AIS Transactions on Human-Computer Interaction

Volume 9 | Issue 1 Article 3

3-30-2017

Conceptual Modeling in Human Resource Management: A Design Research Approach

Stefan Strohmeier
Universität des Saarlandes, s.strohmeier@mis.uni-saarland.de

Friedrich Ro?hrs
Saarland University, f.roehrs@mis.uni-saarland.de

Transactions on Human – Computer Interaction

Research Paper

Conceptual Modeling in Human Resource Management: A Design Research Approach

Stefan Strohmeier
Saarland University, Germany
s.strohmeier@mis.uni-saarland.de

Friedrich Röhrs
Saarland University, Germany

Abstract:

In this paper, we introduce conceptual human resource modeling (CHRM) as a new methodical paradigm in HRM. To this end, we employ a design research approach. On a general level, we identify general tasks that CHRM can tackle and general solutions that it can provide. On a concrete level, we use the specific problem of employee assignment as an example and develop a specific CHRM solution for this HR task. By using the prototypical CHRM tool, we show how one can solve practical HR tasks based on CHRM. Finally, we discuss the lessons learned and implications for future research in CHRM.

Keywords: Modeling, Modeling Language, Support, Design Science, Organization, Human Resource Management.

The manuscript was received 12/14/2015 and was with the authors 6 months for 2 revisions.

1 Introduction

Conceptual modeling refers to building a formal representation of selected phenomena of a specific real-world domain to better analyze and design that domain (Mylopoulos, 1992; Thalheim, 2010; Wand & Weber, 2002). To this end, conceptual modeling uses predefined concepts and predefined rules to build domain representations. These representations are expressed by graphical symbols that are combined in comprehensive graphical models. The entirety of the concepts, rules, and symbols that are employed for a specific modeling purpose constitutes a modeling language that is usually implemented in a corresponding modeling tool (e.g., Frank, 2013; Gemino & Wand, 2003).

The basic justification for using conceptual modeling is that it allows users to considerably more effectively analyze and design real world domains. Compared with other analysis and design approaches, conceptual modeling's major advantage lies in its superior *cognitive effectiveness* (i.e., the speed, ease, and accuracy with which a human mind can process conceptual models) (e.g., Cheng, Lowe, & Scaife, 2001; Larkin & Simon, 1987; Moody, 2009). Thus, using proper diagrammatical representations of domains enhances the human capability of identifying and solving domain problems and tasks in a basic way.

Practitioners first applied conceptual modeling in the software engineering domain. In software engineering, one uses conceptual modeling to analyze and design the content, structures, and functionalities of envisioned information systems (e.g., Mylopoulos, 1992; Thalheim, 2010; Wand & Weber, 2002). For some time, conceptual modeling has become a prominent methodology in practical software engineering. For example, Unified Modeling Language *(*UML) constitutes a prominent language for graphically analyzing and designing envisioned information systems. UML tools that support practical software engineering adopt UML (e.g., Dennis, Wixom, & Tegarden, 2015; Eichelberger, Schmid, & Eldogan, 2011; Rumbaugh, Jacobson, & Booch, 2010).

By now, researchers understand conceptual modeling not only as a method of software engineering but also as an overarching methodology of analyzing and designing any complex real-world domain (e.g., Anaby-Tavor et al., 2010; Frank, 2013; Recker, 2015). To realize the potentials of conceptual modeling in additional domains, researchers have suggested developing domain-specific modeling languages and tools that explicitly consider the peculiar characteristics and requirements of the respective domain (e.g., Frank, 2013; Gray et al., 2007; Tolvanen & Kelly, 2010). Accordingly, individuals in domains other than software engineering have increasingly begun to use it (e.g., Frank et al., 2014; Recker, 2015; Roussopoulos & Karagiannis, 2009). Prominent managerial application domains include, for instance, the conceptual modeling of business strategies (e.g., Giannoulis et al., 2013), business plans (e.g., Francesconi, Dalpiaz, & Mylupoulos, 2013) and, in particular, business processes (e.g., van der Aalst, 2013). In these domains, researchers have developed and used domain-specific modeling languages and tools to analyze and design business strategies, business plans, or business processes based on graphical models. Moreover, in these domains, conceptual modeling shows considerable success. In business process management, for instance, conceptual modeling has been the clearly dominant methodical paradigm for more than two decades (e.g., Scheer, Thomas, & Adam, 2005; van der Aalst, 2013; Weske, 2012).

However, in the domain of human resource management (HRM), conceptual modeling has played only a minor role if any to date. Of course, practitioners in this domain have also employed the languages and tools of conceptual software modeling to analyze and design HR software (e.g., Ibrahim, Ahmed, Bahrudin, Moftah, & Algarai, 2014; Meegana & Ranatunga, 2009) and the languages and tools of conceptual business process modeling to analyze and design HR processes (e.g., Cakar and Bititci 2002; Cakar, Bititci, & MacBryde, 2003). However, domain-specific conceptual modeling languages and tools that explicitly address the graphical analysis and design of HR (sub)domains, such as the recruitment, development, or compensation of employees, do not yet exist. The domain of HRM seems to be broadly unaware of the possibilities of domain-specific conceptual modeling and does not apply it.

Given that the literature on domain-specific modeling implies that one can, in fact, apply conceptual modeling in the HR domain and that one could see benefits from doing so, one can characterize the current situation as an application gap or, at least, as an overlooked application possibility. As such, elaborating on this application possibility constitutes a pending research task that we need to perform to consider and exploit a prominent and promising methodical paradigm in HRM. The superior success of conceptual modeling in adjacent managerial domains also clearly supports the necessity of researching domain-specific conceptual modeling in HRM. Against this backdrop, we introduce conceptual modeling to

HRM and facilitate the future exploitation of conceptual modeling in HRM. To this end, we employ a design research approach to uncover how one can employ conceptual modeling in HRM. Design research generally aims to provide a class of solutions to a class of tasks that, to date, either show no solution or a solution that needs improvement (e.g., Hevner, March, Park, & Ram, 2004; March & Smith, 1995; Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007). Thus, design research understands HRM as a class of tasks and conceptual modeling as a potential class of solutions for these tasks. Based on this perspective, two research tasks emerge. First, one most elaborate on which HR tasks one can solve with conceptual modeling and how one can do so. Second, one must elaborate on which HR solutions one can solve with conceptual modeling and how one can do so. Thus, we discuss both research tasks in this paper. Because design research attempts to offer general solutions for general tasks ("class level") and substantiate and concretize these with concrete solutions ("artifacts") for concrete tasks ("instance level") (e.g., March & Smith, 1995), we examine both research tasks on two different levels of abstraction. On a general level, we discuss the overarching application space and the overarching solution space of conceptual HR modeling. On a concrete level, we use the exemplarily selected task of assigning employees to complex task sequences (e.g., Kuhn, 1955; Niknafs, Denzinger, & Ruhe, 2013) to demonstrate an exemplary concrete problem class and to develop a corresponding exemplary concrete solution class of different artifacts. In the final section, we discuss the results that we obtained from this procedure and draw conclusions for future research on conceptual HR modeling (CHRM).

2 Task: Application Areas of Conceptual HR Modeling

2.1 General Application Areas of Conceptual HR Modeling

The first research task refers to identifying the HR task classes that conceptual modeling can actually address. Similar to other already established application domains of conceptual modeling, in HRM the overarching application areas refer to analyzing and designing (certain parts of) the HR domain (e.g., Frank, 2013; Guizzardi, 2007). Analysis and design tasks, however, are quite frequent in HRM. On the one hand, this frequency is certainly an advantage for conceptual modeling because it indicates a broad spectrum of application possibilities. On the other hand, the frequency of analysis and design tasks evidently aggravates the concretization of CHRM application areas. Because a presumably larger number of different HR task classes exist, an extensive elaboration process would be necessary to include all imaginable application areas. Moreover, given that identifying new application areas involves a creative discovery process, it is difficult to ensure the completeness of application areas in advance. For this reason, we discuss the general application areas of HR domain analysis and HR domain design in this section (see Figure 1) and subsequently concretize them based on the general suitability criterion of cognitive task complexity.

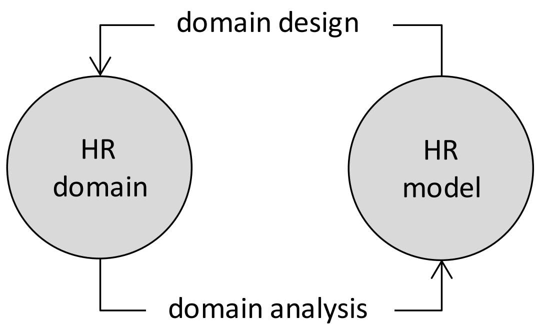


Figure 1. General Task Classes of CHRM

HR domain analysis is the first general task class that CHRM addresses. The analysis attempts to offer a basic understanding of (certain parts and problems of) the HR domain by building and evaluating descriptive domain models ("descriptive modeling"). Which concrete parts and tasks of HR one maps and examines depends on the respective purpose of the user and, thus, may differ considerably throughout different concrete task classes. The iterative steps of building and examining descriptive HR models amply contribute to a basic domain understanding because they force human modelers to explicitly "think through the domain" (e.g., Stachowiak, 1992; Thalheim, 2010). If the process involves several different modelers, this "collaborative modeling" also contributes to a common domain understanding (e.g., Renger, Kolfschoten, & de Vreede, 2008). HR domain design is the second abstract task class that CHRM addresses. The design attempts to create and improve (certain parts of) the HR domain through building and evaluating prescriptive domain models ("prescriptive modeling"). Modelers often build prescriptive HR models based on preceding descriptive HR models that they manually and/or algorithmically modify to meet certain goals. Again, the concrete parts of the prescriptively modeled HR domain depend on the respective goals, and modeled changes may range from correcting smaller flaws to completely revising the entire domain. The iterative steps of building and evaluating prescriptive HR models allow one to pre-evaluate any intended HR change on a model and, thus, avoid costly real-world aberrations (e.g., Stachowiak, 1992; Thalheim, 2010). Though HR domain analysis and design normally occur as consecutive steps, an isolated application of domain analysis *or* domain design is also imaginable depending on the specific HR purpose. To document and/or communicate a certain HR situation, a mere domain analysis is sufficient. In contrast, to design entirely new HR structures ("greenfield projects"), mere domain design is the appropriate (and also the only possible) variety of CHRM. Therefore, similar to other application domains in HRM, analyzing and designing (certain parts of) the domain are general application areas of conceptual modeling.

