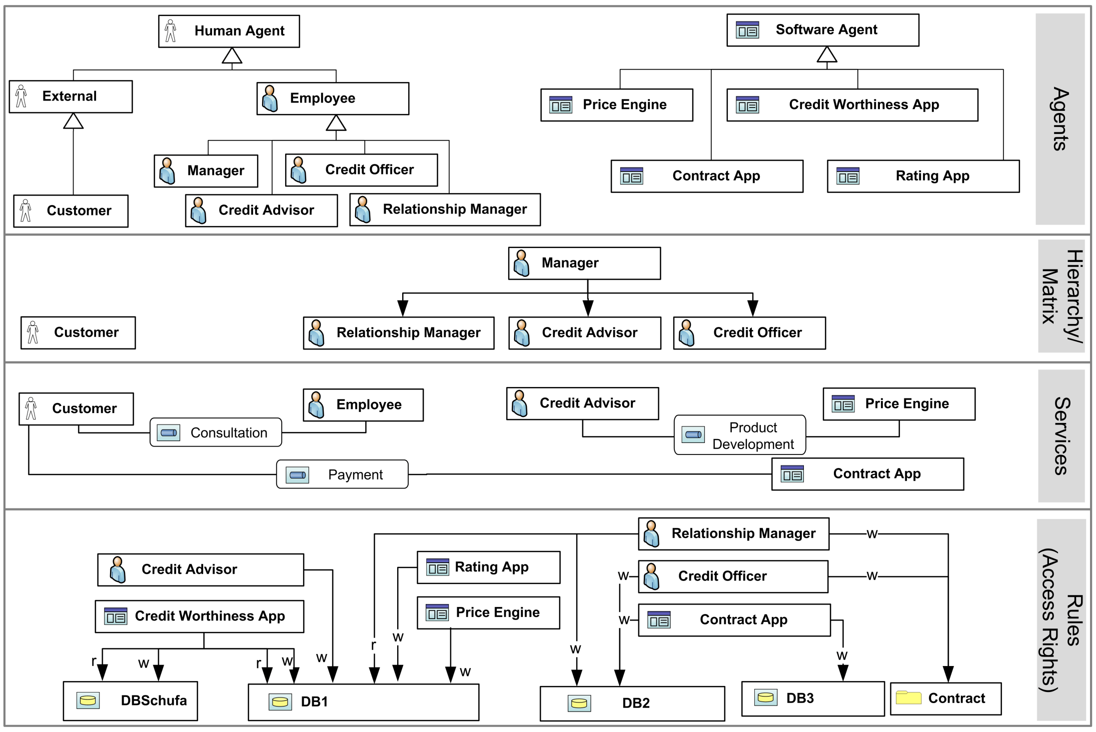


Figure 25: Given and (partially) integrated human-centric organizational models (PIO model $M_2$)

*retrieved from the BP model $M_1$. For this reason, the first two components concerning the business agents and the organizational hierarchy and matrix do only contain nodes but no* *edges that would specify a relation between business agents. The mapping $m : M_2' \to M_2$ is mainly given by the position in visual notation. In particular, the conformance morphism maps the left unnamed channel node (between the customer and the relationship manager in the communication part) to the channel node "Consultation". This mapping is compatible with the inheritance relation, because the relationship manager is an employee. Note that we are not using the abbreviations depicted in the BP model in Fig. 5, but the explicit full names according to the legend that is depicted in the bottom of the same figure. This means that we assume that the names are only abbreviated in the concrete syntax and are stored explicitly in the abstract syntax. For matching in more complex cases see Rem. 1 below.*

*All together, the match found by $AGG$ is a conformance morphism and thus, the RBP model $M_1$ in Fig. 5 conforms to the PIO model $M_2$ in Fig. 6.*

**Remark 1** (Matching of similar Names)**.** *In the present example, we considered models that are based on a fixed vocabulary of names for business objects and actor roles. In particular, we required that conformance morphisms are identities on attribute values. In the more general case, however, names may be ambigous. This means that matching has additionally to cope with an adequate handling of synonyms, homonyms and equipollences as well as false designators [62]. This would be an orthogonal extension of our presented technique and is left for future work.*

## 5.1 Evaluation

In a next step, we evaluate the concrete example against requirements that came out of the analysis of the real-world scenario at Credit Suisse as well as from the analysis of related work. We abstract those requirements that have no concrete anchor in this example. The selected requirements are summarized in the following table (see Fig. 26).

Our concrete example consists of a given human-centric business process, the BP model $M_1$ (see Fig. 5), and given and (partially) integrated human-centric organizational models, the PIO model $M_2$ (see Fig. 6), that are placed into the EM-Cube (see Fig. 4) (R17). The BP model is put into the organizational sub-domain $HBP$ (human-centric business process), the different parts of the PIO model originate from the organizational sub-domains $HBH$ (human-centric hierarchy), $HBS$ (human-centric business service), $HBR$ (human-centric business rules) and $HIR$ (human-centric IT rules) and $HBA$ (human-centric business agents) (R23). The BP model shows up formulated in a graphical domain language, whereas, for example, the rules, even so, they are represented graphically, could equivalently be expressed in a textual domain language (R29). The reduction of the BP model towards a RBP model gives an example for the support of partial models (R15). The loose coupling between the RBP model and the PIO model is realized by the help of triple rules (R16). Derived forward rules are used for the transformation of the RBP model into the PIO' model (R14, R22). Hereby, formal syntax analysis techniques are used to guarantee functional behavior of the model transformation (R26). Automatic matching between the generated PIO' model out of the RBP model and the PIO model helps to check for conformance between both models (R12). The transformations and checks can be executed automatically because the underlying formal syntax transformation techniques are implementable (R11). And because the used formal syntax model of algebraic graph transformation is generic and can encode the languages used here, the same transformation techniques and conformance checks can be applied in all cases (R10). The rule models sketch how the concept of a language family could work here. In detail, elements of a domain language for agents, elements of a domain language for business services, and elements of a domain language for the agent's rights are combined, for example, into one artifact (R13). A mutual alignment between business and IT models could be imagined for the rule models (R3), however, it is not worked out here.

| Name | Description | Check |
|---|---|---|
| R3 | Mutual Alignment of Organizational Business and Organizational IT Models | OK |
| R10 | Use of a Generic Formal Syntax Model for all Types of Languages | OK |
| R11 | Use of Implementable Formal Syntax Transformation Techniques | OK |
| R12 | Support of Conformance Checks between Models | OK |
| R13 | Support of Language Families of Small Domain Languages | OK |
| R14 | Support of bidirectional Model Transformation and Integration | OK |
| R15 | Support of Partial Models | OK |
| R16 | Support of loose Coupling of Models | OK |
| R17 | Use of a Model Framework | OK |
| R22 | Support of Model Transformation Techniques | OK |
| R23 | Support of Language Spaces | OK |
| R26 | Use of Formal Syntax Analysis Techniques | OK |
| R29 | Support of Graphical and Textual Domain Languages | OK |

Figure 26: Selected Requirements for the Concrete Example

## 6 Conclusions and Future Work

The main conclusion is that the EM-Cube and algebraic graph transformation mutually support each other. Without the formal modeling technique the EM-Cube could not come up with all the aligned organizational sub-domains encompassing organizational models that should be mutually conforming. Without the EM-Cube the formal techniques of algebraic graph transformation would not necessarily lead to such organizational sub-domains that require a solution of how to deal with conformance issues.

Promising future work shows up in the area of future versions of the EM-Cube as well as in the area of new model operations based on algebraic graph transformation. In addition to the presented formal results and automated techniques concerning syntactical correctness of model transformations, there is also a need for similar results concerning semantical correctness, where first promising results are presented in [86, 54].

