Frank, Ulrich

Research Report

MEMO Organisation Modelling Language: Requirements and core diagram types

ICB-Research Report, No. 47

Provided in Cooperation with:
University Duisburg-Essen, Institute for Computer Science and Business Information Systems (ICB)

Suggested Citation: Frank, Ulrich (2011) : MEMO Organisation Modelling Language: Requirements and core diagram types, ICB-Research Report, No. 47

This Version is available at:
http://hdl.handle.net/10419/70907

ICB
Institut für Informatik und Wirtschaftsinformatik

Ulrich Frank

MEMO Organisational Modelling Language: Requirements and Core Diagram Types

ICB-RESEARCH REPORT

UNIVERSITÄT DUISBURG ESSEN

ICB-Research Report No. 47
December 2011

Die Forschungsberichte des Instituts für Informatik und Wirtschaftsinformatik dienen der Darstellung vorläufiger Ergebnisse, die i. d. R. noch für spätere Veröffentlichungen überarbeitet werden. Die Autoren sind deshalb für kritische Hinweise dankbar.

The ICB Research Reports comprise preliminary results which will usually be revised for subsequent publications. Critical comments would be appreciated by the authors.

Author'sAddress:

Ulrich Frank

Institut für Informatik und
Wirtschaftsinformatik (ICB)
Universität Duisburg-Essen
Universitätsstr. 9
D-45141 Essen

ulrich.frank@uni-due.de

Contact:

Institut für Informatik und
Wirtschaftsinformatik (ICB)
Universität Duisburg-Essen
Universitätsstr. 9
45141 Essen

Tel.: 0201-183-4041
Fax: 0201-183-4011
Email: icb@uni-duisburg-essen.de

ISSN 1860-2770 (Print)
ISSN 1866-5101 (Online)

Abstract

An enterprise model comprises various abstractions of an enterprise that represent both information systems and the surrounding action systems. These different models are integrated in order to avoid redundant work and to contribute to a tight and consistent integration of action systems and information systems. For this purpose, the method MEMO (Multi-Perspective Enterprise Modelling) features a family of modelling languages, each of which is aimed at representing specific perspectives and aspects of an enterprise. Within the MEMO languages, the Organisation Modelling Language (MEMO OrgML) is of outstanding relevance. It allows for creating elaborate models of business process types (organisational dynamics) and of organisation structures. Therefore, it is a key instrument for analysing and (re-) designing a company's action system, i.e. its key patterns of division of labour and coordination with respect to organisational goals. This report is intended to prepare for a major revision of the MEMO OrgML. Therefore, its focus on terminological foundations and on the comprehensive analysis of requirements related to the design of organisation modelling languages. The analysis was guided by a method for developing domain-specific modelling languages. It suggests focussing on use scenarios to analyse and refine requirements. The analysis is differentiated into general requirements for DSML and general as well as specific requirements for organisation modelling languages. The results are presented in an extensive dictionary of almost 70 requirements on different levels of abstraction.

Table of Contents

Figures

Tables

Typographical Conventions

If textual elements of meta models (or the meta meta model respectively) are referred to in the standard body text, they are printed in Courier italic, e.g. *`OrganisationalUnit`*.

1 Introduction

Driven by an impressive technological progress, information technology has penetrated today's business firms to a large extent. Enterprise information systems (EIS) are of pivotal relevance for many companies' economic performance and their ability to cope with future changes. For many people, using an EIS seems like a natural part of their professional activities. However, despite the ubiquitous presence of information technology in today's business firm, designing, implementing and deploying EIS is still a challenging task. On the one hand, there is the sheer technological complexity of building, adapting and handling large software systems, which is amplified by ever changing technologies as well as a steady stream of new buzzwords and promises of silver bullets. But technology is only one side of the coin. For information systems to serve a firm's goals and to contribute to its competitiveness, they need to be aligned to the business. That does not only require taking into account how the business is done today, but also to anticipate possible future changes. In addition to that, it demands for a professional process of mutual planning and adaptation of information systems and the action systems they are embedded in. Managing such a process efficiently requires business experts to acquire a solid understanding of an information system's functionality, and IT professionals to get a solid understanding of the way, the business is done. However, in many companies, overcoming the notorious "cultural chasm" (Keen 1991) between business experts and IT professionals is a major challenge. This dissatisfactory situation has substantial economic implications. On the one hand, information systems are often not satisfactory. They lack integration and adaptability. As a consequence, redundancy of data, functions and related human work compromises integrity as well as performance. On the other hand, realising and maintaining corporate information systems are consuming time and resources to a remarkable extend making them a serious financial burden for many companies.

To cope with these challenges, there is need for reducing complexity and risk, for improving communication between stakeholders with different professional backgrounds, for accomplishing a higher level of integration and for aligning IT and business tighter. In software engineering, this challenge has been met with the introduction of conceptual models of information systems. Conceptual models of this kind are aimed at abstracting from technical details of underlying implementation languages or platforms. This is for two reasons: On the one hand, these details are often subject of rapid technological change. Therefore a model should not depend on them. On the other hand, abstracting from technological details is a prerequisite for making conceptual models accessible to those who are not IT experts. Conceptual models are focussing on reconstructing the domain of interest using formal or semi-formal specifications of domain-specific concepts. In order to cope with the complexity of software systems, conceptual models are usually restricted to a specific basic abstraction: *static* abstractions such as data models, *functional* abstractions such as data flow diagrams or *dynamic* abstractions such as state diagrams. Object models combine static and functional abstractions. Conceptual models of this kind are restricted to guiding the construction of software systems that comply with certain

requirements. Hence, they do not support the mutual alignment of information system and organisational design.

Enterprise models are intended to fill this gap. There is a plethora of different interpretations of the term around – sometimes also referred to as "enterprise architecture". In the scope of our research, we prefer the following conception:

> An **enterprise model** consists of at least one conceptual model of an information system, e. g. an object model, and at least one conceptual model of the surrounding action system, e. g. of a company's organisation or its strategy. Different from other models of enterprises as they are used e. g. in Business & Administration Sciences or in consulting practice, these models are described through explicitly specified modelling languages. Different from conceptual models in software engineering, enterprise models are not restricted to models of software systems.

To foster communication and collaboration, the different models of an enterprise model are integrated through common concepts within the corresponding modelling languages. An enterprise model serves to analyse and design information systems that are aligned with a company's organisation and strategy. At the same time, enterprise models provide various representations of a company and its information technology infrastructure, which can be regarded as a knowledge repository for various stakeholders. Languages used for enterprise modelling are often domain-specific, i.e. they include concepts characteristic for the domain that is intended to be represented.

Within enterprise models, models of business processes are of outstanding importance. This is for various reasons. Firstly, focussing on business processes emphasises a goal- and customer-oriented approach, which has shown to be a helpful orientation for improving the efficiency of traditional work patterns. Secondly, business process models provide a representation that is accessible both for business and IT experts. Last, but not least, process-orientation is a well-tried heuristic: For analysing and designing complex systems it is often the approach of choice to first focus on functions or processes.