Based on the insights of general conceptual modeling research, one can specify suitable analysis and design tasks in HRM by a general suitability criterion. The general advantage of conceptual modeling in analyzing and designing real-world domains lies in its cognitive effectiveness (i.e., in the improved speed, ease, and accuracy with which a human mind can process conceptual models) (e.g., Cheng et al., 2001; Larkin & Simon, 1987; Moody, 2009). Against this backdrop, the cognitive complexity of a given HR analysis and/or design task can function as a general suitability criterion for CHRM. One can understand HR analysis and design tasks as cognitively complex if they—abstractly speaking—involve a larger number of relevant objects and attributes, relevant relations between these objects and attributes, and/or relevant events that dynamically change these objects, attributes, and relations. When a human actor must process more objects, attributes, relations, and events when analyzing and/or designing a HR domain task, the task's cognitive complexity grows. Evidently, CHRM's suitability increases with the increasing cognitive complexity of the analysis and/or design task. Because humans can effectively handle simple analysis and design tasks, such tasks do not justify investments in developing HR modeling languages and tools. A growing number of objects, attributes, relations, and events, however, will increasingly capitalize on conceptual modeling's cognitive effectiveness by facilitating and improving humans' analysis and design results. As such, although it is not possible to determine a definite threshold (e.g., by determining a concrete number of objects involved), the more cognitively complex the HR analysis and design task, the more CHRM suits it. Thus, in particular, highly complex HR problems that occur regularly justify the effort and costs of developing specific modeling languages and tools.

2.2 Exemplary Concrete Application Area of Conceptual HR Modeling

To concretize and demonstrate the general solution, design research additionally offers instantiations of the solution (i.e., concrete artifacts that reveal its feasibility and usefulness) (e.g., Hevner et al., 2004; Peffers et al., 2007). Therefore, we concretize CHRM based on the concrete task class of *employee assignment* (e.g., Kuhn, 1955; Niknafs et al., 2013). Employee assignment includes both analysis and design tasks; moreover, employee assignment is often a cognitive complex task. However, note that we arbitrarily selected employee assignment as a task class; one could also use other HR task classes that involve designing and analyzing components and show cognitive complexity (e.g., career and succession planning or staff rostering) as examples. Moreover, certain solutions already address the assignment problem (e.g., Kuhn, 1955; Niknafs et al., 2013); however, design research also refers explicitly to the search for new, potentially improved solutions to problems that already have certain solutions (e.g., Hevner et al., 2004; Peffers et al., 2007).

Employee assignment refers to matching a set of employees with a set of tasks based on certain properties of these tasks and employees (e.g., Kuhn, 1955; Niknafs et al., 2013). Because business tasks are not isolated but regularly arranged as temporally and logically ordered sets into business processes (e.g., van der Aalst, 2013; Weske, 2012), one assigns employees usually to the tasks of a business process. The main goal of employee assignment therewith is maximizing business process performance. Additional goals include minimizing employees' idle time and overtime, minimizing staffing costs, and maximizing on-the-job-training effects by an appropriate employee assignment. To reach these goals, one needs to identify the relevant properties of employees and tasks and use them as assignment criteria. Relevant properties include qualifications, wage, location, contract type, and preferences regarding tasks. Other relevant properties also include employees' coworkers, qualification requirements, and the order and duration of tasks (for a more extensive list of possible properties, see van den Bergh, Belien, de Bruecker, Demeulemeester, & de Boech, 2013).

Therefore, one must clearly categorize the employee-assignment problem as an instantiation of both general task classes of CHRM; that is, domain analysis and design. Identifying relevant employees and tasks, including related properties, and deriving corresponding assignment possibilities and constraints evidently involve a domain analysis task. Subsequently, assigning employees to specific tasks falls under domain design. Both, the analysis as the design tasks evidently show considerable cognitive complexity. Employees and tasks constitute relevant objects, the above-mentioned qualifications, wages, contracts, and so on, and existing or potential assignments constitute relevant relations. In particular, larger companies have a large set of heterogeneous employees and an even larger set of heterogeneous tasks that regularly culminate in an abundance of assignment possibilities, which create task complexity in employee assignment. Moreover, employee assignment is a frequent task class that regularly occurs in companies. Based on these characteristics, employee assignment is a HR task class that basically suits CHRM.

3 Solution: Design and Usage of Conceptual HR Modeling

After describing the task classes that conceptual modeling can address, the subsequent task of design research involves elaborating on the class of solutions that CHRM can offer (e.g., Hevner et al., 2004; Peffers et al., 2007). For this purpose, we elaborate CHRM in this section based on the existing knowledge regarding conceptual modeling (Moody, 2009; Paige, Ostroff, & Brooke, 2000; Ráth, Ökrös, & Varró, 2010; Recker, 2012; Thalheim, 2012),and particularly regarding domain-specific conceptual modeling (Frank, 2013; Gray et al., 2007; Kelly & Pohjonen, 2009; Kolovos, Paige, Kelly, & Polack, 2006; Ledeczi et al., 2001). In summarizing the existing knowledge of these sources, we derived a general framework that uncovers the necessary steps and intended results of CHRM, and we basically distinguish between the initial design phase and the subsequent usage phase of CHRM (see Figure 2).

The design phase involves the subsequent development of three artifacts. The first artifact is a HR domain ontology that has to be derived from the HR domain. The HR ontology serves as a systematic foundation for a HR modeling language, which is the second artifact. The HR modeling language is implemented in and complemented with a HR modeling tool as the third artifact. The subsequent usage phase involves using the modeling language and tool to analyze and design the HR domain.

3.1 General Design of Conceptual HR Modeling

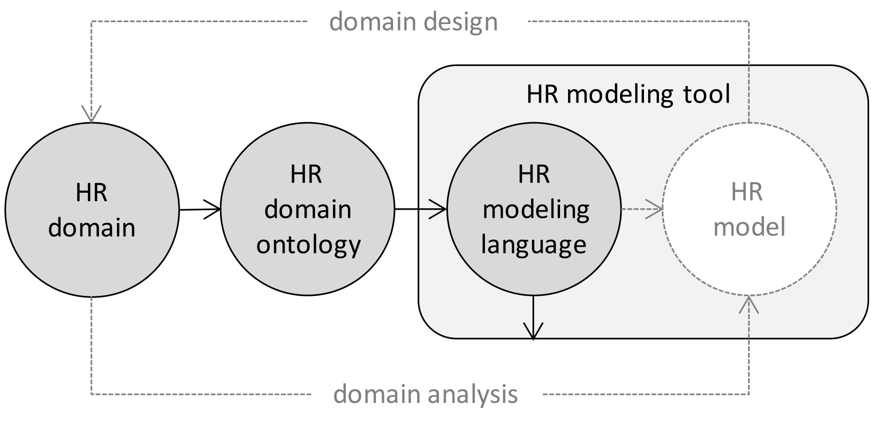
3.1.1 General Design of a HR Domain Ontology

The first design step is to develop a HR domain ontology (e.g., Shanks, Tansley, & Weber, 2003; Tairas, Mernik, & Gray, 2009; Weber, 2003). Ontologies are theories on the basic elements that exist in the real world (e.g., Bunge, 1977; Chandrasekaran, Josephson, & Benjamins, 1999). Top-level ontologies refer to abstract aspects of reality, such as time, space, objects, attributes, or relations, while domain ontologies refer to more concrete aspects that refer to (certain parts of) a specific domain (e.g., Chandrasekaran et al., 1999; Guarino, 1998). Researchers commonly view domain ontologies as an appropriate and necessary foundation for conceptual modeling languages because they directly uncover the relevant concepts and rules that a modeling language must consider and implement (e.g., March & Allen, 2009; Tairas et al., 2009; Weber, 2003).

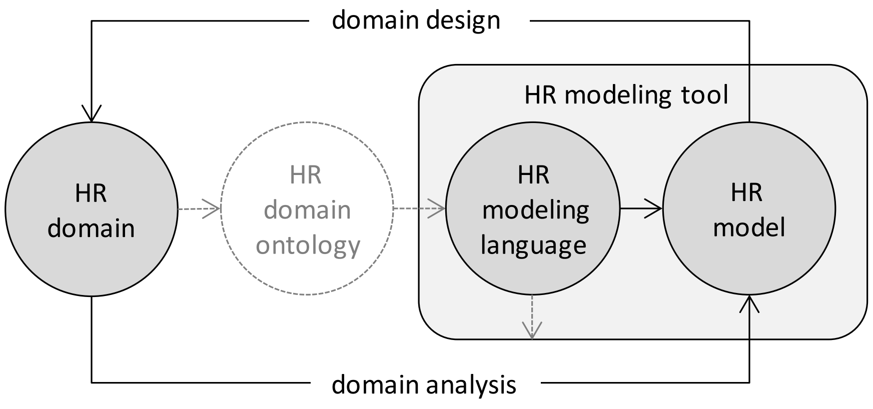
To obtain HR domain ontologies, one can either use (and eventually modify) already existing HR ontologies or develop new HR ontologies (e.g., Tairas et al., 2009). Because researchers have developed few HR ontologies (Jarrar, Vervenne, & Maynard, 2007), usually one has to develop specific HR ontologies. However, no generally accepted and detailed method for developing domain ontologies exists. Thus, one must employ one of the available heuristics to develop domain ontologies (Bergman, 2010).

Although they are called domain ontologies, these ontologies do not attempt to map the HR domain exhaustively; rather, they selectively address the HR aspects that are relevant considering a specific HRM purpose (e.g., Chandrasekaran et al., 1999; Guarino, 1998). Thus, one may also call partial HR domain ontologies that refer to specific HR tasks or that focus on founding specific HR applications task or application ontologies (Guarino, 1998).

As lower-level ontologies, domain ontologies should be based on top-level ontologies (e.g., Guarino, 1998). To this end, one can use (e.g., Chandrasekaran et al., 1999; Guarino, 1998) existing philosophical ontologies (e.g., Bunge, 1977) or existing ontologies specifically developed for conceptual modeling (e.g., Guizzardi & Wagner, 2010).



a) CHRM design phase



b) CHRM usage phase

Figure 2. General Framework of CHRM

3.1.2 General Design of a HR Modeling Language

The second design step is to develop a HR modeling language (also “grammar” or “meta-model”; e.g., Frank, 2013; Luoma, Kelly, & Tolvanen, 2004; Paige et al., 2000). The HR modeling language provides a set of concepts and rules to combine these concepts to model the HR domain; the syntax of the language refers to the entirety of the concepts and rules (e.g., Frank, 2013; Wand & Weber, 2002). The basis for deriving the syntax is the previously developed domain ontology that directly provides relevant concepts and rules. Researchers widely recognize that a modeling language requires ontological completeness (no missing language concepts) and ontological clarity (no redundant, overloaded or superfluous language concepts) (e.g., Frank, 2013; Wand & Weber, 2002). In addition to these ontological criteria, an abundant and heterogeneous set of further requirements for conceptual modeling languages exists (Fettke, 2009; Frank, 2013; Kolovos et al., 2006; Moody, 2009; Paige et al., 2000). Besides rather obvious requirements, such as simplicity or comprehensibility, *modularity*, *granularity* and *compatibility* in particular are directly design-relevant requirements. Modularity refers to the possibility of providing different parts of the language (“modules”) to different user groups; for example, offering a “light” version for novice or occasional users and a full version for power users (e.g., Frank, 2013; Tolvanen & Kelly, 2010). Granularity refers to the possibility of modeling a domain situation on different levels of abstraction (e.g., on an overview and a detailed level) (e.g., Frank, 2013; Smirnov, Reijers, Weske, & Nugteren, 2012). Finally, compatibility refers to the possibility of integrating a given HR modeling language with other HR modeling languages that address related HR problems (e.g., Kolovos et al., 2006; Tolvanen & Kelly, 2010).