## References

[1] *12th International IEEE Enterprise Distributed Object Computing Conference, ECOC 2008, 15-19 September 2008, Munich, Germany.* IEEE Computer Society, 2008. ISBN: 978-0-7695-3373-5.

[2] W. M. P. van der Aalst, Akhil Kumar, and H. M. W. Verbeek. "Organizational modeling in UML and XML in the context of workflow systems". In: *Proceedings of the 2003 ACM symposium on Applied computing.* SAC '03. Melbourne, Florida: ACM, 2003, pp. 603–608. ISBN: 1-58113-624-2. DOI: `http://doi.acm.org/10.1145/952532.952652`. URL: `http://doi.acm.org/10.1145/952532.952652`.

[3] W.M.P. van der Aalst, Marlon Dumas, C. Ouyang, Anne Rozinat, and H. M. W. Verbeek. "Choreography Conformance Checking: An Approach based on BPEL and Petri Nets". In: *The Role of Business Processes in Service Oriented Architectures.* Ed. by Frank Leymann, Wolfgang Reisig, Satish R. Thatte, and Wil van der Aalst. Dagstuhl Seminar Proceedings 06291. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), 2006.

[4] Tarek Abdelzaher, Yixin Diao, Joseph L. Hellerstein, Chenyang Lu, and Xiaoyun Zhu. "Introduction to Control Theory And Its Application to Computing Systems". In: *Performance Modeling and Engineering.* Ed. by Zhen Liu and Cathy H. Xia. Springer, 2008, pp. 185–215. ISBN: 978-0-387-79361-0.

[5] *AGG.* `http://tfs.cs.tu-berlin.de/agg`, retrived on 06-FEB-2012. TFS-Group, TU Berlin. 2011.

[6] S. Aier and M. Schönherr. "Integrating an Enterprise Architecture using Domain Clustering". In: *Proceedings of the Second Workshop on Trends in Enterprise Architecture Research (TEAR 2007) in conjunction with the European Conference on Information Systems (ECIS2007).* Ed. by M. Lankhorst and P. Johnson. 2007, pp. 20–30.

[7] Stephan Aier and Bettina Gleichauf. "Application of Enterprise Models for Engineering Enterprise Transformation". In: *Enterprise Modelling and Information Systems Architectures* 5.1 (2010), pp. 58–75.

[8] Stephan Aier, Stephan Kurpjuweit, Jan Saat, and Robert Winter. "Enterprise Architecture Design as an Engineering Discipline". In: *AIS Transactions on Enterprise Systems* 1.1 (2009), pp. 36–43. URL: `http://www.alexandria.unisg.ch/EXPORT/DL/50816.pdf`.

[9] Stephan Aier, Stephan Kurpjuweit, Otto Schmitz, Jörg Schulz, Andre Thomas, and Robert Winter. "An Engineering Approach to Enterprise Architecture Design and its Application at a Financial Service Provider". In: *MobIS.* Ed. by Peter Loos, Markus Nüttgens, Klaus Turowski, and Dirk Werth. Vol. 141. LNI. GI, 2008, pp. 115–130.

[10] Stephan Aier, Christian Riege, and Robert Winter. "Classification of Enterprise Architecture - An Exploratory Analysis". In: *Enterprise Modelling and Information Systems Architectures* 3.1 (2008), pp. 14–23.

[11] Stephan Aier and Robert Winter. "Virtual Decoupling for IT/Business Alignment - Conceptual Foundations, Architecture Design and Implementation Example". In: *Business & Information Systems Engineering* 1.2 (2009), pp. 150–163.

[12] J. Alves-Foss. *Formal syntax and semantics of Java.* Lecture notes in computer science. Springer, 1999. ISBN: 9783540661580. URL: http://books.google.de/books?id=_Ed9tKi7blIC.

[13] Scott Ambler. *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process.* John Wiley & Sons, Inc., 2002. ISBN: 047127190X.

[14] Víctor Anaya, Giuseppe Berio, Mounira Harzallah, Patrick Heymans, Raimundas Matulevicius, Andreas L. Opdahl, Hervé Panetto, and Maria Jose Verdecho. "The Unified Enterprise Modelling Language - Overview and further work". In: *Computers in Industry* 61.2 (2010), pp. 99–111.

[15] Farhad Arbab. "Reo: A Channel-based Coordination Model for Component Composition". In: *Mathematical Structures in Computer Science* 14.3 (2004). Preprint available at http://homepages.cwi.nl/~farhad/MSCS03Reo.pdf, doi:10.1017/S0960129504004153, pp. 329–366.

[16] Farhad Arbab, Frank S. de Boer, Marcello M. Bonsangue, Marc M. Lankhorst, Erik Proper, and Leendert van der Torre. "Integrating Architectural Models - Symbolic, Semantic and Subjective Models in Enterprise Architecture". In: *Enterprise Modelling and Information Systems Architectures* 2.1 (2007), pp. 40–57.

[17] K. Arnold, J. Gosling, and D.C. Holmes. *The Java programming language.* The Java series. Addison-Wesley, 2006. ISBN: 9780321349804. URL: http://books.google.de/books?id=HJUkJJomJZQC.

[18] David Aveiro. "G.O.D. (Generation, Operationalization & Discontinuation) and Control (sub)organizations: A DEMO-based approach for continuous real-time management of organizational change caused by exceptions". PhD thesis. Lisbon: Instituto Superior Técnico, Universidade Técnica de Lisboa, 2010.

[19] David Aveiro, João Mendes, and José Tribolet. "Organizational modeling with a semantic wiki". In: *Proc. of the 2008 ACM symposium on Applied computing.* SAC '08. Fortaleza, Ceara, Brazil: ACM, 2008, pp. 592–593. ISBN: 978-1-59593-753-7. DOI: http://doi.acm.org/10.1145/1363686.1363829. URL: http://doi.acm.org/10.1145/1363686.1363829.

[20] David Aveiro, A. Rito Silva, and José Tribolet. "Towards a GOD-theory for organizational engineering: continuously modeling the continuous (re)generation, operation and deletion of the enterprise". In: *Proceedings of the 2010 ACM Symposium on Applied Computing*. SAC '10. Sierre, Switzerland: ACM, 2010, pp. 150–157. ISBN: 978-1-60558-639-7. DOI: http://doi.acm.org/10.1145/1774088.1774118. URL: http://doi.acm.org/10.1145/1774088.1774118.

[21] David Aveiro, António Rito Silva, and José M. Tribolet. "Extending the Design and Engineering Methodology for Organizations with the Generation Operationalization and Discontinuation Organization." In: *DESRIST*. Ed. by Robert Winter, J. Leon Zhao, and Stephan Aier. Vol. 6105. LNCS. Springer, 2010, pp. 226–241. ISBN: 978-3-642-13334-3. URL: http://dblp.uni-trier.de/db/conf/desrist/desrist2010.html\#AveiroST10.

[22] Pavel Balabko and Alain Wegmann. "Systemic classification of concern-based design methods in the context of enterprise architecture". In: *Information Systems Frontiers* 8.2 (2006), pp. 115–131.

[23] S.A. Bernard. *An introduction to enterprise architecture*. AuthorHouse, 2005. ISBN: 9781420880502. URL: http://books.google.lu/books?id=s_vpKyas8VcC.

[24] Scott Bernard and John Grasso. "A Need for Formalization and Auditing in Enterprise Architecture Approaches and Programs". In: *Journal of Enterprise Architecture* May (2009), pp. 18–30.