During the last years, a number of languages for modelling business processes have emerged. However, only few of them were designed as part of a method for enterprise modelling. This report is aimed at preparing for developing a major revision of a language for organisation modelling. The language is part of Multi-Perspective Enterprise Modelling (MEMO), a method for enterprise modelling. Since it does not only allow for describing business processes, i.e. dynamic aspects of organisations, but also static aspects, i.e. the organisational structure, it is called MEMO Organisation Modelling Language (OrgML). Different from workflows, business processes are not necessarily subject of automation, since they will usually include human action. Therefore, a business process model will not be executable in many cases. Nevertheless, its semantics should be specified as precisely as possible to support automated analysis and to provide for the transformation into workflow schemata. To prepare for an elaborate language specification, an extensive requirements analysis is necessary. Soon after starting with analysing requirements for a revised version of the MEMO OrgML, it became evident that the complexity of the task demands for the use of an adequate method. This insight motivated work on a method for developing domain-specific modelling languages. The approach presented in this report follows a this method (Frank 2010). Furthermore, the growing extension of respective

requirements resulted in giving up the original plan to present both, requirements analysis and design of the language in a single report. The report starts with introducing a basic technical terminology. Then, general requirements for DSML are considered. Subsequently, requirements for organisation modelling languages are analysed in more detail. Finally, the requirements are presented in an extensive dictionary. The report will be supplemented by two further reports – one on the part of the MEMO OrgML that serves modelling organisation structures, the other one on the part intended for business process modelling.

2 Terminological Background and General Requirements

Designing a modelling language implies the analysis of the requirements it should satisfy. The requirements in turn depend on the purposes, the language is supposed to serve. In a previous research report (Frank and Laak 2003), a comprehensive requirements analysis for business process modelling languages is presented. Therefore, this report is restricted to an overview of key aspects.

2.1 Core Terms

The language serves to model organisations, i. e. business processes and corresponding organisational structures. Before we focus on specific purposes and requirements, we shall first outline our understanding of core terms. These conceptions are in part based on a glossary presented in Frank (2001). A more detailed description of terms will be introduced with the specification of the language.

The term **"organisation"** is semantically overloaded. Therefore, its different interpretations should be clearly distinguished. In an **instrumental** sense, the term **"organisation"** represents a system of more or less restrictive guidelines and rules as well as incentives and sanctions to promote/enforce them. As long as these guidelines etc. are official, i.e. they have been made explicit and supported by management, they can be regarded as the **formal organisation** - in contrast to the **informal organisation**, which is comprised of guidelines, rules etc. that are relevant for collaborative action, but that are not included in the formal organisation. It puts emphasis on power and influence of informal (not official) (opinion) leaders. Note that the main focus of this report is on modelling the formal organisation. In an **institutional** sense, the term **"organisation"** denotes a social or socio-technical system - like a business firm, a non-profit organisation, public administration etc. An (institutional) organisation has an (instrumental) organisation.

An **organisational structure**, also referred to as the static structure of an organisation, represents **organisational units** and relationships between them. An organisational unit can be composed of other organisational units. The elementary organisational unit, which cannot be decomposed any further, is called **position**. An organisational structure may also include **roles**, which define additional responsibilities orthogonal to positions. Organisational units and roles are described in terms of tasks, responsibilities, required skills etc. Relationships between them include lines of command, aggregation etc.

A **business process** is a purposeful organisational construction, which is directed towards the creation of products and/or services for internal or external customers. Executing a business process requires scarce resources. A business process is composed of subprocesses. A subprocess represents a cohesive unit of work, which is triggered by an event and results in one or more further events. The temporal order of subprocesses is subject of a more or less rigid specification, the control structure.

A **business process type** represents a class of business processes of the same kind.

A **business process model** is a purposeful linguistic abstraction of a business process type. It features usually, but not necessarily, a graphical representation.

The terms business process and workflow are sometimes used without explicit distinction. However, it makes sense to differentiate between these terms:

> A **workflow** is an abstraction of a business process, which is focussed on those parts of a business process that are subject of an automated execution. They include the flow of digital objects and documents as well as formalised execution rules. Human activities and decisions within a business process are neglected or reduced to patterns of interaction with the software systems that realise the workflow.
>
> A **workflow type** represents a class of workflows of the same kind.
>
> A **workflow model** is a purposeful linguistic abstraction of a workflow type or a corresponding business process model. I can be accompanied by a graphical representation.

Note that this conception of business process and organisational structure is emphasising a more formal perspective on organisation. This restriction reflects the scope of languages for enterprise modelling. The concepts of these languages require a precise definition. That does not mean, however, that other conceptions of organisation, which emphasise its social construction (Morgan 1986), its political dimensions or the role of subjectivity and deception (Weick 1979), are not relevant for the use and interpretation of enterprise models.

In addition to these core terms of the targeted domain, the development of a modelling language requires a specific terminology, too – including concepts such as conceptual model, modelling language, diagram, modelling method etc. For definitions of these terms, readers may refer to a further report (Frank 2011). With respect to their purpose and the semantics of their concepts, modelling languages can be differentiated into general purpose modelling languages (GPML) such as the ERM or the UML and domain-specific modelling languages (DSML).

> A **GPML** is a modelling language that is thought to be independent from a particular domain of discourse. Instead, it should be suited to cover a wide range of domains. It consists of generic modelling concepts that do not include any specific aspects of a particular domain of discourse.
>
> A **DSML** is a modelling language that is intended to be used in a certain domain of discourse. It enriches generic modelling concepts with concepts that were reconstructed from technical terms used in the respective domain of discourse. A DSML serves to create conceptual models of the domain, it is related to.

The syntax of a modelling language is formal, if it allows for precisely deciding whether a model is syntactically correct. The semantics of a modelling language is defined by interpretations, which exclude those syntactically correct models that make no sense or are wrong due to a certain notion of truth. Usually, conceptual modelling languages are semi-formal: While they include a formal (or nearly formal) specification of the syntax, the definition of the semantics is not entirely formalised.

In the following sections, we will first look at generic requirements for modelling languages. After that, we will describe requirements specific to an organisation modelling language. Finally, a number of prospective diagram types will be considered to promote the identification of further requirements.