Although conceptual models are basically graphical models, they do not visualize all aspects for simplicity. As a consequence, one must develop two related types of syntax: abstract syntax and concrete syntax. Abstract syntax refers to the full set of concepts and rules of a language (e.g., Frank, 2013; Ráth et al., 2010). The closely related concrete syntax (also "visual syntax", "visual grammar" or "graphical notation") refers to the subset of concepts and rules of a language that actually shows a visual representation via symbols (Moody, 2009; Ráth et al., 2010). Thus, a basic decision in modeling language design lies in deciding which parts are graphically represented and which parts are not (i.e., which parts one can see only through the modeling tool's property sheets).

Accordingly, a modeling language offers the "building blocks" for creating a model, and it delimits all possible models that one can construct using the language. One must design the modeling language so that it enables one to develop HR models that actually help one to analyze and design a HR domain (e.g., Frank, 2013; Gemino & Wand, 2004). A HR modeling language, therefore, constitutes a core artifact, and its development is a core step in CHRM.

3.1.3 General Design of a HR Modeling Tool

Although the HR modeling language already allows one to model on a "paper-and-pencil" basis, a third and final design step involves developing a corresponding HR modeling tool (e.g., Frank, 2013; Mylopoulos, 1998; Recker, 2012). The modeling tool refers to a software application with two major functions: 1) providing and 2) complementing the HR modeling language.

The core component for providing the language is a modeling editor that offers functions to create and explore HR models (e.g., Mylopoulos, 1998; Recker, 2012). Creation functions allow one to select and compose symbols as the modeling language intends. Automated syntax checks of the editor should accompany the model creation to avoid syntactical mistakes of the user ex ante. Exploration functions allow one to navigate through larger model diagrams and search for specific parts of the model. Incorporated in this way, one must understand the modeling language as a core part of the modeling tool.

Beyond providing the modeling language, however, modeling tools can and should offer additional functions that complement the modeling language. First, a tool should allow one to manage models (i.e., help one to systematically store, provide, and version HR models). If a HR modeling tool provides more than one HR modeling language (which one may call a "HR modeling suite"), the cross-referencing between different interrelated models also constitutes a necessary function (Kelly & Pohjonen, 2009; Recker, 2012). Second, a tool may also allow one to provide reference models (also "templates", "generic models" or "model patterns"). Reference models are prefabricated, generally valid (parts of) HR models that a larger user group can directly use as a basis for its modeling. Users typically use reference models to not only reduce the manual modeling effort but also obtain missing domain knowledge inherent in the reference models (e.g., Anaby-Tavor et al., 2010; Fettke & Loos, 2003). Third, a tool may also support HR-specific tasks (e.g., Recker, 2012). The type of support clearly depends on the specific HR problem class that the language addresses. For instance, if a modeler uses CHRM in succession planning, the tool may help the modeler automatically assign employees to (future) positions via an algorithm. In this way, automatically generated prescriptive HR-succession models can support users. Fourth, a tool may allow one to make user-specific language extensions and modifications. If HR domain situations vary and/or change considerably and unforeseeably, users may find it useful to customize the abstract and concrete parts of the modeling language to meet their specific requirements (e.g., Recker, 2012).

Once completed, HR modeling tools enable users in the subsequent utilization phase to create descriptive domain models, develop suitable prescriptive models that the tool complements or even suggests, and implement these models in the domain. The literature includes a voluminous and heterogeneous set of model requirements for the HR models that one creates in the utilization phase (e.g., Krogstie, Sindre, & Lindland, 2013; Miao, Sun, Ge, & Ren, 2013; Moody, 2005). Major requirements include syntactical correctness (a model's compliance with the syntactical rules of the language), semantic correctness (a model's validity and completeness regarding relevant aspects of the domain) and pragmatic correctness (a model's suitability to solve the intended HR task) (e.g., Krogstie et al., 2013; Thalheim, 2012).

In this way, the HR modeling tool not only enables one to conveniently apply the HR modeling language but also offers valuable additional user support. Thus, the HR modeling tool constitutes a third artifact that, similar to other application domains, is highly critical to CHRM's practical success (e.g., Davies, Green, Rosemann, Indulska, & Gallo, 2006; Recker, 2012).

3.2 Exemplary Concrete Design of Conceptual HR Modeling

In the next step of design research, one must demonstrate the feasibility of the general design procedure based on instantiations of the solution. Therefore, we present the interrelated artifacts of a HR domain ontology, a HR modeling language, and a HR modeling tool for assigning employees in Sections 3.2.1 to 3.2.2.

3.2.1 Concrete Design of a HR Ontology

For the first artifact, we created a domain ontology for employee assignment. The process of constructing the ontology comprises two main steps: 1) choosing an upper-level ontology to base the domain ontology on and 2) developing the domain ontology in reference to this upper-level ontology. One conducts the development by identifying the relevant domain concepts, describing these concepts and determining the upper-level ontological concepts that they refer to, and elaborating the domain constraints that affect the concepts. Because any domain concept "inherits" the abstract characteristics of the upper-level ontological concept that it refers to, the specification of the upper-level concepts plays a fundamental role. For instance, if a domain concept refers to the upper-level concept of "thing" and if "thing" shows the characteristics of "relations to other things" and "properties", one must identify the relevant domain relations and properties that exist. Elaborating on domain concepts constraints also refines the domain ontology because constraints represent the business rules of the inspected domain and the general semantics associated with a concept (Noy & McGuinness, 2001; Sugumaran & Storey, 2002).

Following this process, we based our ontology on a well-established upper-level ontology: the Bunge-Wand-Weber ontology (e.g., Rosemann & Green, 2002; Wand & Weber, 1990a, 1990b). To identify relevant lower-level ontological concepts, we analyzed the scientific literature on the employee-assignment problem (e.g., Constantopoulos, 1989; de Bruecker, van den Bergh, Belien, & Demeulemeester, 2014; Holness, Drury, & Batta, 2006; Holness, 2003; Niknafs et al., 2013; Peters & Zelewski, 2007), on modeling tasks in the frame of business processes (Leyking & Angeli, 2008; Mendling, 2008; Scheer et al., 2005; van der Aalst, 1999), and on already existing ontologies for the related employee-scheduling problem (e.g., Rajpathak, Motta, & Roy, 2001; Smith & Becker, 1997).

We exemplify the steps that we followed to elaborate our domain ontology based on the "employee" concept. The first step involves identifying the relevant domain concepts. The "employee" concept is obviously relevant because it represents a major component of the assignment problem (e.g., Constantopoulos, 1989; Niknafs et al., 2013). In the next step, one must describe the concept adequately. One can define an "employee" as a person who works for pay in a contractual relationship with the organization. In the third step, one should categorize the concept into the upper-level ontology. Accordingly, "employee" refers to the upper-level concept of "thing", which shows the characteristics of having properties, relationships, and constraints (e.g., Rosemann & Green, 2002; Wand & Weber, 1990a, 1990b). Therefore, the next step involves elaborating on the relevant domain relationships and properties of the "employee" concept. A relevant relationship of the "employee" concept, for instance, refers to the "function" concept, and one can concretize the relationship as "is responsible for". A relevant property of the "employee" concept, for instance, refers to the "qualifications" of "employees" because these constitute a major criterion to (not) assign employees to functions (e.g., Niknafs et al., 2013; van den Bergh et al., 2013). In the final step, one must identify the relevant constraints of the domain concept. A simple example of a constraint is that an "employee" is contractually constrained regarding their working hours per time unit. Elaborating all relevant domain concepts in this way results in an ontology that richly describes the domain (Table 1 provides a smaller exemplary sample of the ontology).

Table 1. Sample of the HR Ontology

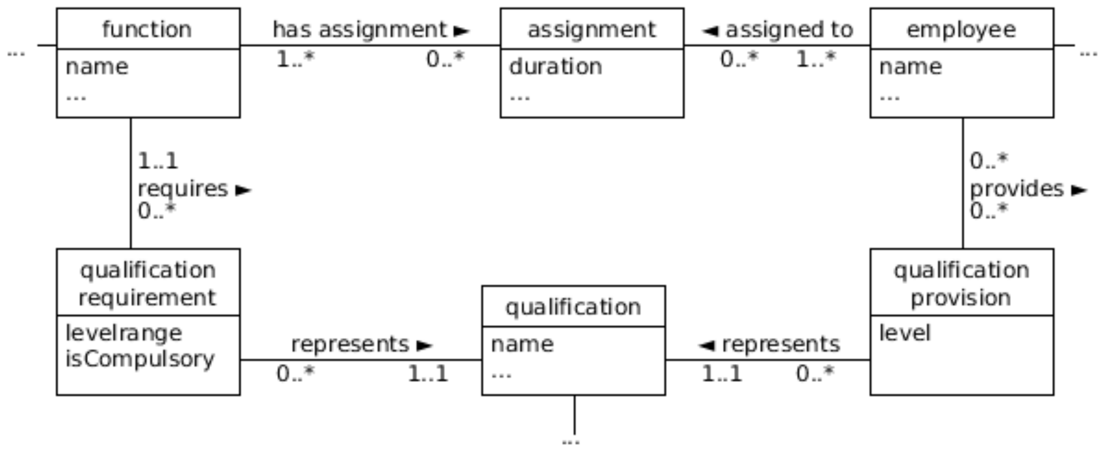
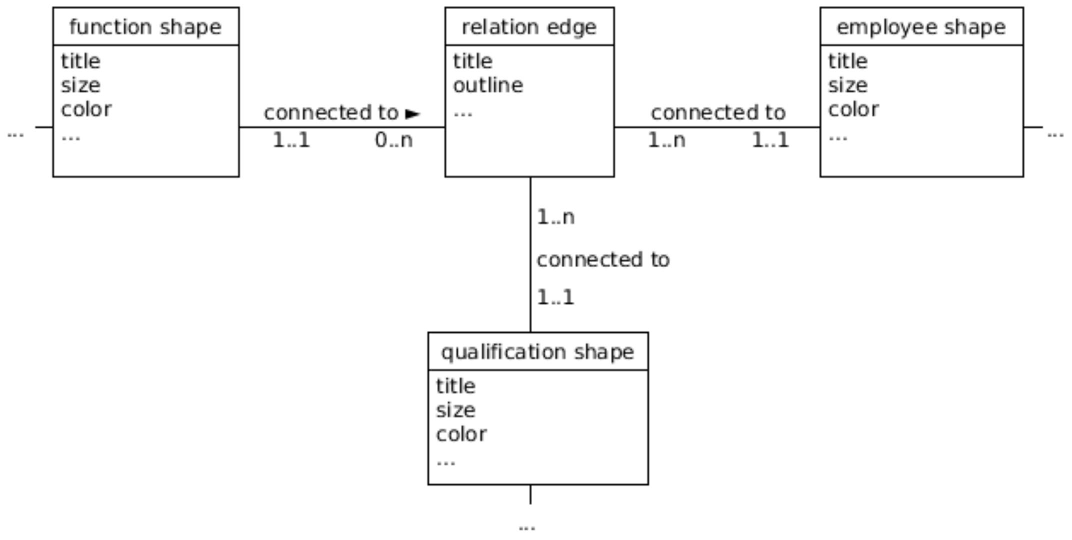
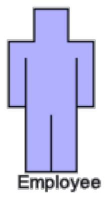
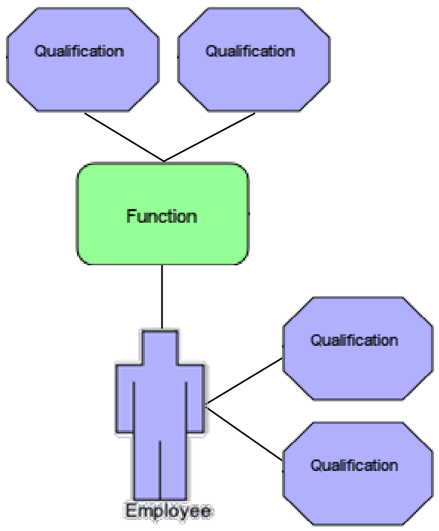
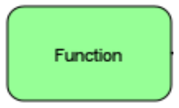
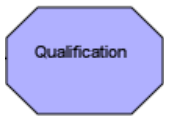
Domain concept	**Description**	**Upper-level concept**	**Concept-dependent characteristics**	
	Constraints			
Employee	… a person with a contractual relationship with the organization who works for pay.	Thing	Relation to • Function • Employee • …	Properties • Name • Qualifications • Working hours • Time spent / function • …
	• An employee is limited in the amount of working hours per time unit • An employee has certain qualifications that determine the functions he / she can perform • …			
Function	… a task that should be performed to allow an organization to meet its business goals.	Thing	Relation to • Organization • Employee • Resource • …	Properties • Duration • Qualifications • Location • …
	• A function takes a specific (mean) amount of time • A function requiring a specific resource cannot be executed at the same time as another function requiring the same resource • …			
Assignment	… responsibility of an employee for performing a specific function.	Relation	N / A	
	• Assignments must consider employee working hours per time unit as upper limit • …			
Qualification	… a specified ability offered by an employee and/or required by a function.	Property	Property of • Employee • Function	
	• If the qualifications of an employee do not match with required compulsory qualifications of a function this employee cannot be assigned to the function • …			
…	…			