[25] P. Bernus, L. Nemes, and G. Schmidt. *Handbook on enterprise architecture*. International handbooks on information systems. Springer, 2003. ISBN: 9783540003434. URL: http://books.google.lu/books?id=LTR93xiadtEC.

[26] Peter Bernus and Laszlo Nemes. "A Framework to Define a Generic Enterprise Reference Architecture and Methodology". In: *Proceedings of the International Conference on Automation, Robotics and Computer Vision (ICARCV)* (1994), pp. 88–92.

[27] D. Bjørner. *Software Engineering 3: Domains, Requirements, and Software Design*. Texts in theoretical computer science. Springer, 2006. ISBN: 9783540211518. URL: http://books.google.lu/books?id=Uti5FaDQqdQC.

[28] W. Boehmer, C. Brandt, and J.F. Groote. "Evaluation of a business continuity plan using process algebra and modal logic". In: *IEEE TIC Toronto* (2009).

[29] Rafael H. Bordini, Álvaro F. Moreira, Renata Vieira, and Michael Wooldridge. "On the Formal Semantics of Speech-Act Based Communication in an Agent-Oriented Programming Language". In: *CoRR - Computing Research Repository* abs/1111.0041 (2011).

[30] Benjamin Braatz and Christoph Brandt. “Domain-Specific Modelling Languages with Algebraic Graph Transformations on RDF”. In: *SLE*. Ed. by Brian A. Malloy, Steffen Staab, and Mark van den Brand. Vol. 6563. Lecture Notes in Computer Science. Springer, 2010, pp. 82–101. ISBN: 978-3-642-19439-9.

[31] Benjamin Braatz and Christoph Brandt. “How to Modify on the Semantic Web? - A Web Application Architecture for Algebraic Graph Transformations on RDF”. In: *ICWE Workshops*. Ed. by Florian Daniel and Federico Michele Facca. Vol. 6385. Lecture Notes in Computer Science. Springer, 2010, pp. 187–198. ISBN: 978-3-642-16984-7.

[32] Christoph Brandt, Thomas Engel, Frank Hermann, Benjamin Braatz, and Hartmut Ehrig. “An approach using formally well-founded domain languages for secure coarse-grained IT system modelling in a real-world banking scenario”. In: *Proc. Australasian Conf. on Information Systems (ACIS’07)*. 2007. URL: http://tfs.cs.tu-berlin.de/publikationen/Papers07/BBE+07.pdf.

[33] Christoph Brandt and Frank Hermann. “How Far Can Enterprise Modeling for Banking Be Supported by Graph Transformation?” In: *Int. Conf. on Graph Transformation (ICGT 2010)*. Ed. by Hartmut Ehrig, Arend Rensink, Grzegorz Rozenberg, and Andy Schürr. Vol. 6372. LNCS. Springer, 2010, pp. 3–26. ISBN: 978-3-642-15927-5.

[34] Christoph Brandt, Frank Hermann, Hartmut Ehrig, and Thomas Engel. *Enterprise Modelling using Algebraic Graph Transformation - Extended Version*. Tech. rep. 2010/06. TU Berlin, Fak. IV, 2010. URL: http://www.eecs.tu-berlin.de/menue/forschung/forschungsberichte/2010.

[35] Christoph Brandt, Frank Hermann, and Thomas Engel. “Security and Consistency of IT and Business Models at Credit Suisse realized by Graph Constraints, Transformation and Integration using Algebraic Graph Theory”. In: *Proc. Int. Conf. on Exploring Modeling Methods in Systems Analysis and Design 2009 (EMMSAD’09)*. Vol. 29. LNBIP. Heidelberg: Springer Verlag, 2009, pp. 339–352. DOI: 10.1007/978-3-642-01862-6_28. URL: http://dx.doi.org/10.1007/978-3-642-01862-6_28.

[36] Christoph Brandt, Frank Hermann, and Jan Friso Groote. “Generation and Evaluation of Business Continuity Processes; Using Algebraic Graph Transformation and the mCRL2 Process Algebra”. In: *Journal of Research and Practice in Information Technology* 43.1 (2011), pp. 65–85. ISSN: 1443-458X.

[37] Christoph Brandt, Jens Otten, Christoph Kreitz, and Wolfgang Bibel. “Specifying and Verifying Organizational Security Properties in First-Order Logic”. In: *Verification, Induction, Termination Analysis*. Ed. by Simon Siegler and Nathan Wasser. Vol. 6463. Lecture Notes in Computer Science. Springer, 2010, pp. 38–53. ISBN: 978-3-642-17171-0.

[38] C. Braun. *Modellierung der Unternehmensarchitektur: Weiterentwicklung einer bestehenden Methode und deren Abbildung in einem Meta-Modellierungswerkzeug.* Logos Verlag Berlin, 2007. ISBN: 9783832514716. URL: `http://books.google.lu/books?id=oXv8MQAACAAJ`.

[39] Sabine Buckl, Ulrik Franke, Oliver Holschke, Florian Matthes, Christian M. Schweda, Teodor Sommestad, and Johan Ullberg. "A Pattern-based Approach to Quantitative Enterprise Architecture Analysis". In: *AMCIS*. Ed. by Robert C. Nickerson and Ramesh Sharda. Association for Information Systems, 2009, p. 318.

[40] Artur Caetano, António Rito Silva, and José Tribolet. "A role-based enterprise architecture framework". In: *Proceedings of the 2009 ACM symposium on Applied Computing.* Ed. by Sung Y. Shin and Sascha Ossowski. SAC '09. Honolulu, Hawaii: ACM, 2009, pp. 253–258. ISBN: 978-1-60558-166-8. DOI: `http://doi.acm.org/10.1145/1529282.1529337`. URL: `http://doi.acm.org/10.1145/1529282.1529337`.

[41] Artur Caetano, António Rito Silva, and José M. Tribolet. "Business Process Decomposition - An Approach Based on the Principle of Separation of Concerns". In: *Enterprise Modelling and Information Systems Architectures* 5.1 (2010), pp. 44–57.

[42] Mats Carlsson and Per Mildner. "SICStus Prolog – the first 25 years". In: *CoRR - Computing Research Repository* abs/1011.5640 (2010).

[43] Sven A. Carlsson. "Developing Knowledge Through IS Design Science Research: For Whom, What Type of Knowledge, and How?" In: *Scandinavian J. Inf. Systems* 19.2 (2007).

[44] Rob Davis. *ARIS Design Platform: Advanced Process Modelling and Administration.* 1st. Springer, 2008. ISBN: 184800110X, 9781848001107.

[45] G. Doucet, J. Gøtze, P. Saha, and S. A. Bernard, eds. *Coherency Management – Architecting the Enterprise for Alignment, Agility, and Assurance.* Bloomington, AuthorHouse, 2009.

[46] G. Doucet, J. Gøtze, P. Saha, and S. A. Bernard. "Commencing the Journey: Realizing Coherency Management". In: ed. by G. Doucet, J. Gøtze, P. Saha, and S. A. Bernard. Bloomington, AuthorHouse, 2009.

[47] G. Doucet, J. Gøtze, P. Saha, and S. A. Bernard. "Introduction to Coherency Management: The Transformation of Enterprise Architecture". In: ed. by G. Doucet, J. Gøtze, P. Saha, and S. A. Bernard. Bloomington, AuthorHouse, 2009.

[48] G. Doucet, Bernard Scott, and Saha Pallab. *Coherency Management: Architecting the Enterprise for Alignment, Agility and Assurance.* AuthorHouse, 2009. ISBN: 9781438996066. URL: `http://books.google.lu/books?id=\_bXQjcc9lWgC`.