2.2 Domain-Specific Languages: General Requirements

Independent from specific purposes, there are certain generic requirements any modelling language should fulfil. Publications on requirements for modelling languages can be differentiated into three categories. Approaches that focus on formal requirements, usually originating in Computer Science, emphasise aspects such as correctness and completeness. Typically, they do not account for characteristics of the targeted domain or for specific use cases. Approaches that stress pragmatic aspects focus the question how people actually experience the use of a modelling language. For this purpose, they conduct empirical studies. Approaches that make use of ontologies are aimed at developing a foundation of modelling languages by referring to philosophical ontologies (for a more comprehensive overview Frank 2006). So far, there seem to be no empirical studies that analyse requirements for organisation modelling. Also, for feasibility reasons, an empirical study will usually be no option. In addition to that, they face severe epistemological problems. Nevertheless, it can certainly be helpful to conduct further empirical studies, because so far we know only little about how people perceive and use modelling languages. However, one should not expect empirical studies to deliver objective requirements. This is mainly for three reasons: the dual nature of language as an expression and prerequisite of human thought, the attitudes of those who create and use modelling languages, and the various tradeoffs that have to be dealt with during an evaluation process. Although we are able to reflect upon language, for instance by distinguishing between object and meta level language, our ability to speak and understand a language, is commonly regarded as a competence that we cannot entirely comprehend (Lorenz 1996, p. 49). Therefore any research that either aims at analysing a language and its use or at inventing new "language games" (i.e. artificial languages and actions built upon them), has to face a subtle challenge: Every researcher is trapped in a network of language, patterns of thought and action he cannot completely transcend - leading to a paradox that can hardly be resolved: Understanding a language is not possible without using a language. At the same time, any language we use for this purpose will bias our perception and judgement.

In order to develop a theoretical foundation of conceptual modelling, Weber suggests referring to an ontology introduced by the philosopher Bunge (Weber 1997). Based on Bunge's work, he proposes "ontological completeness" as a core requirement, modelling languages should satisfy. As a consequence, a modelling language should offer concepts to express ontological terms, such as "things", "properties of things", "types", "states", "laws" (comparable to constraints), "lawful states" (comparable to invariants), events. While Weber's approach was adopted by others (e.g. Opdahl and Henderson-Sellers 1999, Fettke and Loos 2003), it suffers from a severe misconception: Usually, a modelling language is focusing on a particular abstraction, e.g. static, functional or dynamic. Hence, it is not ontological complete *on purpose* – and for a good reason. Nevertheless, philosophical ontologies like the one suggested by Bunge (1977) are relevant for our purpose. If we assume that such ontologies provide reconstructions of basic linguistic concepts that are characteristic for human perception and conceptualisation, these concepts should be accounted for in the design of a modelling language. That does not mean, however, that a modelling language needs to be ontologically complete. The level of completeness would depend on its purpose.

In this report, we apply and refine the framework presented in Frank and Laak (2003). It is intended to account for various criteria that are relevant for domain specific modelling languages. For analytical purposes, these criteria can be differentiated into formal, user-oriented and application-oriented requirements. Note that these are not orthogonal dimensions. These generic requirements need to be further refined for a specific language.

Formal requirements are of special relevance, if formal procedures (that allow for automation) are required for analysing or transforming models.

User-oriented requirements refer to the prospective modellers' perception of language concepts and their visualisation.

Application-oriented requirements are related to the intended modelling domains and generic modelling purposes.

2.2.1 Formal Requirements

While the formalisation of a modelling language is not an end in itself, it can be very useful with respect to analysing and transforming models. In computer science, it is common to demand for a *correct* and *complete* specification of modelling languages. A specification is correct and complete, if it allows for the clear identification of all models, which include syntactic or semantic errors. At the same time, a correct and complete specification allows for generating the set of all models that are syntactically and semantically correct. In other words, a correct and complete specification refers to the formalisation of both, the syntax and the semantics of a language. Formalisation requires the use of a formal (meta) language. For our purpose, the quest for formalising a modelling language seems too rigid. While it makes sense to formalise the syntax in order to foster formal analysis and the construction of modelling tools, formalising the semantics is hardly feasible. That is for two reasons. Firstly, the domain of interest may include concepts that balk at formalisation, e. g. "competence" or "customer satisfaction". In this case, the only option is "pseudo" formalisation. It occurs in two variants, which may be combined. A concept can be condensed to a formal concept with more or less meaning. Describing a resource type's value by referring to an ordinal scale would be an example of this kind of formalisation. The second variant features representations with no formal semantics that depend on human interpretations, e. g. a string that represents a natural language term or statement. In both cases, one can hardly speak of formal semantics, since it is not possible to formally decide whether a particular description is appropriate or not. Nevertheless, they can be helpful for documenting and analysing organisational models. Secondly, analysing models of action systems will often require taking into account an empirical context. In these cases, the (informal) semantics of a model depends on topics that are not modelled. Hence, they cannot be formalised within a model's semantics.

Against this background, we shall relax the demand for formalisation:

> **Requirement F1**: The specification of a modelling language should include a precise and complete specification of its syntax. In an ideal case, this will be a formal specification. In any case, the syntax specification should allow a human to clearly decide whether a specific model is syntactically correct or not.

Requirement F2: In order to support the implementation of corresponding modelling tools, the specification language should correspond to languages used for software design.

Within the specification of a language, there is no clear distinction between syntax and semantics. Syntactical rules can be used to exclude models that make no sense. The semantics can be specified by formal and informal rules.

Requirement F3: The rules defining the semantics of a modelling language should be suited to clearly guide prospective users with the construction of appropriate models and their adequate interpretation. These rules should be formalised, if this does not compromise the intended meaning.

The convenient and safe development and maintenance of models recommends concepts that allow for a high level of abstraction. A high level of abstraction fosters reuse (e.g. through generalisation/specialisation) and integrity.

Requirement F4: The modelling language should feature concepts that foster a high level of abstraction to support model integrity and reuse.

2.2.2 User-Oriented Requirements

The prospective users of the OrgML include domain experts, systems analysts, and software developers. While it is likely that these groups will emphasise different specific requirements, there are three generic requirements that make sense for all users: simplicity, comprehensibility and convenience of use. Note that these requirements are not necessarily compatible.

The simpler a language, the more likely it is that users are not overcharged. As a consequence, one can expect that a simple language will foster the construction of error-free models. While there is no clear definition of simplicity, we can assume that a language is the more simple, the less the number of its concepts and the number of the rules to define its syntax and semantics.

The comprehensibility of a language may be fostered by its simplicity. But it seems that it is more important that the concepts it provides correspond to the terminology prospective users are familiar with.

Requirement U1: The concepts of a modelling language should correspond to concepts prospective users are familiar with. That recommends reconstructing existing terminology. Furthermore, it recommends using graphical symbols that are suited to illustrate the corresponding concepts' meaning.

Note, however, that satisfying this requirement may be aggravated by the diversity of domains – and corresponding terminologies – a modelling language is supposed to cover. Convenience of use refers to the effort that is required to build a model. This requirement is contrasting the demand for simplicity. If a language provides domain specific concepts, it supports those users' productivity who develop models for this domain, since they do not need to construct these concepts on their own. However, the more specialised concepts a modelling language includes, the higher will be the effort to learn it.

Requirement U2: To overcome the conflict between convenience of use and simplicity, a language should provide a core of basic concepts that are sufficient for creating simple models.

Usually, different groups of users have different demands for the level of abstraction and detail provided by a model. Some will be bothered by too much detail, while others focus problems that require a higher degree of detail.

> **Requirement U3**: The modelling language should allow for building models on various levels of detail and abstraction. A modeller should not be forced to specify detail he does not need.