3.2.2 Concrete Design of a HR Modeling Language

Based on the domain ontology, we developed the HR domain language as the second artifact. Because the language focuses on assigning employees to ordered sets of tasks (i.e., business processes), we employed an existing prominent suggestion of business process modeling—event-driven process chains (EPC) (e.g., Mendling, 2008; Scheer et al., 2005; van der Aalst, 1999)—to model the task side of the assignment problem; we developed additional language parts from scratch. This partial reuse of already existing modeling languages avoids redundancy and facilitates efficiency in language development (e.g., Vallecillo, 2010). As we describe above, developing a modeling language includes the defining the abstract and concrete syntax. Although based on the domain ontology, these definitions still leave a certain scope for design and, thus, different equally suitable possibilities to define the syntax. We aimed for a workable syntax that is easy to comprehend and implement.

We directly derived the abstract syntax from the domain ontology because it provides all relevant concepts and rules. Considering the requirements of ontological completeness and clarity, the abstract syntax must basically map the domain ontology. However, one sometimes needs to concretize the concepts in the ontology. One typically presents the abstract syntax as a graphical meta-model based on modeling languages from software engineering because this approach allows for an unambiguous and systematic description. Therefore, we created a UML class diagram to describe the abstract syntax (see Table 2).

Table 2. Sample of the HR Modeling Language Syntax

Abstract syntax	 function name ... has assignment ► 1..* 0..* assignment duration ... ◄ assigned to 0..* 1..* employee name ... 1..1 requires ► 0..* 0..* provides ► 0..* qualification requirement levelrange isCompulsory represents ► 0..* 1..1 qualification name ... ◄ represents 1..1 0..* qualification provision level ...			
Concrete syntax	**Meta model**	 function shape title size color ... connected to ► 1..1 0..n relation edge title outline ... connected to 1..n 1..1 employee shape title size color ... 1..n connected to 1..1 qualification shape title size color ...		
	Graphical notation	Employee shape		 Qualification Qualification Function Qualification Qualification Employee
		Function shape		
		Qualification shape		
		Relation edge	———	
	…	…	…	…

We considered the ontological concepts of employees and functions (including their qualification provisions/requirements and the assignment relation) as Table 1 shows. Although an assignment ontologically constitutes a relation, by definition, it does not show properties; for language convenience, we modeled the assignment as a class with attributes, which is necessary to consider ontological constraints. We used the same approach with qualifications. Although qualifications ontologically comprise one single concept, we represented them in three explicit classes of qualification, qualification requirement, and qualification provision for language convenience and utility.

The abstract syntax defines the general concepts of the modeling language, whereas the concrete syntax establishes the visual representation of the concepts. One uses a meta-model to generally describe the graphical representations of the abstract syntax and their interactions. As such, we again created a UML class diagram to describe the meta-model of the concrete syntax (see Table 2).

As we describe above, the concrete syntax includes only a subset of the elements of the abstract syntax. We decided that shapes should not show the qualification requirement, qualification provision, and assignment explicitly but that the edges between function shapes, employee shapes, and qualification shapes should show them implicitly. We then transferred the meta-model into a concrete graphical notation with different shapes and edges. We carefully considered the rules for graphical notation design (Frank, 2013; Moody, 2009). For instance, to emphasize different semantic categories, all employee-related concepts show the same color. To address perceptual discriminability, we used clearly different shapes for different concepts, and we preferred semantic reference icons over geometric shapes where possible. Furthermore, all shapes systematically follow the principle of dual coding (Moody, 2009); that is, we use text to complement the graphical notation. Table 2 depicts a smaller sample of graphical elements and their combination based on the rules from the meta-model.

3.2.3 Concrete Design of a HR Modeling Tool

For the final artifact, we designed a HR modeling tool. As we discuss above, a modeling tool has two major functions: 1) to enable the creation of HR models based on the HR modeling language and 2) to provide additional functionalities that complement the HR modeling language in creating and using the HR model. One can directly derive the necessary components of the tool from these two intended functionalities. Obviously, a modeling editor constitutes a core component of any HR modeling tool because one needs it to use the language. Moreover, one also obviously needs a complementation component that offers diverse complementing functionalities. To demonstrate complementation functions, we developed several useful subcomponents as described below, but one can imagine further complementation functions. Given the necessity to store and (re-)provide the created HR models, we also needed a model repository. Therefore, a modeling editor, complementation components, and a model repository constitute the general core components of the tool (see Figure 3).

To capitalize on the advantages of Web-based applications, we decided to realize the tool as a Web-application with server-side and client-side components. Although there is a certain scope for decision making regarding the location of the respective components, we generally placed the components that refer to several different models and users at the server and the components that (mostly and usually) refer to one single model and one single user at the client (e.g., Jeffery, Maes, & Bratton-Jeffrey, 2005; Rittgen, 2009). For instance, the generation of reports may simultaneously refer to several different models of several different users. Thus, the report-generation component is located at the server. Conversely, as a rule, syntax checks refer to single models of single users; thus, we located the syntax check component at the client. Likewise, we split the complementation component into two parts: one located at the client-side and one located at the server-side of the tool. We briefly depict the individual components in this section.

The central component of the tool is certainly the graphical modeling editor. It basically allows one to view, edit, and create HR assignment models. Because the modeling editor represents the primary user interface, it also delivers the respective complementation functions. To ensure usability, the editor is based on well-established functionalities of contemporary graphical user interfaces, such as the “drag and drop” of graphical items (see Figure 4).