[49] Gary Doucet, ed. *Coherency management: Architecting the enterprise for alignment, agility and assurance.* Blommington, Ind.: Authorhouse, 2009. ISBN: 9781438996073.

[50] H.D. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical logic.* Undergraduate texts in mathematics. Springer-Verlag, 1994. ISBN: 9780387942582. URL: http://books.google.de/books?id=VYLA8m7cqYcC.

[51] H. Ehrig, K. Ehrig, and F. Hermann. "From Model Transformation to Model Integration based on the Algebraic Approach to Triple Graph Grammars". In: *Proc. Workshop on Graph Transformation and Visual Modeling Techniques (GT-VMT'08).* Ed. by C. Ermel, J. de Lara, and R. Heckel. Vol. 10. Budapest, Hungary: EC-EASST, 2008. ISBN: ISSN 1863-2122. URL: http://eceasst.cs.tu-berlin.de/index.php/eceasst/issue/view/19.

[52] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. "Formal Integration of Inheritance with Typed Attributed Graph Transformation for Efficient VL Definition and Model Manipulation". In: *Proc. IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05).* 2005. URL: http://tfs.cs.tu-berlin.de/publikationen/Papers05/EEPT05a.pdf.

[53] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformation.* EATCS Monographs in Theor. Comp. Science. Springer, 2006.

[54] H. Ehrig and C. Ermel. "Semantical Correctness and Completeness of Model Transformations using Graph and Rule Transformation". In: *Proc. International Conference on Graph Transformation (ICGT'08).* Vol. 5214. LNCS. Heidelberg: Springer, 2008, pp. 194–210. URL: http://tfs.cs.tu-berlin.de/publikationen/Papers08/EE08a.pdf.

[55] H. Ehrig, C. Ermel, and F. Hermann. "Transformation of Type Graphs with Inheritance for Ensuring Security in E-Government Networks". In: *Proc. International Conference on Fundamental Aspects of Software Engineering (FASE'09).* Ed. by M. Wirsing and M. Chechik. Vol. 5503. Lecture Notes in Computer Science. York, UK: Springer, 2009, pp. 325–339. ISBN: 978-3-642-00592-3. URL: http://tfs.cs.tu-berlin.de/publikationen/Papers09/EEH09.pdf.

[56] H. Ehrig, C. Ermel, F. Hermann, and U. Prange. "On-the-Fly Construction, Correctness and Completeness of Model Transformations based on Triple Graph Grammars". In: *ACM/IEEE 12th International Conference on Model Driven Engineering Languages and Systems (MODELS'09).* Ed. by A. Schürr and B. Selic. Vol. 5795. lncs. Springer, 2009, pp. 241–255.

[57] Hartmut Ehrig, Karsten Ehrig, Claudia Ermel, Frank Hermann, and Gabriele Taentzer. "Information Preserving Bidirectional Model Transformations". In: *Fundamental Approaches to Software Engineering.* Ed. by Matthew B. Dwyer and Antónia Lopes. Vol. 4422. LNCS. Springer, 2007, pp. 72–86. ISBN: 978-3-540-71288-6.

[58] Hartmut Ehrig, Karsten Ehrig, Claudia Ermel, and Ulrike Prange. "Consistent Integration of Models based on Views of Meta Models". In: *Formal Aspects of Computing* 22 (3) (2010), pp. 327–345. DOI: 10.1007/s00165-009-0127-6.

[59] Hartmut Ehrig, Frank Hermann, Hanna Schölzel, and Christoph Brandt. "Propagation of Constraints along Model Transformations Based on Triple Graph Grammars". In: *ECEASST* 41 (2011), pp. 1–14.

[60] Mathias Ekstedt, Ulrik Franke, Pontus Johnson, Robert Lagerström, Teodor Sommestad, Johan Ullberg, and Markus Buschle. "A Tool for Enterprise Architecture Analysis of Maintainability". In: *CSMR*. Ed. by Andreas Winter, Rudolf Ferenc, and Jens Knodel. IEEE, 2009, pp. 327–328.

[61] Mathias Ekstedt, Pontus Johnson, Åsa Lindström, Magnus Gammelgård, Erik Johansson, Leonel Plazaola, Enrique Silva, and Joakim Liliesköld. "Consistent Enterprise Software System Architecture for the CIO – A Utility-Cost Based Approach". In: *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 8 - Volume 8*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 80225.1–. ISBN: 0-7695-2056-1. URL: `http://dl.acm.org/citation.cfm?id=962756.963270`.

[62] Brigitte Eller. "Anwendungsentwicklung vom Standpunkt der sprachbasierten Informatik". In: *Usability Engineering in der Anwendungsentwicklung*. Gabler, 2009, pp. 99–178. ISBN: 978-3-8349-8459-3.

[63] D.W. Embley and B. Thalheim. *Handbook of Conceptual Modeling: Theory, Practice, and Research Challenges*. Springer, 2011. ISBN: 9783642158643. URL: `http://books.google.de/books?id=tSqvcQAACAAJ`.

[64] Claudia Ermel. "Visual Modelling and Analysis of Model Transformations based on Graph Transformation". In: *Bulletin of the EATCS* 99 (2009), pp. 135 –152.

[65] Claudia Ermel, Karsten Ehrig, Gabriele Taentzer, and Eduard Weiss. "Object Oriented and Rule-based Design of Visual Languages using Tiger". In: *ECEASST* 1 (2006).

[66] O.K. Ferstl and E.J. Sinz. *Modeling of business systems using the Semantic Object Model (SOM): a methodological framework*. Bamberger Beiträge zur Wirtschaftsinformatik. Otto-Friedrich-Univ., 1997. URL: `http://books.google.lu/books?id=I5nYSgAACAAJ`.

[67] Otto K. Ferstl and Elmar J. Sinz. "Der Ansatz des Semantischen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen". In: *Wirtschaftsinformatik* 37.3 (1995), pp. 209–220.

[68] Otto K. Ferstl, Elmar J. Sinz, Michael Amberg, Udo Hagemann, and Carsten Malischewski. "Tool-Based Business Process Modeling Using the SOM Approach". In: *24. GI Jahrestagung*. Ed. by Bernd E. Wolfinger. Springer, 1994, pp. 430–436.

[69] R. Fischer. *Organisation der Unternehmensarchitektur: Entwicklung der aufbau- und ablauforganisatorischen Strukturen unter besonderer Berücksichtigung des Gestaltungsziels Konsistenzerhaltung*. PhD Thesis. University of St. Gallen. 2008. URL: `http://books.google.lu/books?id=mydqPgAACAAJ`.

[70] M. Fontoura, W. Pree, and B. Rumpe. *The UML profile for framework architectures.* Addison-Wesley object technology series. Addison-Wesley, 2002. ISBN: 9780201675184. URL: http://books.google.de/books?id=991QAAAAMAAJ.

[71] Mark S. Fox and Michael Grüninger. "Enterprise Modeling". In: *AI Magazine* 19.3 (1998), pp. 109–121.

[72] U. Frank. "MEMO: A Tool Supported Methodology for Analyzing and (Re-)Designing Business Information Systems". In: *Technology of Object-Oriented Languages ans Systems* (1994). Ed. by R. Ege, M. Singh, and B. Meyer, pp. 367–380.

[73] Ulrich Frank. "The MEMO Meta Modelling Language (MML) and Language Architecture". In: *Online* 24.24 (2010). http://www.icb.uni-due.de/fileadmin/ICB/research/research_reports/ICB-Report_No24.pdf, p. 48.