2.2.3 Application-Oriented Requirements

Incorporating domain specific concepts into a language is only one option to foster productivity. A further option is to provide reference models for reuse. Only if the semantics of a concept is invariant across the entire class of intended models, it is suited to be incorporated into the language. Otherwise, it would limit the usability of the language. In addition to that, to not compromise the quest for simplicity too much, only those concepts should be included in the language that are needed regularly. Including a concept in the language has one major advantage over representing it in an accompanying reference model: Its adequate use can be enforced through the language specification.

> **Requirement A1**: A modelling language should provide domain specific concepts as long as they are regularly used and their semantics is invariant within the scope of the language's application.

The concepts a modelling language should include depend on the purpose the corresponding models should fulfil, hence, on its intended application. In general, the concepts of a language should be *adequate* with respect to the set of intended applications. On the one hand, that implies that a language should not be overloaded with concepts that are not required. On the other hand, it means that the language concepts should allow for descriptions on a level of detail that is implied by the application purposes. This recommends carefully analysing possible applications. However, one cannot show that a certain set of applications is sufficient for all times. Therefore, a language should allow for application-specific extensions.

> **Requirement A2**: The concepts of a language should allow for modelling at a level of detail that is sufficient for all foreseeable applications. To cover further possible applications, it should provide extension mechanisms.

Some authors demand for "ontological completeness" of modelling languages (e. g. Weber 1997, Opdahl and Henderson-Sellers 1999, Fettke and Loos 2003). However, such a requirement is misleading. A modelling language may emphasise a specific basic abstraction only, e. g. a data modelling language. In this case, concepts needed for further basic abstractions (and for the language to fulfil the claim for "ontological completeness") are not required. Since one cannot be sure about the set of all foreseeable applications, this requirement suggests applying a thorough analysis of possible use cases, which includes modelling domains and purposes.

Conceptual models are focussing the type level: The concepts of a model do not represent particular instances, but types. However, sometimes the elaborate design of conceptual models requires further levels of abstraction. Consider for instance the feature that every business process starts at a certain point in time and terminates at a certain point in time. While this feature applies to all instances of a

type, it is not a direct feature of the type. The type may have been created at a certain point in time. But that represents clearly a different meaning.

> **Requirement A3**: A modelling language should provide concepts that allow for clearly distinguishing different levels of abstraction within a model.

Often, conceptual models are transformed into other representations, e. g. into implementation level documents. These transformations should not be biased by ambiguity.

> **Requirement A4**: There should be a clear mapping of the language concepts to the concepts of relevant target representations. In an ideal case, all information required by the target representations can be extracted from the model. That requires that the concepts of the language allow for expressing all concepts of relevant target representations.

The generic requirements correspond to generic criteria for evaluation modelling languages. For a more comprehensive discussion of this topic see Frank (1998).

2.3 Organisation Modelling Languages: General Requirements

Developing specific requirements for a language to model business processes and organisation structures recommends thoroughly analysing the subject as well as the modelling purposes. However, for now, our analysis of specific requirements will be restricted to some main categories. Detailed requirements will be proposed only later together with the corresponding concepts of the language. Hopefully, this will contribute to the readability of the report. To give an overview of the intended applications of the language, we will first look at purposes of organisation modelling. A core purpose of modelling organisations is to support organisation analysis and design. Hence, a basic requirement would be to allow for adequate representations of organisation structure and business processes. Firstly, a representation can be regarded as adequate, if it supports the goals of organisation analysis and design. Secondly, adequacy is also related to comprehensibility and convenience. Organisation analysis and design are guided by goals such as improving efficiency – of business processes and decision making, increasing flexibility, reducing cost etc. To support this kind of analysis, there is need for concepts that allow for elaborate descriptions of organisational structures. Such concepts should allow for describing relevant types of organisational units as well as relevant associations between them. Furthermore, concepts are required that allow for describing the actions to be performed within a business process as well as the flow of control. In addition to these basic requirements, there is need for concepts that are related to these goals in the language specification. Examples would be concepts to express the time required to perform a process, the resources it consumes, lines of command between organisational units or decision competencies of positions. Furthermore, it should be possible to describe organisations on different levels of detail, since the level of detail will vary with the scope and purpose of a particular project. Both abstractions, organisational structure and business processes, are mutually dependent. For example: An organisational unit is responsible for a process, a process is operated by a particular position etc. Therefore, it should be possible that concepts to model processes can refer to concepts to model organisational structures et vice versa.

> **Requirement OM1**: The language should include concepts to describe organisational structures and business processes on various levels of detail. It should also feature concepts that support specific analysis and design tasks (corresponds to **Requirement U1**).

For a modelling language to be comprehensible and convenient to use, its concepts should correspond to the technical language, its prospective users are familiar with – in this case for the language of organisation analysis and design. While there is no long tradition in business process modelling, organisation analysis and design have been subject of research and professional education for many years. A previous report (Frank 2001) was focussed on analysing concepts to describe organisations and corresponding visualisations – in the fields of organisation studies and Information Systems. While there is hardly a unified terminology, there are several terms that are widely used. The report includes a glossary of such terms (Frank 2001, p. 52). While not all of these terms are suited for being represented in a modelling language, e. g. "organisation culture", they give an idea of the terms to be used.

> **Requirement OM2**: The modelling language should be aimed at reconstructing the technical language used in organisation analysis and design. This requirement is a specialisation of the generic **Requirement U1**.

Enterprise models allow for enriching organisation analysis and design by including representations of related topics, e. g. of a company's strategy or its information system. Through the integration with other abstractions of an enterprise, organisational models are supplemented with additional semantics, which allows for satisfying further analysis and design tasks. Business process models are at the centre of enterprise models. Figure 1 to Figure 13 illustrate their pivotal role from various perspectives. Figure 14 shows their integration with other models of an idealised enterprise model. In order to explain the goals and constraints of a business process type, its models can be associated with models of the strategy perspective, e. g. value chains or goal nets. To foster analysis of the economics of a business process type, its model can be associated with resources represented in specialised models. Resource models serve to describe relevant resource types, such as machines, devices, facilities etc. For a detailed specification of the MEMO resource modelling language see Jung (2007). IT resources are a special case of resources. Due to their specific features and the related technical terminology, it makes sense to represent them in separate models. For this purpose, MEMO features the ITML (Kirchner 2007). By allocating resource types to business process types, it is possible to analyse the actual and required contribution of resources to the services or products that are produced within business processes. A further important purpose of organisation modelling, particularly of business process modelling, is the support of information systems analysis and design. For this purpose, there is need to describe the use of information or software artefacts within a business process. It should be possible to refer to concepts within models used for software development, e. g. object models.

> **Requirement OM3**: The OrgML should include concepts of other modelling languages to support references to respective models. This requirement applies especially to other languages used for enterprise modelling.