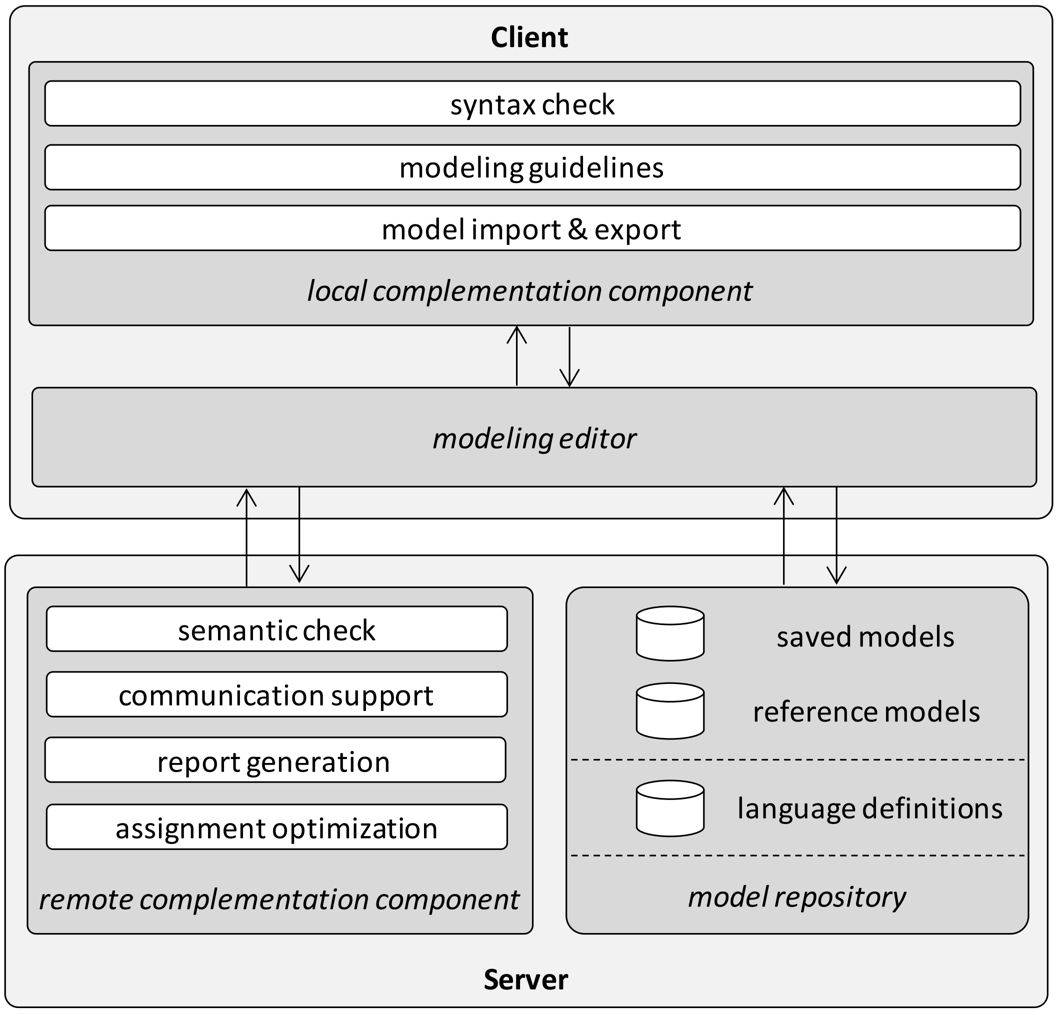
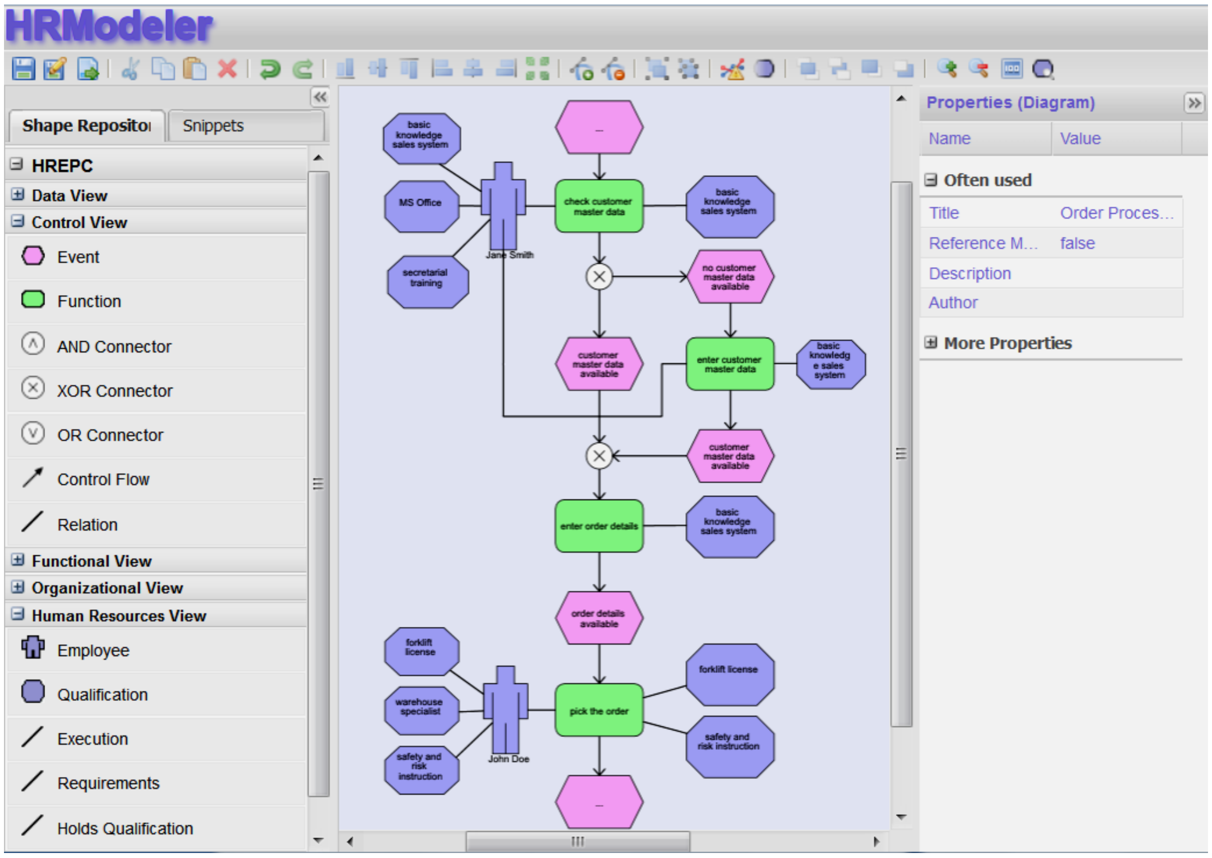


Figure 3. Architecture of the HR Modeling Tool

Furthermore, the editor offers the possibility to show or hide selected elements of the model and, thus, to display only the elements that are relevant to a specific application situation.

A local complementation component with three subcomponents supports the modeling editor: 1) syntax checks, 2) modeling guidelines and documentation, and 3) the ability to import or export models. First, syntax checks prevent users from creating syntactical errors ex ante by not allowing obvious false modeling, and they prevent errors ex post by checking whether a model considers all relevant syntax rules. For instance, the syntax of our language does not allow one to connect an employee shape with an event shape. If a user who is not familiar with this rule tries to establish this connection, the syntax check will recognize this attempt as inadmissible and prevent it. Second, modeling guidelines and documentation of the different usable elements simplify the modeling process. In particular, novice users can refer to this component to basically learn the modeling language and the operation of the tool. In this way, one can clarify open issues on modeling and tool application in a self-determined way. Third, users can import or export models to avoid costly and effortful “duplicate modeling”. For instance, because many companies already dispose of numerous process models, one can easily import these models in the tool and directly use them to assign employees.



Name	Value
Title	Order Proces...
Reference M...	false
Description	
Author	

Figure 4. Modeling Editor of the HR Modeling Tool

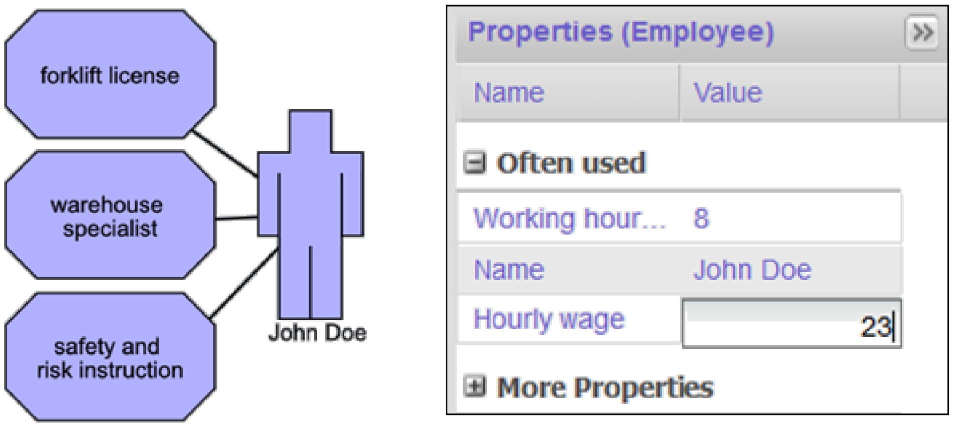
Server-based complementation functions complement this client-based complementation. As a first important component, a semantic check evaluates the compliance of the models with the domain's constraints and requirements. For example, because employees show an upper limit of daily, weekly, and so on working hours, this component checks whether an employee has sufficient time capacity to be assigned to other tasks and prevents new assignments if one has reached the upper limit. Additionally, the complementation component offers communication possibilities for different users. These possibilities refer to, for instance, annotations of the models regarding problematic aspects or unclear modeling choices. Communication possibilities should support and enable several users who are spatially distributed to cooperatively build one model. A third complementation function involves report generation (i.e., the possibility of querying models regarding any interesting aspect, such as the number of all functions, employees with certain qualifications, functions without assigned employees, and so on). Finally, assignment optimization, which supports users by automatically assigning employees to functions, offers another important domain-specific complementation. This component initially allows one to assign one employee to one function based on diverse criteria, such as qualifications, capacities, or costs. In a comparable manner, the component also allows an automated collective assignment of several employees to several tasks.

The final component, the model repository, persistently stores all HR models created with the modeling editor and makes them available for further processing by the modeling editor. The basis for this storage is language syntax definitions that are saved on the server-side and allow a mapping between the concrete syntax models that users create and the abstract syntax models that reflect the domain's semantics. The respective models that are held in the model repository can also act as reference models. One can mark any reusable (part of a) model as a reference model and make it available in the modeling editor to ease and accelerate further modeling tasks by reusing the reference model.

3.3 Exemplary Concrete Usage of Conceptual HR Modeling

To demonstrate how the modeling tool works, we show how to assign employees to the functions of an operative sales process.

First, the user of the tool must perform a domain analysis to develop a descriptive model of the respective assignment problem. One can begin a domain analysis by descriptively modeling the employees who are available to perform the operative sales process. One can easily model the employees by using the graphical elements (employee and qualification shapes) from the concrete syntax, which one can complement further with relevant employee properties (working hours, hourly wage, etc.) from the abstract syntax (see Figure 5).



Name	Value
Working hour...	8
Name	John Doe
Hourly wage	23

Figure 5. Descriptive Modeling of Employees

After modeling the available employees, one must model the operative sales process or, if a process model already exists in the company, import it using the import function of the tool. If one has to model the process, one can reuse a combination of graphical notations (such as events, functions or operators) and properties (such as duration or costs of a function) to model the entire process. Through descriptive modeling, one can create a comprehensive graphical model (complemented by non-graphical properties) of all employees and all functions. As a further result, users develop their own deep insights into the structure and peculiarities of the assignment task under consideration by elaborating on this descriptive model.

A second step of the assignment task involves the domain design (i.e., the user must develop a prescriptive model with concrete assignments of individual employees and functions). Basically, users can “manually” develop a descriptive model by searching for fitting pairs of employees and tasks and connecting them with a corresponding edge in the graphical model. The graphical model supports this manual assignment by, for instance, allowing one to easily overview the functions that are already assigned and the functions that still need to be assigned. However, given that usually a larger number of employees must be assigned to a larger number of functions based on different criteria, a manual assignment constitutes a voluminous and complex task. Therefore, the assignment optimization component offers automated suggestions of assignments for the user. For instance, if a user marks an unassigned function, the assignment component offers a list of suitable employees who are ranked by the degree of fit between employee and function. The user can then just select an employee from this list and assign the employee to the function (see Figure 6).