[74] Ulrich Frank. *The MEMO Object Modelling Language (MEMO-OML).* Arbeitsberichte des Instituts für Wirtschaftsinformatik. Universität Koblenz 10. http://www.wi-inf.uni-due.de/FGFrank/documents/Arbeitsberichte_Koblenz/Nr10.pdf. 1998.

[75] P.A. Fritzson. *Principles of object-oriented modeling and simulation with Modelica 2.1.* IEEE Press, 2004. ISBN: 9780471471639. URL: http://books.google.de/books?id=IzqY8Abz1rAC.

[76] Magnus Gammelgård, Åsa Lindström, and Mårten Simonsson. "A reference model for IT management responsibilities". In: *EDOC Workshops.* IEEE Computer Society, 2006, p. 26.

[77] J.O. Grady. *System requirements analysis.* Elsevier Academic Press, 2006. ISBN: 9780120885145. URL: http://books.google.de/books?id=FkpqAnHUNLYC.

[78] Jan Friso Groote, Aad Mathijssen, Michel Reniers, Yaroslav S. Usenko, and Muck van Weerdenburg. "The Formal Specification Language mCRL2." In: *MMOSS.* Ed. by Ed Brinksma, David Harel, Angelika Mader, Perdita Stevens, and Roel Wieringa. Vol. 06351. Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2006. URL: http://drops.dagstuhl.de/opus/volltexte/2007/862.

[79] Martin Hafner and Robert Winter. "Processes for Enterprise Application Architecture Management". In: *HICSS.* IEEE Computer Society, 2008, p. 396.

[80] Inge Hanschke. *Strategic IT Management: A Toolkit for Enterprise Architecture Management.* 1st Edition. Springer, Dec. 2009. ISBN: 3642050336. URL: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/3642050336.

[81] R. Heckel, J. Küster, and G. Taentzer. "Confluence of Typed Attributed Graph Transformation with Constraints". In: *Proc. of 1st Int. Conference on Graph Transformation.* Ed. by A. Corradini, H. Ehrig, H.-J. Kreowski, and Rozenberg. G. Vol. 2505. LNCS. Springer, 2002. URL: http://tfs.cs.tu-berlin.de/%7Egabi/gHKT02b.pdf.

[82] F. Hermann. *Analysis and Optimization of Visual Enterprise Models - Based on Graph and Model Transformation.* PhD thesis, online: http://opus.kobv.de/tuberlin/volltexte/2011/3008/. Universitätsverlag der TU Berlin, 2011. ISBN: 978-3-7983-2321-6.

[83] Frank Hermann, Hartmut Ehrig, Ulrike Golas, and Fernando Orejas. "Efficient Analysis and Execution of Correct and Complete Model Transformations Based on Triple Graph Grammars". In: *Proc. Int. Workshop on Model Driven Interoperability (MDI'10).* Ed. by J. Bézivin, R.M. Soley, and A. Vallecillo. MDI '10. Oslo, Norway: ACM, 2010, pp. 22–31. ISBN: 978-1-4503-0292-0. DOI: http://doi.acm.org/10.1145/1866272.1866277. URL: http://tfs.cs.tu-berlin.de/publikationen/Papers10/HEGO10.pdf.

[84] Frank Hermann, Hartmut Ehrig, Fernando Orejas, Krzysztof Czarnecki, Zinovy Diskin, and Yingfei Xiong. "Correctness of Model Synchronization Based on Triple Graph Grammars". In: *ACM/IEEE 14th International Conference on Model Driven Engineering Languages and Systems (MODELS'11).* Vol. 6981. LNCS. Springer, 2011, pp. 668–682. ISBN: 978-3-642-24484-1.

[85] Frank Hermann, Hartmut Ehrig, Fernando Orejas, and Ulrike Golas. "Formal Analysis of Functional Behaviour of Model Transformations Based on Triple Graph Grammars". In: *Proceedings of Intern. Conf. on Graph Transformation ( ICGT' 10).* Ed. by H. Ehrig, A. Rensink, G. Rozenberg, and A. Schürr. Vol. 6372. LNCS. Springer, 2010, pp. 155–170. ISBN: ISBN 978-3-642-15927-5. URL: http://tfs.cs.tu-berlin.de/publikationen/Papers10/HEOG10.pdf.

[86] Frank Hermann, Mathias Hülsbusch, and Barbara König. "Specification and Verification of Model Transformations". In: *ECEASST* 30 (2010). Ed. by C. Ermel, H. Ehrig, F. Orejas, and G. Taentzer, pp. 1–21. URL: http://journal.ub.tu-berlin.de/index.php/eceasst/issue/archive.

[87] IEEE. *Recommended Practice for Architectural Description of Software Intensive Systems.* Tech. rep. IEEE P1471:2000, ISO/IEC 42010:2007. The Architecture Working Group of the Software Engineering Committee, Standards Department, IEEE, 2000.

[88] IFIP-IFAC Task Force. *GERAM: Generalised Enterprise Reference Architecture and Methodology – Version 1.6.3.* Tech. rep. Annex to ISO WD15704. 1999, pp. 1–31.

[89] ISO. *Enterprise integration – Framework for enterprise modelling.* ISO 19439:2006(E). 2006.

[90] P. Johnson and M. Ekstedt. *Enterprise Architecture: Models and Analyses for Information Systems Decision Making*. Studentlitteratur, 2007. ISBN: 9789144027524. URL: `http://books.google.lu/books?id=2LdxPQAACAAJ`.

[91] P. Johnson, M. Ekstedt, E. Silva, and L. Plazaola. *Using enterprise architecture for CIO decision-making: On the importance of theory*. Tech. rep. KTH, Royal Institute of Technology, Stockholm - Sweden, 2004. URL: `{http://www.ee.kth.se/php/modules/publications/reports/2004/IR-EE-ICS\_2004\_001.pdf}`.

[92] Pontus Johnson, Lars Nordström, and Robert Lagerström. "Formalizing Analysis of Enterprise Architecture". In: *Enterprise Interoperability*. Ed. by Guy Doumeingts, Jörg Müller, Gérard Morel, and Bruno Vallespir. Springer London, 2007, pp. 35–44. ISBN: 978-1-84628-714-5.

[93] H. Jonkers, M. Lankhorst, H. ter Doest, F. Arbab, H. Bosma, and R. Wieringa. "Enterprise architecture: Management tool and blueprint for the organisation". In: *Information Systems Frontiers* 8.2 (2006). Springer, pp. 63–66.

[94] Henk Jonkers, René van Buuren, Farhad Arbab, Frank de Boer, Marcello Bonsangue, Hans Bosma, Hugo ter Doest, Luuk Groenewegen, Juan Guillen Scholten, Stijn Hoppenbrouwers, Maria-Eugenia Iacob, Wil Janssen, Marc Lankhorst, Diederik van Leeuwen, Erik Proper, Andries Stam, Leon van der Torre, and Gert Veldhuijzen van Zanten. "Towards a Language for Coherent Enterprise Architecture Descriptions". In: *Proceedings of the 7th International Conference on Enterprise Distributed Object Computing*. EDOC '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 28–. ISBN: 0-7695-1994-6. URL: `http://dl.acm.org/citation.cfm?id=942793.943148`.

[95] Henk Jonkers, Marc Lankhorst, RenÃľ Van Buuren, Marcello Bonsangue, and Leendert Van Der Torre. "Concepts for Modeling Enterprise Architectures". In: *International Journal of Cooperative Information Systems* 13 (2004), pp. 257–287.