Furthermore, an organisation model that is used as a foundation for systems design should include all concepts that are required to generate design documents – e. g. workflow schemata (see Jung 2004 for an example). This requirement is a specialisation of **Requirement A1**. With respect to generating

workflow specifications, the modelling language should cover all relevant control structures. It is not possible to prove that a certain set of control structures covers all relevant cases. However, there are comprehensive lists of control structures. A well-known proposal was made by Von der Aalst et al. (2003). It includes 20 control structures, referred to as "workflow patterns". Later, it was extended to a list of 43 control structures (Russel et al. 2006). While some of these control structures are redundant – see Börger (2007) for a reduction to eight basic "categories" – they can still serve as a reference.

> **Requirement OM4**: The OrgML should support all of those known control structures that are relevant for the purpose of the language.

It is an essential purpose of modelling business processes to support analysis and design of corresponding information systems. This includes the specification of the required data and IT resources (e.g. by referring to data/object models and models of IT resources) as well as mapping control structures to those of implementation level languages, e.g. workflow specification languages.

> **Requirement OM5**: The OrgMLshould allow for detailed references to models used for systems analysis and design. It should provide mappings to relevant workflow specification languages.

Often, the scope of organisation models will be restricted to one enterprise. However, sometimes it may be relevant to analyse cross-organisational patterns of cooperation, e. g. joint ventures, cross-organisational business processes or communication with external partners. The language should offer concepts that support this kind of analysis.

> **Requirement OM6**: The OrgML should include concepts that allow for assigning model elements to different organisations/enterprises and to express relevant patterns of communication and cooperation.

To foster ease of use, a modelling language should be kept simple. This recommends avoiding redundant concepts. However, if there are reoccurring patterns of using a language, aggregated – and hence: redundant – concepts may contribute to a more convenient use. Consider, for instance, a certain control structure within a business process that is composed of other control structures. Providing a modelling element – maybe within the graphical notation only – that represents this high level structure would not only foster productivity but would also improve a model's comprehensibility.

> **Requirement OM7**: While the specification of the language should avoid conceptual redundancy, reoccurring modelling patterns should be represented by specific model elements – if they promise to improve productivity and readability.

In some cases, the analysis of business processes can be supported by running simulations. While simulation it not a paramount purpose of the OrgML, it should nevertheless be supported to some degree. Different from mere conceptual models, simulation models require accounting for instance populations.

> **Requirement OM8**: The OrgML should include concepts that allow for specifying business process models which include information required for running simulations.

3 Analysis of Use Scenarios

A modelling language serves to create models. A comprehensive modelling language, such as the UML or the family of MEMO modelling languages, allows for creating a wide range of different abstractions. Diagram types are intended to provide the modeller with templates for abstractions that are regarded as particularly useful for certain purposes. Usually, they define a view on the language specification, which selects all language concepts that are intended for a certain diagram type. The specification of a diagram type may also include a special notation. The following examples of diagram types to be supported by the OrgML should help with elucidating the scope of purposes it is supposed to address. Exemplary questions are used to illustrate the purposes, a diagram type should serve. If these questions can be answered through an automated analysis (in case a corresponding tool is available), they are marked with an **A**. If answering them can be partially supported by an automated analysis, they are marked with a **P**. If they are subject to human interpretation/analysis only, they are marked with a **H**. Note that these assessments refer to existing diagrams, not to their creation. The specific requirements that these purposes generate for designing the selected diagram types supplement the requirements discussed above. In addition to these specific requirements, the diagram types may reveal specific challenges that relate to conflicting design goals. Note that further detailed requirements and challenges will be discussed with the specification of the language concepts.

Organisation models can be used as a semantic supplement to information systems: They enrich the data managed by an information system with a conceptual context. While this is similar to a data or object schema, it offers a higher level of semantics and should be clearly more comprehensible. If conceptual models defined through the OrgML (or other languages used within MEMO) are integrated with the software that constitutes organisational information systems, such an architecture would allow for powerful navigation operations (“drill down”, “What is the meaning of?”). Therefore, the corresponding parts of an information system that could be interfaced with a diagram type are indicated – either by a system category (e.g. “ERP”) or by the functions to be supported (e.g. “IT Management”). In an ideal case, these systems would provide a modelling environment with data that is aggregated from instance populations (e.g. variance of business process execution time).

The MEMO OrgML is not restricted to these diagram types. Also, the basic diagram types can be combined to more expressive ones. Note that the examples serve as illustrations only. They do not feature the actual graphical notation of the language; nor do they include specific details that can be expressed within a diagram type.

3.1 Organisational Chart

This diagram type corresponds to prevalent graphical representations of organisational structures. It depicts organisational units as well as associations between them. Note that – different from typical organisational charts – organisational charts expressed through the MEMO OrgML feature a precise

semantics of relationships between organisational units. It can also be applied to represent the temporal organisational structure of a project.

Purpose: Versatile representation of organisational structure to support the analysis of division of labour, decision and control mechanisms etc. as well as their redesign. At the same time: a representation/documentation of the organisational structure as part of organisational knowledge management, e.g. to support new employees or external consultants with getting an appropriate picture of an organisation. Possible questions to be addressed – depending on the level of detail features by an organisational chart:

- What is the average span of control (number of directly subordinated units)? **A**
- What is the percentage of positions that require at least a bachelor degree? **A**
- What is the percentage of positions that constitute the organisation's core competence? **A**
- Which organisational units suffer from poorly qualified employees? **A**
- Are there conflicting responsibilities of organisational units? **P**
- Are there similar positions that could be merged? **P**
- Does is make sense to reduce or widen the span of control? **H**
- What kind of impact would this have on the qualification required for employees? **H**
- What are the effects of organisation culture on work coordination? **H**

Potentially integrated with: Human Resource Management, ERP, IT Management

Key concepts: organisational unit, various kinds of relationships

Example: The example in Figure 1 illustrates an organisational chart of a strategic business unit. The edges between the symbols representing organisational units express aggregations.