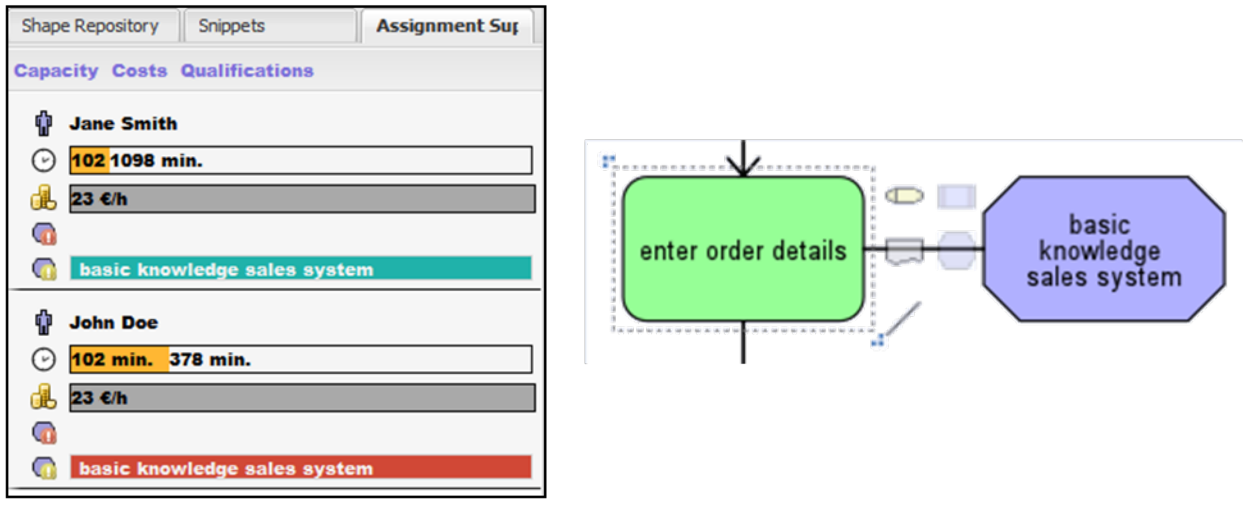
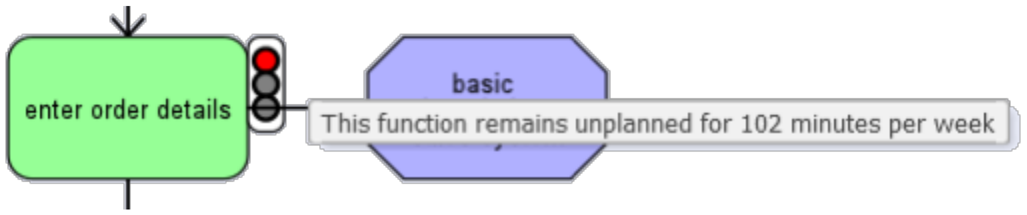


Figure 6. Individual Assignment Support

As an interim result, the user gains a draft model of employee assignments. Users can manually evaluate this draft model (i.e., by searching for functions without assigned employees or the qualification gaps of assigned employees). The graphical model that, for instance, allows one to easily identify functions without assigned employees supports this manual evaluation. However, again, the large number of assignments and the existence of non-visible assignment criteria, such as the limited working hours of employees, require users' input. The semantic check component helps users by evaluating the respective assignments automatically. Traffic light icons show the evaluation results, while mouse-over functions show existing problems. For instance, a function without an assigned employee will show a red traffic light to indicate a problem, while the mouse-over function will concretize the lack of assignment (see Figure 7).



This function remains unplanned for 102 minutes per week

Figure 7. Semantic Model Check

In this way, one can automatically detect and resolve existing assignment problems. The final diagram offers a prescriptive model of assigning employees to functions that one can directly implement. Figure 8 shows a small sample of the business process and the final assigned employees.

Therefore, we can see that one can design domain-specific HR modeling languages and tools. Against the backdrop of design research, this presentation constitutes a "proof-of-concept" validation of CHRM (e.g., Peffers et al., 2007).

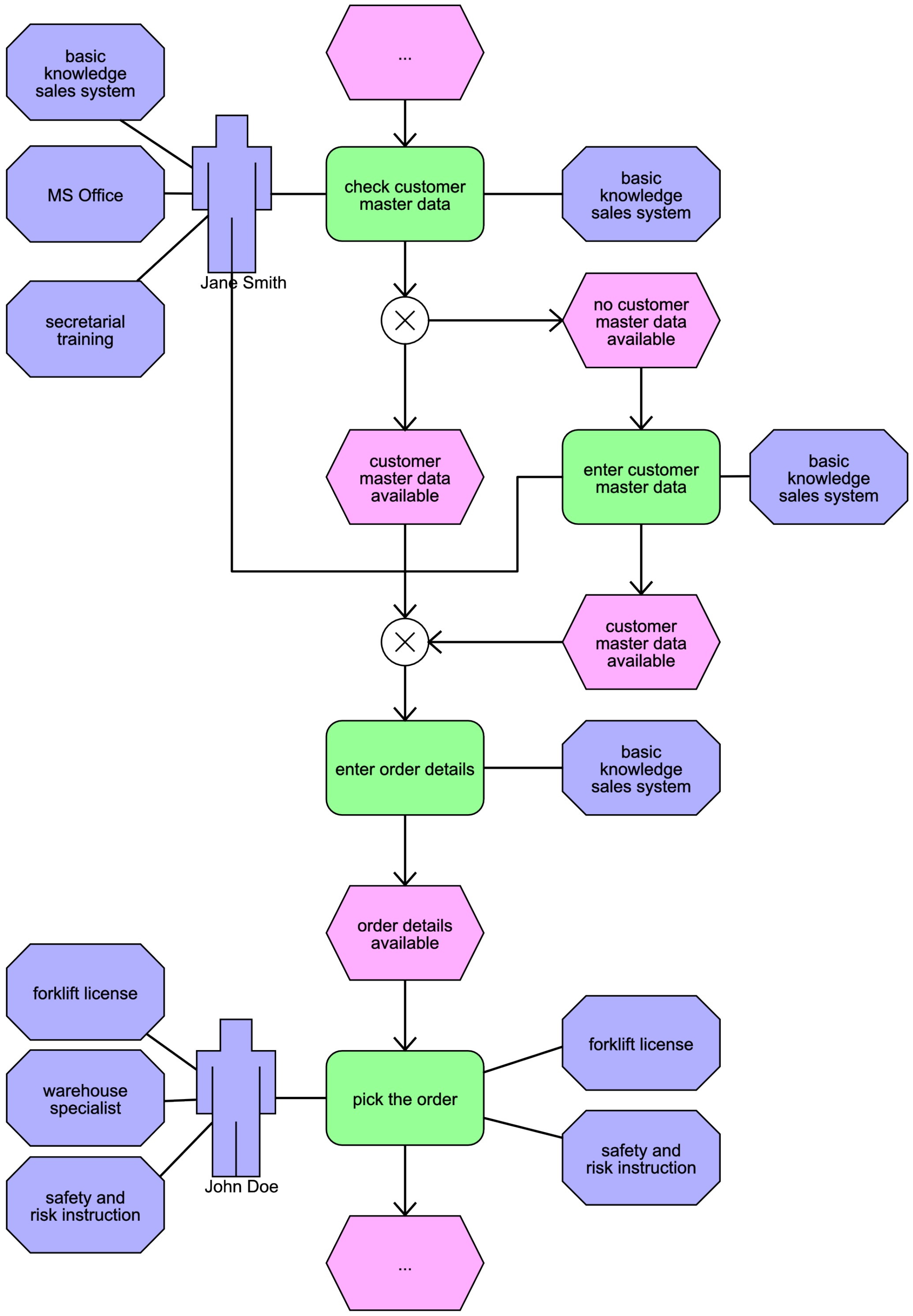

...
basic knowledge sales system
MS Office
secretarial training
Jane Smith
check customer master data
basic knowledge sales system
no customer master data available
customer master data available
enter customer master data
basic knowledge sales system
customer master data available
enter order details
basic knowledge sales system
order details available
forklift license
warehouse specialist
safety and risk instruction
John Doe
pick the order
forklift license
safety and risk instruction
...

Figure 8. Sample of the Prescriptive Assignment Model

4 Conclusion: Prospects of Conceptual HR Modeling

4.1 Lessons Learned

In this paper, we introduce the methodical paradigm of conceptual modeling to the HRM domain. In particular, we explore which HR tasks conceptual modeling can address and which HR solutions CHRM can offer. On a general level, we outline that CHRM shows the potential to support the analysis and design of the HR domain and that it is particularly suitable for cognitively complex HR tasks. Moreover, we elaborate a general process model of designing CHRM-related artifacts (i.e., HR domain ontologies, HR modeling languages, and HR modeling tools). On a concrete level, we use the task of assigning employees to complex sequences of tasks and show, based on this example, how one can put the general process model into practice. We created three interrelated artifacts: an employee-assignment ontology, an employee-assignment language, and an employee-assignment modeling tool. By using the modeling tool, we demonstrate that the assignment task is actually solvable based on CHRM. Therefore, we show that one can actually use conceptual modeling in HRM by following the respective design and usage steps we suggest here. Thus, in terms of design research, we provide a proof-of-concept validation of CHRM.

Compared with previous solutions and systems in the HR domain in general and in employee assignment in particular, CHRM has noteworthy peculiarities. The most important difference lies in the character-orientation of conventional systems and the graphics-orientation of conceptual modeling systems. Although they have graphical interfaces, conventional HR systems rely heavily on characters to represent real-world objects and attributes in the frame of forms and tables. Characters on buttons or menu items represent manipulation possibilities, and the results of the manipulation are the changes of these characters as presented in forms or tables. CHRM systems heavily use graphics to represent real-world objects and attributes in the frame of graphical models.

Compared with character-based representations, these graphical models are more concrete, more intuitive, and more vivid. One can manipulate relevant domain objects by literally “manipulating” the graphics with a mouse or finger to, for instance, assign certain objects, add new properties to objects, and so on. Thus, user interactions with the system allow for direct actions on the represented objects and allow for incremental reversible actions whose impact on the object of interest one can directly see. These abilities enable a successive self-directed exploration of the respective situation and an experimental search for appropriate solutions. Moreover, graphical models also facilitate and improve the communication among different users because they allow them to express, compare, and adjust different mental models of a given situation. This communication supports consent regarding the problem/situation among different involved parties. Although CHRM generally constitutes a powerful tool for visually analyzing and designing a HR domain, its effectiveness crucially depends on how well one designed the modeling language. The expressiveness of the language plays an important role in how well one can represent the problem domain. For instance, if the modeling language lacks the required elements, the system cannot model specific problems, and one can find no solution. Moreover, even based on appropriate designs, the resulting model’s inherent complexity can be an issue. The fact that conceptual models offer a wide range of information in a visual way, on the one hand, is clearly desirable because they can map the actual situation. On the other hand, this information may become overwhelming to the user in more complex models and situations where the problem comprises multiple subproblems with different characteristics. However, the problem of real-world complexity to an even greater extent applies to conventional character-based HR systems that demand an even greater act of abstraction from users. Moreover, modeling tools offer the possibility to display only specific views of the model to reduce the complexity.

4.2 Research Implications

Although this paper constitutes an initial small step towards CHRM, we still need much research regarding the three interrelated areas of theoretically explaining, technically designing, and empirically evaluating CHRM.

As for theoretical explanation, a future task lies in deepening our theoretical understanding of CHRM. Along this vein, we have seen diverse endeavors to develop specific theories of conceptual modeling that, for instance, refer to a theory of graphical notations (Moody, 2009) or a general theory of conceptual modeling (Thalheim, 2010). Of course, one can use the insights from these approaches to improve the future understanding, development and evaluation of CHRM. Moreover*,* cognitive science theories should offer a deeper insight into the human analysis and solution of problems based on graphical models compared with textual, tabular, or form-based representations of reality (e.g., Cheng et al., 2001; Gemino & Wand, 2003). Some example theories are the cognitive load theory (e.g., Sweller, 2006) or the cognitive theory of multimedia learning (e.g., Mayer, 2005). Additional information systems theories are also suitable to gain deeper insights into the requirements and the design of CRHM tools. Prominent theories refer to, for instance, the acceptance (e.g., Venkatesh, Morris, Davis, & Davis, 2003) or the success (e.g., DeLone & McLean, 2003) of information systems. Beyond these theoretical categories, additional theories may offer explanations of specific aspects of CHRM.