[96] Henk Jonkers, Marc M. Lankhorst, René van Buuren, Stijn Hoppenbrouwers, Marcello M. Bonsangue, and Leendert W. N. van der Torre. "Concepts For Modeling Enterprise Architectures". In: *Int. J. Cooperative Inf. Syst.* 13.3 (2004), pp. 257–287. URL: `http://icr.uni.lu/leonvandertorre/papers/ijcis04.pdf`.

[97] J. Jung. "Entwurf einer Sprache für die Modellierung von Ressourcen im Kontext der Geschäftsprozessmodellierung". Dissertation. Universitaet Duisburg Essen, 2007.

[98] Adrian Kallgren, Johan Ullberg, and Pontus Johnson. "A Method for Constructing a Company Specific Enterprise Architecture Model Framework". In: *SNPD*. Ed. by Haeng-Kon Kim and Roger Y. Lee. IEEE Computer Society, 2009, pp. 346–351. ISBN: 978-0-7695-3642-2.

[99] M. Kifer, G. Lausen, and J. Wu. "Logical Foundations of Object-Oriented and Frame-Based Languages". In: *Journal of the ACM* 42 (1995), pp. 741–843.

[100] E. Kindler and R. Wagner. *Triple Graph Grammars: Concepts, Extensions, Implementations, and Application Scenarios.* Tech. rep. TR-ri-07-284. Universität Paderborn, 2007.

[101] L. Kirchner. *Eine Methode zur Unterstützung des IT-Managements im Rahmen der Unternehmensmodellierung.* Logos Verlag Berlin, 2008. ISBN: 9783832518202. URL: `http://books.google.lu/books?id=uJpnPgAACAAJ`.

[102] Natallia Kokash, Christian Krause, and Erik de Vink. " Reo+mCRL2: A Framework for Model-checking Dataflow in Service Compositions ". In: *Formal Aspects of Computing* (2011), pp. 1–30. ISSN: 0934-5043.

[103] S. Kurpjuweit. *Stakeholder-orientierte Modellierung und Analyse der Unternehmensarchitektur unter besonderer Berücksichtigung der Geschäfts- und IT-Architektur.* Logos Verlag Berlin, 2009. ISBN: 9783832522520. URL: `http://books.google.lu/books?id=JJOnzo43-OYC`.

[104] Stephan Kurpjuweit and Robert Winter. "Concern-oriented business architecture engineering". In: *Proceedings of the 2009 ACM symposium on Applied Computing.* SAC '09. Honolulu, Hawaii: ACM, 2009, pp. 265–272. ISBN: 978-1-60558-166-8. DOI: `http://doi.acm.org/10.1145/1529282.1529339`. URL: `http://doi.acm.org/10.1145/1529282.1529339`.

[105] Stephan Kurpjuweit and Robert Winter. "Viewpoint-based Meta Model Engineering". In: *EMISA.* Ed. by Manfred Reichert, Stefan Strecker, and Klaus Turowski. Vol. P-119. LNI. GI, 2007, pp. 143–161. ISBN: 978-3-88579-213-0.

[106] Robert Lagerström, Moustafa Chenine, Pontus Johnson, and Ulrik Franke. "Probabilistic Metamodel Merging". In: *CAiSE Forum.* Ed. by Zohra Bellahsene, Carson Woo, Ela Hunt, Xavier Franch, and Remi Coletta. Vol. 344. CEUR Workshop Proceedings. CEUR-WS.org, 2008, pp. 25–28. URL: `http://ceur-ws.org/Vol-344/paper7.pdf`.

[107] Robert Lagerstrom and Pontus Johnson. "Using Architectural Models to Predict the Maintainability of Enterprise Systems". In: *Proceedings of the 2008 12th European Conference on Software Maintenance and Reengineering.* Washington, DC, USA: IEEE Computer Society, 2008, pp. 248–252. ISBN: 978-1-4244-2157-2. DOI: `10.1109/CSMR.2008.4493320`. URL: `http://dl.acm.org/citation.cfm?id=1545010.1545369`.

[108] M. Lankhorst. *Enterprise architecture at Work: Modelling, Communication, and Analysis.* Springer, 2005. ISBN: 9783540243717. URL: `http://books.google.lu/books?id=ZR\_1gXt9vcYC`.

[109] M. M. Lankhorst. "Enterprise architecture modelling—the issue of integration". In: *Advanced Engineering Informatics* 18.4 (2004). Elsevier, pp. 205–216.

[110] Marc Lankhorst. *Enterprise Architecture at Work: Modelling, Communication and Analysis.* 2. Berlin: Springer, 2009. ISBN: 978-3-642-01309-6. DOI: `http://dx.doi.org/10.1007/978-3-642-01310-2`.

[111] Marc M. Lankhorst, Henderik Alex Proper, and Henk Jonkers. “The Anatomy of the ArchiMate Language”. In: *IJISMD* 1.1 (2010), pp. 1–32.

[112] Lam-Son Le and Alain Wegmann. “Definition of an Object-Oriented Modeling Language for Enterprise Architecture”. In: *Proceedings of the Proceedings of the 38th Annual Hawaii International Conference on System Sciences - Volume 08*. HICSS ’05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 222.1–. ISBN: 0-7695-2268-8-8. DOI: `http://dx.doi.org/10.1109/HICSS.2005.186`. URL: `http://dx.doi.org/10.1109/HICSS.2005.186`.

[113] Lam-Son Lê and Alain Wegmann. “SeamCAD: Object-Oriented Modeling Tool for Hierarchical Systems in Enterprise Architecture”. In: *HICSS*. IEEE Computer Society, 2006. ISBN: 0-7695-2507-5.

[114] Wan Li and Shengfeng Tian. “XSWRL, an Extended Semantic Web Rule Language and prototype implementation”. In: *Expert Syst. Appl.* 38 (3 2011), pp. 2040–2045. ISSN: 0957-4174. DOI: `http://dx.doi.org/10.1016/j.eswa.2010.07.141`. URL: `http://dx.doi.org/10.1016/j.eswa.2010.07.141`.

[115] João Marques, Marielba Zacarias, and José Tribolet. “A Bottom-Up Competency Modeling Approach”. In: *Advances in Enterprise Engineering IV*. Ed. by Antonia Albani, Jan L. G. Dietz, Wil Aalst, John Mylopoulos, Michael Rosemann, Michael J. Shaw, and Clemens Szyperski. Vol. 49. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, 2010, pp. 50–64. ISBN: 978-3-642-13048-9.

[116] A.C.N. Martens. “Relevance of conformance analysis information”. MA thesis. Eindhoven University of Technology (TU/e), 2009.

[117] *MOFLON*. `http://www.moflon.org/`, retrived on 06-FEB-2012. Real-Time Systems Lab, TU Darmstadt. 2011.

[118] Emmanuel Mulo, Uwe Zdun, and Schahram Dustdar. “An event view model and DSL for engineering an event-based SOA monitoring infrastructure”. In: *Proceedings of the Fourth ACM International Conference on Distributed Event-Based Systems*. DEBS ’10. Cambridge, United Kingdom: ACM, 2010, pp. 62–72. ISBN: 978-1-60558-927-5. DOI: `http://doi.acm.org/10.1145/1827418.1827428`. URL: `http://doi.acm.org/10.1145/1827418.1827428`.

[119] Per Närman, Marten Schönherr, Pontus Johnson, Mathias Ekstedt, and Moustafa Chenine. “Using Enterprise Architecture Models for System Quality Analysis”. In: *EDOC*. IEEE Computer Society, 2008, pp. 14–23. ISBN: 978-0-7695-3373-5.