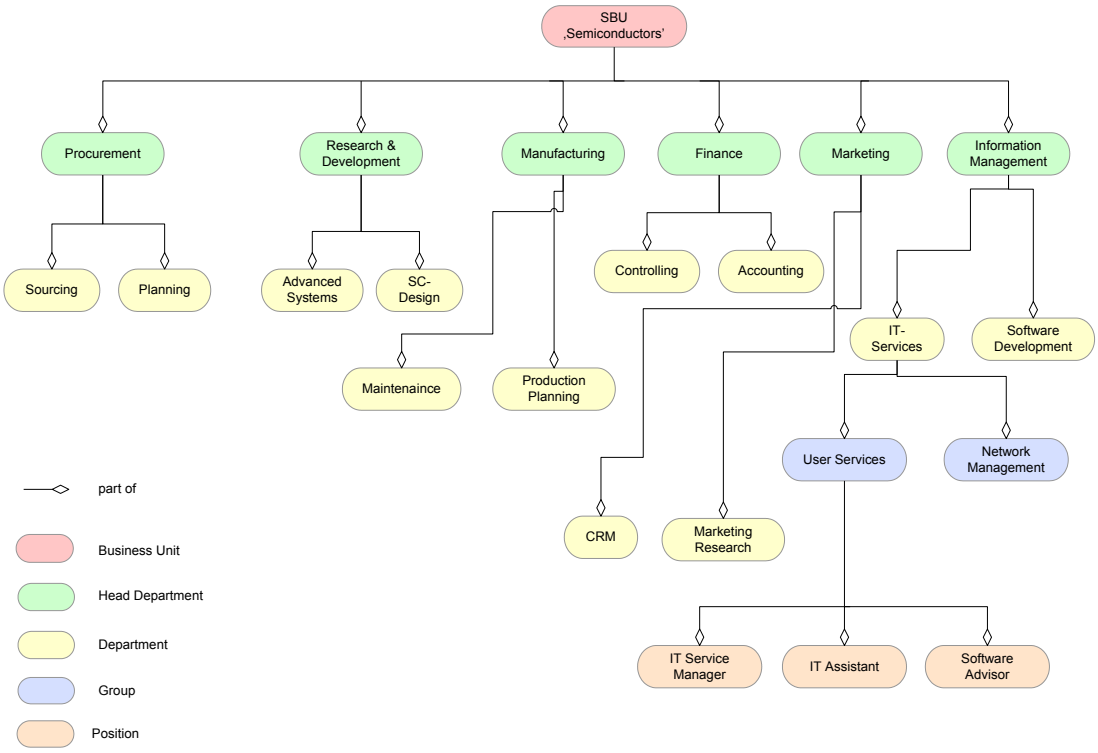

Figure 1: Example of Organisational Chart

Specific Requirements

Requirement SR1: To allow for elaborate analyses, it should be possible to describe organisational units in a differentiated way. This includes concepts to describe formal qualification, skills, tasks, responsibilities etc. as well as different kinds of associations between organisational units.

Requirement SR2: An organisational chart represents organisational units. These may be types or instances. Therefore, the OrgML should provide concepts that allow for both, defining organisational units as types and as instances.

Requirement SR3: Sometimes, certain assertions do not only apply to one organisational unit (or role or committee), but to more. It may be, for example, that various organisational units are assigned to a certain region. To elegantly specify such commonalities, there is need for concepts that allow for expressing abstractions over a set of organisational units (or roles and committees respectively).

Requirement SR 4:With the increasing spread of cross-organisational networks, joint ventures etc., it becomes more and more important to account for modelling structures that include more than one legal institution. Therefore, the OrgML should provide concepts that allow distinguishing between different organisations.

Requirement SR5: While some organisational units will usually occur only once within a particular enterprise, others – this is typically the case for positions – can exist in multiple instances of a certain type. Firstly, there is need to allow for differentiating between multiplicity constraints (“There must not be more than one marketing department.”, “There must be one and exactly one head of the marketing department.”) and actual numbers (“The current headcount

of the marketing department is 26.”). The notation should support a clear differentiation of these two meanings.

Challenges

Challenge C1: The representation of actual headcounts – or the actual number of positions of a certain kind – faces a conflict. On the one hand, a model should feature a level of abstraction that makes it widely independent from state changes on the instance level. On the other hand, information about these numbers may be regarded as relevant for an organisational chart. In case the model is managed by a tool which is integrated with a corresponding information system, these numbers could be updated automatically whenever changes occur. Otherwise it will probably be better to use numbers that are marked as estimations.

3.2 Extending Organisational Charts with Roles and Committees

In addition to organisational units, responsibilities within organisations can be defined through roles and committees. A role can be assigned to one particular position or person, e. g. “database administrator”, “quality assurance officer”. It can also serve as an abstraction that covers a range of people, e. g. “user”. A committee is an assembly that prepares or makes decisions, e. g. a financial audit committee. Usually, membership is restricted to certain positions or roles. A committee can be part of another committee (“subcommittee”). If roles and/or committees are related to organisational units, role models can be integrated with organisational charts.

Purpose: Allow for elaborate representations of roles and committees that support the analysis of the corresponding responsibilities. Possible questions:

- What is the percentage of roles that require at master degree? **A**
- Are there conflicting responsibilities of roles or committees? **P**
- Are there conflicting responsibilities of roles and positions? **P**
- Are there similar roles that could be merged? **P**
- Does it make sense to introduce further roles, e.g. to promote certain topics? **H**

Key concepts: role, committee, various kinds of relationships

Potentially integrated with: Human Resource Management, ERP, IT Management

Example: The example in Figure2 illustrates a role model that is integrated with an organisational chart.

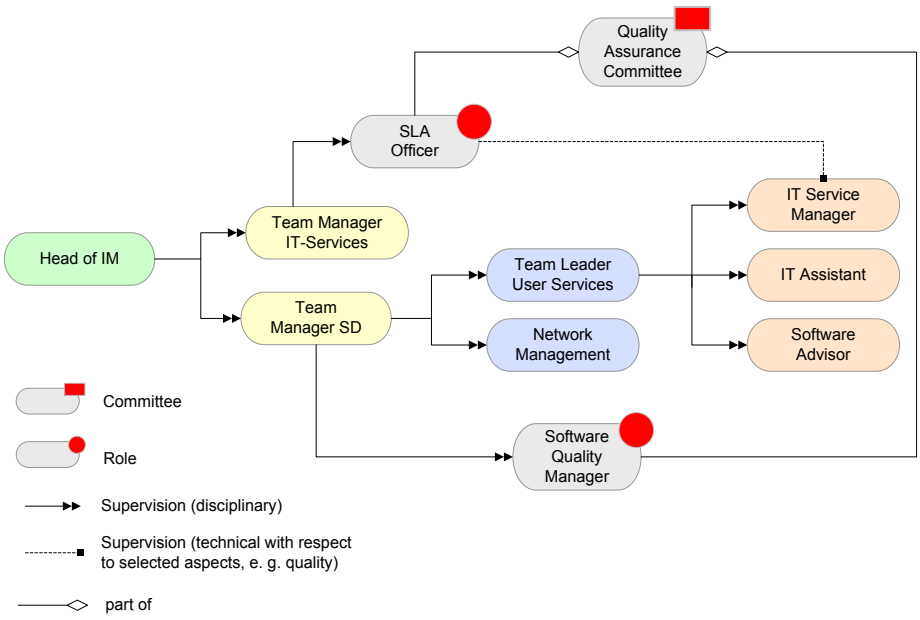

Figure2: Exemplary of organisational chart extended with roles and committees

Specific Requirements

Requirement SR6: Assigning employees to roles can be restricted to certain constraints, e.g. to the position an employee fills or to other roles he fills or must not fill. They may also be related to specific features of an employee, e.g. his skills. The OrgML should provide concepts that allow for expressing such constraints conveniently. It is a specific challenge to account for features of employees, since they are not within the direct scope of the language.

Requirement SR7: There may be rules, too, that define the preconditions for joining a committee – as well as the conditions that apply to terminating a membership. They may be related to roles, organisational units or other aspects. There should be concepts that allow for expressing these rules on an appropriate level of detail (in some cases the specific complexity of corresponding regulations would exceed the scope of an organisational model).