As for technical design, we offer only one suggestion for only one arbitrarily selected HR problem. Thus, one can imagine numerous further and different realizations of CHRM. To gain further experience with CHRM and to provide more evidence of CHRM's feasibility, a second task refers to further developing CHRM artifacts. The procedure that we present in this paper may offer a blueprint for further CHRM development projects. Further developments may refer to expanding the above CHRM suggestions to enable users to execute further HR tasks. A plain and easily realizable example is training need analysis, which can be realized by just adding a report, that compares the qualifications of employees with the qualification requirements of the functions that they are assigned to. Further developments may also integrate the present CHRM suggestion with additional compatible modeling languages. For instance, one may complement the current modeling language with a compatible modeling language for employee succession planning. Future developments may also design entirely new sets of HR domain ontologies, domain languages, and modeling tools. These methods should offer insights into different approaches of realizing CHRM. However, this also entails the risk of proliferating redundant and incompatible languages and tools (e.g., Moody, 2005; Mylopoulos, 1998). As a long-term objective, we need a HR modeling suite, which would incorporate a broader set of HR modeling languages for a broader set of HR tasks (e.g., Frank, 2013; Thalheim, 2010; Vallecillo, 2010). Interesting extensions could also enrich the modeling of HR tasks with social and psychological aspects, which existing "social modeling" approaches suggest (e.g., Cervenka, Trencansky, & Calisti, 2006; Yu, Giorgini, Maiden, & Mylopoulos, 2011; Yu, 2009). These approaches include multiple socio-psychological aspects (e.g., power and dependency relations; employees' motivations, goals, or beliefs; and so on) and overcome a merely "mechanistic" modeling of the HR domain by adding highly relevant social aspects.

As for empirical evaluation*,* we offer only a proof of concept. Thus, we urgently need work that empirically evaluates CHRM. To this end, existing frameworks on the empirical evaluation of general conceptual modeling may also offer valuable guidance for empirical CHRM research (e.g., Burton-Jones, Wand, & Weber, 2009; Gemino & Wand, 2004; Recker, 2005a, 2005b). Future evaluations should consider the relevant assumptions that underlie CHRM but that have not been investigated empirically to date. A major assumption refers to CHRM's cognitive efficiency (e.g., Moody, 2009). As a consequence, the evaluation of the speed, ease, and accuracy with which humans can process graphical representations of the HR domain doubtlessly constitutes an important empirical research topic. A related assumption is that the results gained via CHRM exceed or at least equal the conventional results achieved by, for instance, conventional HRIS. Obviously, this assumption requires empirical evaluation in the same way that numerous additional assumptions do. Given the presumable lack of CHRM adoption in HR practice, in the foreseeable future, empirical research will mainly refer to experimental designs that do not rely on a broader adoption of CHRM in practice (e.g., Parsons and Cole, 2005). Moreover, empirical research evidently relies on the existing CHRM languages and tools that one can use as stimuli. Thus, empirical evaluations must be preceded by technical design activities. Empirical studies may then evaluate single artifacts (e.g., one single HR modeling language) or may comparatively evaluate multiple artifacts (e.g., two or more HR modeling languages) (Parsons & Cole, 2005; Wand & Weber, 2002).

In sum, in this paper we show that conceptual modeling constitutes an interesting new methodical paradigm for HRM practice. Correspondingly, we uncover conceptual modeling as an interesting new topic for HRM research. We hope that this paper contributes to the spread of conceptual modeling in HRM research and practice.

References

Anaby-Tavor, A., Amid, D., Fisher, A., Bercovici, A., Ossher, H., Callery, M., Desmond, M., Krasikov, S., & Simmonds, I. (2010). Insights into enterprise conceptual modeling. *Data and Knowledge Engineering*, *69*(12), 1302-1318.

Bergman, M. (2010). A brief survey of ontology development methodologies. Retrieved from http://www.mkbergman.com/906/a-brief-survey-of-ontology-development-methodologies/

Bunge, M. (1977). *Ontology 1: The furniture of the world*. Boston, MA: Reidel.

Burton-Jones, A., Wand, Y., & Weber, R. (2009). Guidelines for empirical evaluations of conceptual modeling grammars. *Journal of the Association for Information Systems*, *10*(6), 495-532.

Cakar, F., & Bititci, U. S. (2002). Modeling the HRM business process. *International Journal of Human Resources Development and Management*, *2*(3/4), 223-248.

Cakar, F., Bititci, U. S., & MacBryde, J. (2003). A business process approach to human resource management. *Business Process Management Journal*, 9(2), 190-207.

Cervenka, R., Trencansky, I., & Calisti, M. (2006). Modeling social aspects of multi-agent systems: The AML approach. In J. P. Müller & F. Zambonelli (Eds.), *Agent-oriented software engineering* (LNCS 3950, pp. 28-39). Berlin: Springer.

Chandrasekaran, B., Josephson, J. R., & Benjamins, V. R. (1999). What are ontologies, and why do we need them? *IEEE Intelligent Systems*, *14*(1), 20-26.

Cheng, P. C.-H., Lowe, R. K., & Scaife, M. (2001). Cognitive science approaches to understanding diagrammatic representations. *Artificial Intelligence Review*, *15*(1/2), 79-94.

Constantopoulos, P. (1989). Decision support for massive personnel assignment. *Decision Support Systems*, *5*(4), 355-363.

Davies, I., Green, P., Rosemann, M., Indulska, M., & Gallo, S. (2006). How do practitioners use conceptual modeling in practice? *Data and Knowledge Engineering*, *58*, 358-380.

de Bruecker, P., van den Bergh, J., Beliën, J., & Demeulemeester, E. (2014). Workforce planning incorporating skills: State of the art. *European Journal of Operational Research*, *243*(1), 1-16.

DeLone, W. H., & McLean, E. R. (2003). The DeLone and McLean model of information systems success: A ten-year update. *Journal of Management Information Systems*, *19*(4), 9-30.

Dennis, A., Wixom, B. H., & Tegarden, D. (2015). *Systems analysis and design: An object-oriented approach with UML*. Hoboken: John Wiley & Sons.

Eichelberger, H., Schmid, K., & Eldogan, Y. (2011). *A comprehensive analysis of UML tools, their capabilities and their compliance*. Software Systems Engineering.

Fettke, P. (2009). How conceptual modeling is used. *Communications of the Association for Information Systems*, *25*, 571-592.

Fettke, P., & Loos, P. (2003). Ontological evaluation of reference models using the Bunge-Wand-Weber model. In *Proceedings of the Americas Conference on Information Systems* (pp. 2944-2955)..

Francesconi, F., Dalpiaz, F., & Mylopoulos, J. (2013). TBIM: A language for modeling and reasoning about business plans. In N. Wilfred, V. C. Storey, & J. Trujillo (Eds.), *Conceptual modeling* (LNCS 8217, pp. 33-46). Berlin: Springer.

Frank, U. (2013). Domain-specific modeling languages: requirements analysis and design guidelines. In I. Reinhartz-Berger, A. Sturm, T. Clark, S. Cohen, J. Bettin (Hrsg.), *Domain engineering* (pp. 133-157). Berlin: Springer.

Frank, U., Strecker, S., Fettke, P., Brocke, J., Becker, J., & Sinz, E. (2014). The research field “modeling business information systems”. *Business & Information Systems Engineering*, *6*(1), 39-43.

Gemino, A., & Wand, Y. (2003). Evaluating modeling techniques based on models of learning. *Communications of the ACM*, *46*(10), 79-84.

Gemino, A., & Wand, Y. (2004). A framework for empirical evaluation of conceptual modeling techniques. *Requirements Engineering*, *9*(4), 248-260.

Giannoulis, C., Zikra, I., Bergholtz, M., Zdravkovic, J., Stirna, J., & Johannesson, P. (2013). A comparative analysis of enterprise modeling approaches for modeling business strategy. In J. Grabis, M. Kirikova, J. Zdravkovic, & J. Stirna (Eds.), *PoEM Short Papers* (pp. 193-204). Retrieved from http://ceur-ws.org/Vol-1023/

Gray, J., Tolvanen, J.-P., Kelly, S., Gokhale, A., Neema, S., & Sprinkle, J. (2007). Domain-specific modeling. In P. A. Fishwick (Ed.), *Handbook of dynamic system modeling*. Boca Raton, FL: Chapman & Hall.

Guarino, N. (1998). Formal ontology and information systems. In *Proceedings of Formal Ontology in Information Systems.*

Guizzardi, G. (2007). On ontology, ontologies, conceptualizations, modeling languages, and (meta) models. In *Proceedings of the 7th International Baltic Conference on Databases and Information Systems* (pp. 18-39).

Guizzardi, G., & Wagner, G. (2010). Using the unified foundational ontology (UFO) as a foundation for general conceptual modeling languages. In Poli, R., Healy, M., & Kameas, A. (Eds.), *Theory and applications of ontology: Computer applications* (pp. 175-196). Berlin: Springer.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *28*(1), 75-105.

Holness, K. S. (2003). Personnel assignment from a human factors perspective. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *47*(12), 1394-1398.

Holness, K. S., Drury, C. G., & Batta, R. (2006). A systems view of personnel assignment problems. *Human Factors and Ergonomics in Manufacturing*, *16*(3), 285-307.

Ibrahim, R., Ahmed, A., Bahrudin, I. A., Moftah, A. A., & Algarai, E. E. (2014). Development of staff management system using UML-based object-oriented approach. In *Proceedings of the International Conference on Computing Technology and Information Management* (pp. 119-124).

Jarrar, M., Vervenne, L., & Maynard, D. (2007). *HR-semantics roadmap: The semantic challenges and opportunities in the human resources domain*. Brussels, Belgium: The Ontology Outreach Advisory.

Jeffery, A. B., Maes, J. D., & Bratton-Jeffery, M. F. (2005). Improving team decision-making performance with collaborative modeling. *Team Performance Management*, *11*(1/2), 40-50.

Kelly, S., & Pohjonen, R. (2009). Worst practices for domain-specific modeling. *IEEE Software*, *26*(4), 22-29.

Kolovos, D. S., Paige, R. F., Kelly, T., & Polack, F. A. C. (2006). Requirements for domain-specific languages. In *Proceedings of the 1st ECOOP Workshop on Domain-specific Program Development.*

Krogstie, J., Sindre, G., & Lindland, O. I. (2013). 20 years of quality of models. In J. Bubenko, J. Krogstie, O. Pastor, B. Pernici, C. Rolland, & A. Solvberg (Eds.), *Seminal contributions to information systems engineering* (pp. 103-107). Berlin: Springer.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1-2), 83–97.

Larkin, J. H., & Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science*, *11*(1), 65-99.

Ledeczi, A., Bakay, A., Maroti, M., Volgyesi, P., Nordstrom, G., Sprinkle, J., & Karsai, G. (2001). Composing domain-specific design environments. *Computer*, *34*(11), 44-51.