[120] Tien N. Nguyen and Ethan V. Munson. “A Formalism for Conformance Analysis and Its Applications”. In: *Software Engineering and Formal Methods, IEEE International Conference on* 0 (2004), pp. 330–339. DOI: `http://doi.ieeecomputersociety.org/10.1109/SEFM.2004.10019`.

[121] Tien N. Nguyen and Ethan V. Munson. “A Model for Conformance Analysis of Software Documents”. In: *Proceedings of the 6th International Workshop on Principles of Software Evolution*. IEEE Computer Society, 2003, pp. 24–. ISBN: 0-7695-1903-2.

[122] K.D. Niemann. *From enterprise architecture to IT governance: elements of effective IT management.* Edition CIO. Vieweg, 2006. ISBN: 9783834801982. URL: http://books.google.lu/books?id=1NfMdTwqFcUC.

[123] O. Noran. "A mapping of individual architecture frameworks (GRAI, PERA, C4ISR, CIMOSA, ZACHMAN, ARIS) onto GERAM". In: *Handbook on enterprise architecture.* Springer, 2003, pp. 65–212.

[124] Object Management Group. *Meta Object Facility (MOF) Core Specification Version 2.0.* http://www.omg.org/technology/documents/modeling_spec_catalog.htm#MOF. Object Management Group (OMG), 2008.

[125] B. Oestereich. *Developing software with UML: object-oriented analysis and design in practice.* The Addison-Wesley object technology series. Addison-Wesley, 2002. ISBN: 9780201756036. URL: http://books.google.de/books?id=WThOswOGv7kC.

[126] Erich Ortner. "From Software Engineering to Enterprise Engineering - Introduction to a Language-critical Approach". In: *Innovative Techniques in Instruction Technology, E-learning, E-assessment, and Education.* Ed. by Magued Iskander. Springer, 2008, pp. 135–143. ISBN: 978-1-4020-8738-7.

[127] H. Österle and R. Winter. *Business Engineering: auf dem Weg zum Unternehmen des Informationszeitalters.* Business Engineering. Springer, 2003. ISBN: 9783540000495. URL: http://books.google.lu/books?id=g9xZXceC9IsC.

[128] H. Österle, R. Winter, F. Höning, S. Kurpjuweit, and P. Osl. "Business Engineering: Core-Business-Metamodell". In: *WISU – Das Wirtschaftsstudium.* 2007, pp. 191–194.

[129] Judith Pavón, Sidney Viana, and Grassiani Lino de Campos. "A rule meta-model for business rules". In: *The 16th IASTED International Conference on Applied Simulation and Modelling.* Palma de Mallorca, Spain: ACTA Press, 2007, pp. 277–282. ISBN: 978-0-88986-688-1. URL: http://dl.acm.org/citation.cfm?id=1659042.1659095.

[130] H.A. Proper, S.J.B.A. Hoppenbrouwers, and G.E. Veldhuijzen van Zanten. "Communication of Enterprise Architectures". In: *Enterprise Architecture at Work: Modelling, Communication and Analysis* (2005), pp. 67–82.

[131] Jakob Raderius, Per Närman, and Mathias Ekstedt. "Service-Oriented Computing — ICSOC 2008 Workshops". In: ed. by George Feuerlicht and Winfried Lamersdorf. Berlin, Heidelberg: Springer-Verlag, 2009. Chap. Assessing System Availability Using an Enterprise Architecture Analysis Approach, pp. 351–362. ISBN: 978-3-642-01246-4. DOI: http://dx.doi.org/10.1007/978-3-642-01247-1_36. URL: http://dx.doi.org/10.1007/978-3-642-01247-1_36.

[132] Boyce Raynard. *TOGAF The Open Group Architecture Framework 100 Success Secrets - 100 Most Asked Questions: The Missing TOGAF Guide on How to achieve and then sustain superior Enterprise Architecture execution.* Emereo Pty Ltd, 2008. ISBN: 1921523131, 9781921523137.

[133] Gil Regev, Olivier Hayard, Donald C. Gause, and Alain Wegmann. "Modeling Service-Level Requirements: A Constancy Perspective". In: *RE.* IEEE Computer Society, 2009, pp. 231–236. ISBN: 978-0-7695-3761-0.

[134] Christian Riege and Stephan Aier. "Service-Oriented Computing — ICSOC 2008 Workshops". In: ed. by George Feuerlicht and Winfried Lamersdorf. Berlin, Heidelberg: Springer-Verlag, 2009. Chap. A Contingency Approach to Enterprise Architecture Method Engineering, pp. 388–399. ISBN: 978-3-642-01246-4. DOI: `http://dx.doi.org/10.1007/978-3-642-01247-1\_39`. URL: `http://dx.doi.org/10.1007/978-3-642-01247-1\_39`.

[135] A. Rozinat and W.M.P. van der Aalst. "Conformance checking of processes based on monitoring real behavior". In: *Information Systems* 33.1 (2008), pp. 64 –95. ISSN: 0306-4379. DOI: `DOI:10.1016/j.is.2007.07.001`.

[136] A. Rozinat and W.M.P. van der Aalst. "Conformance Testing: Measuring the Alignment Between Event Logs and Process Models". In: ed. by S. Sadiq, M. Indulska, M. zur Muehlen, E. Dubois, and P. Johannesson. Vol. WP 144. BETA Working Paper Series. Eindhoven University of Technology, 2005.

[137] A. Rozinat, I. S. M. De Jong, C. W. Günther, and W.M.P. van der Aalst. "Conformance Analysis of ASML's Test Process". In: *Proceedings of the Second International Workshop on Governance, Risk and Compliance (GRCIS'09).* Ed. by S. Sadiq, M. Indulska, M. zur Muehlen, E. Dubois, and P. Johannesson. Vol. 459. CEUR Workshop Proceedings. Springer-Verlag, 2006, pp. 1–15.

[138] Irina Rychkova and Alain Wegmann. "A Method for Functional Alignment Verification in Hierarchical Enterprise Models". In: *BUSITAL.* Ed. by Yves Pigneur and Carson Woo. Vol. 237. CEUR Workshop Proceedings. CEUR-WS.org, 2006.

[139] Irina Rychkova and Alain Wegmann. "Formal Semantics for Property-Property Relations in SEAM Visual Language: Towards Simulation and Analysis of Visual Specifications". In: *MSVVEIS.* Ed. by Juan Carlos Augusto, Joseph Barjis, and Ulrich Ultes-Nitsche. INSTICC PRESS, 2007, pp. 138–147. ISBN: 978-972-8865-95-5.

[140] August-Wilhelm Scheer. *Architektur integrierter Informationssysteme. Grundlagen der Unternehmensmodellierung.* 2nd. Springer, 1992. XIII, 210. ISBN: 3540554017. URL: `http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+033279780&sourceid=fbw_bibsonomy`.

[141] A.W. Scheer. *ARIS - Modellierungsmethoden, Metamodelle, Anwendungen.* Springer, 2001. ISBN: 9783540416012. URL: `http://books.google.lu/books?id=ipdUUCkfCaEC`.

[142] A.W. Scheer. *ARIS - vom Geschäftsprozess zum Anwendungssystem.* Springer, 1999. ISBN: 9783540658238. URL: http://books.google.lu/books?id=Oa0qXJqZxRkC.

[143] Joachim Schelp and Matthias Stutz. "A Balanced Scorecard Approach to Measure the Value of Enterprise Architecture". In: *Architecture* 3.1 (2007). Ed. by Marc M Lankhorst and PontusEditors Johnson, pp. 5–12. URL: http://www.via-nova-architectura.org/files/TEAR2007/Schelp.pdf.