Challenges

Challenge C2: Often there are rules that define who is eligible to fill a role or to participate in a committee. While formalising these conditions would contribute to model integrity, it would increase the language's complexity. Possible solution: Concepts are provided that allow for defining simple constraints – e.g. that only employees that hold certain positions are eligible. With growing knowledge about relevant types of conditions, this basic set of constraints can be extended.

3.3 Organisation Interaction Diagram

Organisational charts focus on static relationships between organisational units. They are not sufficient to capture relevant patterns of interaction and cooperation within an enterprise. An organisation interaction diagram serves to depict interaction relationships, e. g. communication, counseling, cooperation.

Purpose: Allow for elaborate representations of interactions between organisational units (as well as roles and committees) in order to analyse and support communication and cooperation. Interactions include communication, potentially differentiated by frequency, intensity, subject, used media, quality etc.; cooperation, potentially differentiated by efficiency, satisfaction etc. Possible questions:

- How is the interaction frequency between organisational units/roles? **A**
- What is the share of interactions that take place within business processes? **A**
- How do participants perceive the interaction quality? **P**
- Are there options to improve the use of interaction media? **P**
- Is there need to build incentives for communicating more efficiently? **H**

Key concepts: organisational unit, roles, committee, subject, media, frequency, various types of interaction relationships

Example: The example in Figure 3 illustrates an interaction diagram that is focused on communication relationships. The thickness of the connecting edges indicates the frequency of communication.

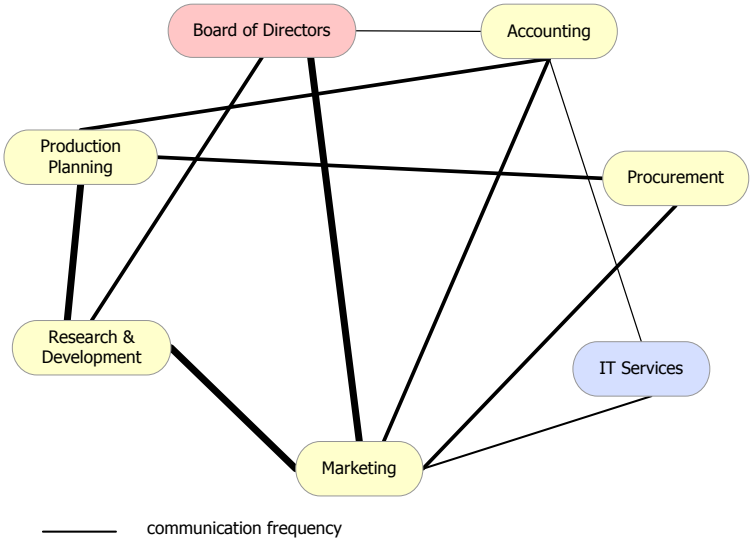

Figure 3: Example of simple organisation interaction diagram

Specific Requirements

Requirement SR8: An interaction diagram should allow for representing also other types of interaction. For example, interactions could be enriched by referring to occasions, resources, tasks, subjects, communication media etc. Furthermore, the OrgML should provide concepts that allow for representing cross-organisational interactions. To adequately represent these various kinds of interactions, it will be required to make use of different kinds of diagrams/tables.

3.4 Business Process Decomposition Diagram

Composition/decomposition is a well known concept to deal with complexity. Hierarchies that show the decomposition of functions are a common representation in functional analysis within software

engineering. Business process decomposition diagrams correspond to this abstraction. They show how business processes are composed of other business process. Note that they do not represent any flow of control.

Purpose: Analysis of processes, business processes are composed of. In addition to traditional decomposition diagrams, this includes combining decompositions of various business process types into one diagram. Questions to be addressed include:

- Are there similar or redundant processes? **P**
- Are there processes that could be outsourced? **A**
- Which processes suffer from poor performance? **P**
- How is the current performance of the represented processes? **P**
- Which processes could be further automated? **H**

Key concepts: business process, aggregation relationship

Potentially integrated with: ERP, IT Management

Example: The example in Figure 4 illustrates the composition of a sales process, where processes that are candidates for potential outsourcing, processes that suffer from poor performance as well as processes that allow for further automation are marked.

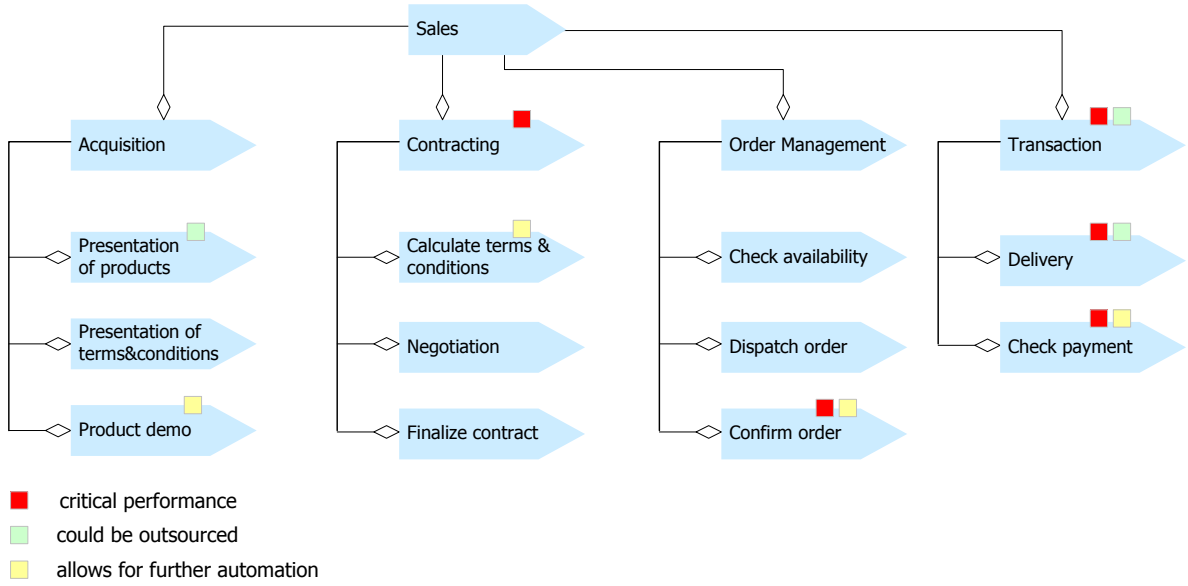

Figure 4: Example of process decomposition diagram

Specific Requirements

Requirement SR9: It should be possible to supplement the set of predefined analysis features ("critical performance" etc.) with additional user defined features.

Requirement SR10: It should be possible to assign every process within a decomposition diagram to one or more business process types, it is part of.

RequirementSR11: There should be specific graphical notations for business process types and decomposable processes to distinguish them from other processes.

Requirement SR12: There should be concepts that allow for describing features required for various kinds of analysis to be performed on decomposition diagrams – e.g. those indicated by the questions outlined above. This includes assigning the resources that are required by every process in order to analyse potential conflicts.