Leyking, K., & Angeli, R. (2008). Model-based competency-oriented business process analysis. In P. Loos, M. Nüttgens, K. Turowski, & D. Werth (Eds.), *Modellierung betrieblicher Informationssysteme—modellierung zwischen SOA und compliance management* (pp. 39-58). Saarbrücken: Gesellschaft fur Informatik.

Luoma, J., Kelly, S., & Tolvanen, J.-P. (2004). Defining domain-specific modeling languages: Collected experiences. In *Proceedings of the 4th Workshop on Domain-Specific Modeling* (pp. 198-209).

March, S. T., & Allen, G. N. (2009). Challenges in requirements engineering: A research agenda for conceptual modeling. In K. Lyytinen, P. Loucopoulos, J. Mylopoulos & B. Robinson (Eds.), *Design requirements engineering: A ten-year perspective* (vol. 14, pp. 157-165). Berlin: Springer.

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251-266.

Mayer, R. E. (2005). Cognitive theory of multimedia learning. In R. E. Mayer (Ed.), *The Cambridge handbook of multimedia learning* (2nd ed., pp. 31-48). New York, NY: Cambridge University Press.

Mendling, J. (2008). Event-driven process chains (EPC). In W. Aalst, J. Mylopoulos, N. M. Sadeh, M. J. Shaw, & C. Szyperski (Eds.), *Metrics for process models: Empirical foundations of verification, error prediction, and guidelines for correctness* (pp. 17-57). Berlin: Springer.

Meegama, R. G. N., & Ranatunga, S. P. K. (2009). UML-based framework for human resource management. *International Journal of Software Engineering and Computing*, *1*(1), 31-35.

Miao, H., Sun, J., Ge, S., & Ren, N. (2013). A literature review on conceptual model quality of information systems. *International Journal of Digital Content Technology and its Applications*, *7*(5), 574-583.

Moody, D. L. (2005). Theoretical and practical issues in evaluating the quality of conceptual models: Current state and future directions. *Data & Knowledge Engineering*, *55*, 243-276.

Moody, D. L. (2009). The “physics” of notations: Toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering*, *35*(6), 756-779.

Mylopoulos, J. (1992). Conceptual modelling and Telos. In P. Loucopoulos & R. Zicari (Eds.), *Conceptual modeling, databases, and case an integrated view of information systems development* (pp. 49-68). New York, NY: John Wiley & Sons.

Mylopoulos, J. (1998). Information modeling in the time of the revolution. *Information Systems*, *23*(3-4), 127-155.

Niknafs, A., Denzinger, J., & Ruhe, G. (2013). A systematic literature review of the personnel assignment problem. In S. I. Ao, O. Castillo, C. Douglas, D. D. Feng, & J.-A. Lee (Eds.), *Proceedings of the International MultiConference of Engineers and Computer Scientists* (pp. 1121-1128). Hong Kong, China: Newswood Limited.

Noy, N. F., & McGuinness, D. L. (2001). *Ontology development 101: A guide to creating your first ontology* (Laboratory Technical Report KSL-01-05). Stanford Knowledge Systems.

Paige, R. F., Ostroff, J. S., & Brooke, P. J. (2000). Principles for modeling language design. *Information and Software Technology*, *42*(10), 665-675.

Parsons, J., & Cole, L. (2005). What do the pictures mean? Guidelines for experimental evaluation of representation fidelity in diagrammatical conceptual modeling techniques. *Data and Knowledge Engineering*, *55*, 327-342.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, *24*(3), 45-77.

Peters, M. L., & Zelewski, S. (2007). Assignment of employees to workplaces under consideration of employee competences and preferences. *Management Research News*, *30*(2), 84–99.

Rajpathak, D., Motta, E., & Roy, R. (2001). A generic task ontology for scheduling applications. In *Proceedings of the International Conference on Artificial Intelligence.*

Ráth, I., Ökrös, A., & Varró, D. (2010). Synchronization of abstract and concrete syntax in domain-specific modeling languages: By mapping models and live transformations. *Software and Systems Modeling*, *9*(4), 453-471.

Recker, J. (2005a). Conceptual model evaluation: Towards more paradigmatic rigor. In *Proceedings of the 10th International Workshop on Exploring Modeling Methods for Systems Analysis and Design* (pp. 183-194).

Recker, J. (2005b). Evaluation of conceptual modeling languages: An epistemological discussion. *Proceedings of the 11th Americas Conference on Information Systems* (pp. 329-337).

Recker, J. (2012). Modeling with tools is easier, believe me: The effects of tool functionality on modeling grammar usage beliefs. *Information Systems*, *37*(3), 213-226.

Recker, J. (2015). Research on conceptual modeling: Less known knowns and more unknown unknowns, please. In M. Saeki & H. Koehler (Eds.), *Proceedings of the 11th Asia-Pacific Conference on Conceptual Modelling* (pp. 3-7). Sydney, Australia: Australian Computer Society.

Renger, M., Kolfschoten, G. L., & de Vreede, G. J. (2008). Challenges in collaborative modelling: A literature review and research agenda. *International Journal of Simulation and Process Modelling*, *4*(3/4), 248.

Rittgen, P. (2009). Collaborative modeling—a design science approach. In *Proceedings of the 42nd Hawaii International Conference on System Sciences* (pp. 1-10). Washington, DC: IEEE Computer Society.

Rosemann, M., & Green, P. (2002). Developing a meta model for the Bunge-Wand-Weber ontological constructs. *Information Systems*, *27*(2), 75-91.

Roussopoulos, N., & Karagiannis, D. (2009). Conceptual modeling: Past, present and the continuum of the future. In A. T. Borgida, V. K. Chaudhri, P. Giorgini, & E. S. Yu (Eds.), *Conceptual modeling: Foundations and applications* (pp. 1390152). Berlin: Springer.

Rumbaugh, J., Jacobson, I., & Booch, G. (2010). *The unified modeling language reference manual* (2nd ed.). London: Pearson Education.

Scheer, A.-W., Thomas, O., & Adam, O. (2005). Process modeling using event-driven process chains. In M. Dumas, W. M. P. van der Aalst, & A. H. M. ter Hofstede (Eds.), *Process-aware information systems* (pp. 119-146). Hoboken, NJ: John Wiley & Sons.

Shanks, G., Tansley, E., & Weber, R. (2003). Using ontology to validate conceptual models. *Communications of the ACM*, *46*(10), 85-89.

Smirnov, S., Reijers, H. A., Weske, M., & Nugteren, T. (2012). Business process model abstraction: A definition, catalog, and survey. *Distributed and Parallel Databases*, *30*(1), 63-99.

Smith, S. F., & Becker, M. A. (1997). An ontology for constructing scheduling systems. *In Working Notes from 1997 AAAI Spring Symposium on Ontological Engineering*, 120-129.

Stachowiak, H. (1992). Modell. In H. Seiffert & G. Radnitzky (Eds.), *Handlexikon zur Wissenschaftstheorie* (pp. 219-222). München: Deutscher Taschenbuch Verlag.

Sugumaran, V., & Storey, V. C. (2002). Ontologies for conceptual modeling: Their creation, use, and management. *Data & Knowledge Engineering*, *42*(3), 251-271.

Sweller, J. (2006). How the human cognitive systems deals with complexity. In J. Elen & R. E. Clark (Eds.), *Handling complexity in learning environments: Reseach and theory* (pp. 13-26). Bingley, UK: Elsevier Science.

Tairas, R., Mernik, M., & Gray, J. (2009). Using ontologies in the domain analysis of domain-specific languages. In M. R. V. Chaudron (Ed.), *Models in software engineering: Workshops and symposia* (pp. 332-342). Berlin: Springer.

Thalheim, B. (2010). Towards a theory of conceptual modelling. *Journal of Universal Computer Science, 16*(20), 3102-3137.

Thalheim, B. (2012). The science and art of conceptual modelling, In A. Hameurlain, J. Küng, R. Wagner, S. Liddle, K.-D. Schewe, & X. Zhou (Eds.), *Transactions on large-scale data- and knowledge-centered systems* (pp. 76-105). Berlin: Springer.

Tolvanen, J., & Kelly, S. (2010). Integrating models with domain-specific modeling languages. In *Proceedings of the 10th Workshop on Domain-specific Modeling.*

Vallecillo, A. (2010). On the combination of domain specific modeling languages. In T. Kühne, B. Selic, M.-P. Gervais, & F. Terrier (Eds.), *Proceedings of the 6th European Conference on Modelling Foundations and Applications* (pp. 305-320). Berlin, NY: Springer.

van den Bergh, J., Beliën, J., de Bruecker, P., Demeulemeester, E., & de Boeck, L. (2013). Personnel scheduling: A literature review. *European Journal of Operational Research*, *226*(3), 367-385.

van der Aalst, W. M. P. (1999). Formalization and verification of event-driven process chains. *Information and Software Technology*, *41*(10), 639-650.

van der Aalst, W. M. P. (2013). Business process management: A comprehensive survey. *ISRN Software Engineering*, 1-37.

Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User acceptance of information technology: toward a unified view. *MIS Quarterly*, *27*(3), 425-478.

Wand, Y., & Weber, R. (1990a). An ontological model of an information system. *IEEE Transactions on Software Engineering*, *16*(11), 1282-1292.

Wand, Y., & Weber, R. (1990b). Mario Bunge's ontology as a formal foundation for information systems concepts. In P. Weingartner & G. J. W. Dorn (Eds.), *Studies on Mario Bunge's Treatise* (pp. 123-150). Amsterdam: Rodopi.

Wand, Y., & Weber, R. (2002). information systems and conceptual modeling—a research agenda. *Information Systems*, *13*(4), 363-376.

Weber, R. (2003). Conceptual modelling and ontology. *Journal of Database Management*, *14*(3), 1-20.

Weske, M. (2012). *Business process management: Concepts, languages, architectures* (2nd ed.). Berlin: Springer.

Yu, E. S. (2009). Social modeling and i*. In. A. T. Borgida, V. K. Chaudhri, P. Giorgini, & E. S. Yu (Eds.), *Mylopoulos festschrift* (LNCS 5600, pp. 99-121). Berlin: Springer.

Yu, E. S., Giorgini, P., Maiden, N., & Mylopoulos, J. (2011). Social modeling for requirements engineering: An introduction. In E. S. Yu, P. Giorgini, N. Maiden, J. Mylopoulos, & S. Fickas (Eds.), *Social Modeling for requirements engineering* (pp. 3-10). Cambridge, MA: MIT Press.

About the Authors

Stefan Strohmeier is a full professor and holder of the chair for Management Information Systems at Saarland University, Saarbruecken, Germany. His research interests refer to the intersection of human resources and digital technologies – both from a technical as well as a managerial perspective. Current research projects include to the digital transformation of HRM, HR analytics and metrics, Big HR Data and Smart HRM.

Friedrich Röhrs was a research assistant at the chair of Management Information Systems at Saarland University, Saarbruecken, Germany. Currently he works at Insiders Technologies in Kaiserslautern as a software engineer, where his work focuses on information extraction and automation of business processes. His research interests include business intelligence, business process management, business process modeling and modeling notations, as well as the application of the core concepts of business process management in human resource management.