[144] Bruno Schienmann. *Objektorientierter Fachentwurf: ein terminologiebasierter Ansatz für die Konstruktion von Anwendungssystemen.* Teubner-Texte zur Informatik, 1997. ISBN: 3-8154-2305-8.

[145] A. Schürr. "Specification of Graph Translators with Triple Graph Grammars". In: *Proc. Int. Workshop on Graph-Theoretic Concepts in Computer Science (WG'94).* Ed. by G. Tinhofer. Vol. 903. LNCS. Springer, 1994, pp. 151–163.

[146] Andy Schürr and Felix Klar. "15 Years of Triple Graph Grammars". In: *Proceedings of the 4th international conference on Graph Transformations.* Ed. by Hartmut Ehrig, Reiko Heckel, Grzegorz Rozenberg, and Gabriele Taentzer. Vol. 5214. LNCS. Springer, 2008, pp. 411–425. ISBN: 978-3-540-87404-1. DOI: http://dx.doi.org/10.1007/978-3-540-87405-8_28. URL: http://dx.doi.org/10.1007/978-3-540-87405-8_28.

[147] Christian M. Schweda. "Development of Organization-Specific Enterprise Architecture Modeling Languages Using Building Blocks". http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20110728-1072071-1-0. Dissertation. Technische Universität München, 2011.

[148] J. F. Sowa and J. A. Zachman. "Extending and formalizing the framework for information systems architecture". In: *IBM Syst. J.* 31 (3 1992), pp. 590–616. ISSN: 0018-8670. DOI: http://dx.doi.org/10.1147/sj.313.0590. URL: http://dx.doi.org/10.1147/sj.313.0590.

[149] Steffen Staab and Rudi Studer, eds. *Handbook on Ontologies.* International Handbooks on Information Systems. Springer, 2004. ISBN: 3-540-40834-7.

[150] Thomas Stahl and Markus Völter. *Model-Driven Software Development: Technology, Engineering, Management.* Wiley, 2006. ISBN: 978-0-470-02570-3.

[151] Gernot Starke and Stefan Tilkov, eds. *SOA-Expertenwissen: Methoden, Konzepte und Praxis serviceorientierter Architekturen.* dpunkt, 2007. ISBN: 978-3-89864-437-2.

[152] Rudi Studer, V. Richard Benjamins, and Dieter Fensel. "Knowledge engineering: principles and methods". In: *Data Knowl. Eng.* 25 (1-2 1998), pp. 161–197. ISSN: 0169-023X. DOI: 10.1016/S0169-023X(97)00056-6. URL: http://dl.acm.org/citation.cfm?id=290547.290553.

[153] The Open Group. *TOGAF 9 - The Open Group Architecture Framework Version 9*. The Open Group, 2009, p. 744.

[154] *Unified Modeling Language: Superstructure – Version 2.3*. http://www.omg.org/technology/documents/modeling_spec_catalog.htm. Object Management Group. 2010.

[155] Pieter Van Gorp, Anne Keller, and Dirk Janssens. "Transformation Language Integration Based on Profiles and Higher Order Transformations". In: *Software Language Engineering*. Ed. by Dragan Gasevic, Ralf Lämmel, and Eric Van Wyk. Vol. 5452. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2009, pp. 208–226. ISBN: 978-3-642-00433-9.

[156] André Vasconcelos, Artur Caetano, Joao Neves, Pedro Sinogas, Ricardo Mendes, and José Tribolet. "A Framework for Modeling Strategy, Business Processes and Information Systems". In: *Proc. Fifth International Enterprise Distributed Object Computing Conference (EDOC 2001*. IEEE Computer Society, 2001, pp. 69–81.

[157] André Vasconcelos, Carla Marques Pereira, Pedro Manuel Antunes Sousa, and José M. Tribolet. "Open Issues on Information System Architecture Research Domain: The Vision". In: *Proc. of the 6th International Conference on Enterprise Information Systems (ICEIS), Volume 3*. 2004, pp. 273–282.

[158] André Vasconcelos, Pedro Sousa, and José Tribolet. "Information System Architecture Evaluation: From Software to Enterprise Level Approaches". In: *Proc. 12th European Conference On Information Technology Evaluation (ECITE 2005)*. 2005. URL: http://en.scientificcommons.org/42235328.

[159] André Vasconcelos, Pedro Manuel Antunes Sousa, and José M. Tribolet. "Enterprise Architecture Analysis - An Information System Evaluation Approach." In: *Enterprise Modelling and Information Systems Architectures* 3.2 (2008), pp. 31–53. URL: http://dblp.uni-trier.de/db/journals/emisaij/emisaij3.html#VasconcelosST08.

[160] J.B. Filipe, B. Aharp, and P. Miranda, eds. *On the Systemic Enterprise Architecture Methodology (SEAM)*. Vol. 3. Angers, France: Kluwer, 2003, pp. 483–490. ISBN: 978-1-4020-0563-3.

[161] Alain Wegmann, Anders Kotsalainen, Lionel Matthey, Gil Regev, and Alain Giannattasio. "Augmenting the Zachman Enterprise Architecture Framework with a Systemic Conceptualization". In: *EDOC*. IEEE Computer Society, 2008, pp. 3–13. ISBN: 978-0-7695-3373-5.

[162] Alain Wegmann, Lam-Son Lê, Gil Regev, and Bryan Wood. "Enterprise modeling using the foundation concepts of the RM-ODP ISO/ITU standard". In: *Inf. Syst. E-Business Management* 5.4 (2007), pp. 397–413.

[163] Alain Wegmann, Gil Regev, Irina Rychkova, Lam-Son Lê, and Philippe Julia. "Business and IT Alignment with SEAM for Enterprise Architecture". In: *EDOC*. IEEE Computer Society, 2007, pp. 111–121.

[164] Robert Winter and Ronny Fischer. “Essential Layers, Artifacts, and Dependencies of Enterprise Architecture”. In: *Proceedings of the 10th IEEE on International Enterprise Distributed Object Computing Conference Workshops*. EDOCW ’06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 30–. ISBN: 0-7695-2743-4. DOI: http://dx.doi.org/10.1109/EDOCW.2006.33. URL: http://dx.doi.org/10.1109/EDOCW.2006.33.

[165] Robert Winter and Joachim Schelp. “Enterprise architecture governance: the need for a business-to-IT approach”. In: *SAC*. Ed. by Roger L. Wainwright and Hisham Haddad. ACM, 2008, pp. 548–552. ISBN: 978-1-59593-753-7.

[166] Robert Winter and Joachim Schelp. “Enterprise architecture governance: the need for a business-to-IT approach”. In: *Proceedings of the 2008 ACM symposium on Applied computing*. SAC ’08. Fortaleza, Ceara, Brazil: ACM, 2008, pp. 548–552. ISBN: 978-1-59593-753-7. DOI: http://doi.acm.org/10.1145/1363686.1363820. URL: http://doi.acm.org/10.1145/1363686.1363820.

[167] M. Zacarias, H. S. Pinto, R. Magalhães, and J. Tribolet. “A ’context-aware’ and agent-centric perspective for the alignment between individuals and organizations”. In: *Inf. Syst.* 35 (4 2010), pp. 441–466. ISSN: 0306-4379. DOI: http://dx.doi.org/10.1016/j.is.2009.03.014. URL: http://dx.doi.org/10.1016/j.is.2009.03.014.

[168] John A. Zachman. “A framework for information systems architecture”. In: *IBM Syst. J.* 26 (3 1987), pp. 276–292. ISSN: 0018-8670. DOI: http://dx.doi.org/10.1147/sj.263.0276. URL: http://dx.doi.org/10.1147/sj.263.0276.