Challenges

Challenge C3: Adding formal semantics to analysis features would contribute to model consistency. To give an obvious example: The two features "critical performance" and "outstanding performance" must not be assigned together. However, this would add to the complexity of the language, especially with respect to the specification of user-defined features. Possible solution: Basal semantic constraints could be built into the predefined features and provided for the specification of user defined features.

3.5 Business Process Diagram

This diagram type is at the core of the OrgML. It serves the detailed representation of business process types, including the flow of control and other aspects. It will often be used as an instrument to guide further analysis – following a well-tried heuristics: Firstly, the focus is on clarifying how processes are or should be performed. In a second step, the focus shifts to the context (organisational units, resources etc.) that is required to operate the process efficiently.

Purpose: Elaborate representation of business process types to foster various kinds of process analysis and (re-) design. Also: foundation for transformations into representations used for software development, such as workflow schemata. A plethora of questions can be addressed, many of which will include concepts within associated diagrams, e.g.:

- What is the minimum and maximum execution time of process instances? **A**
- What is the variance of process execution times? **P**
- Are the skill profiles of the participating positions and roles appropriate? **P**
- What information is required for every process? **A**
- Are there any media clashes? **A**
- What application systems are used within the process? **A**
- How could the process be changed in order to reduce costs? **H**
- How could the process be reorganised in order to accomplish a higher level of customer satisfaction? **H**

Key concepts: process, event

Potentially integrated with: ERP, IT Management, WFMS

Example: Figure 5 represents an example business process model.

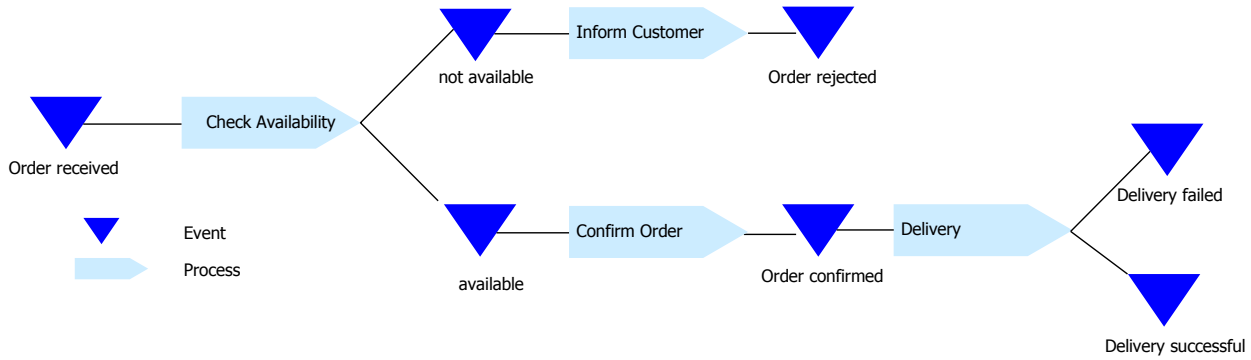

Figure 5: Example of business process diagram

Specific Requirements

Business process models serve as a key instrument to guide a plethora of analyses. Therefore, the requirements that are related to corresponding concepts of the OrgML and their visualisation are manifold. The following specific requirements supplement those described already in section2.3. They refer to the control structure as well as to the allocation of information and the communication that is required during the execution of a business process.

Requirement SR13: There should be concepts that allow for assigning probabilities to alternative paths of executing a business process. This requirement corresponds to OM6, since it is a prerequisite for running simulations.

Requirement SR14: Many business process types are characterised through a default flow of control. In certain, rare constellations, alternative flows of control need to be activated. However, modelling all possible constellations can result in all too complex representations that are difficult to understand and that distract from the essential flow of control. Therefore, the OrgML should provide the concept of an *exception* which is used to model unusual flows of control only. This requirement corresponds to **F3**, because an exception is an abstraction. It also corresponds to **Requirement OM7**, because it allows for representing reoccurring modelling patterns in a more readable fashion.

Requirement SR15: Business processes are usually information intensive processes. Hence, for analysing and improving their efficiency, the flow of information is of pivotal relevance. There are numerous ways how information can be transmitted through a business process. Therefore, the OrgML should provide concepts that allow for the differentiated description of the information flow.

Requirement SR16: Business processes are pivotal for an organisation's competitiveness. Therefore it is important to evaluate and – if necessary – improve their performance. This requires concepts that allow for describing the performance of a process – e.g. by comparing its actual performance against a reference performance.

Requirement SR17: Often, it will be important to distinguish different kinds of processes, e.g. a process type that is automated from one that is only partially automated. The modelling language should provide relevant types of processes together with a self-explanatory notation. Furthermore, it should allow for defining further process types (corresponds to **Requirement A2**).

Requirement SR18: It should be possible to express process invariants (similar to class invariants known from object-oriented modelling): If, e.g., a specific resource type is required for all processes of a business process, this could be expressed through an invariant; thus contributing to modelling convenience and model integrity (corresponds to requirements **Requirement F1**and**Requirement OM7**).

Requirement SR19: There should be concepts for representing data or objects that correspond to concepts used in systems design. Thereby, friction between business process analysis and systems design could be avoided. In an ideal case, documents used for systems design such as object models could be generated from the representation of information in business process models.

Requirement SR20: It should be possible to represent different physical media information is stored on, such as traditional media (paper, micro fiche etc.) and various kinds of digital media. Accounting for different physical media is pivotal for analysing the efficiency of information allocation within a business process

Requirement SR21: There should be concepts that allow for expressing different levels of formal semantics, e.g. bitmap, ASCII, structured document, class. The higher the level of formal semantics, the better are the chances for processing the corresponding information automatically.

Requirement SR22: It should be possible to represent the information life cycle. This includes the creation, modification and deletion of information.

Requirement SR 23: Sometimes, it is important to distinguish between different instances of information objects or resources in general – or to make sure that a certain instance is used. Therefore, it should be possible to assign identifiers.

Requirement SR24: In order to support the detection of media clashes, it should be possible to represent the transformation of information into a new representation. This requirement is related to Requirement SR20, Requirement SR21 and Requirement SR22.

Requirement SR25: It should be possible to **differentiate** between value and reference semantics of data that is transferred from one process to another. This is important with respect to the efficiency of a business process, since transferring values will usually be more costly than transferring references. It is also relevant for systems analysis and design. In general, reference semantics is preferably with respect to system integrity. However, sometimes – e.g. in offline-mode – it cannot be accomplished. Transferring copies (value semantics) requires implementing specific procedures to cater for system integrity.

Requirement SR26: The flow of information will usually include actors such as customers, suppliers or internal employees. On the one hand, it should be possible to represent the information that is requested or provided by actors. On the other hand, there should be concepts that allow for representing communication relationships between actors, e.g. cause, frequency, duration, media etc.

R**equirement SR 27**: To support analysing the economics of a business process, it should be possible to assign the resources that are required to execute a process.

Requirement SR 28: With respect to the economics of a business process the number (or the volume) of resources is important. Therefore, it should be possible to express this aspect.

3.6 Service Diagram

In recent years, the "service" metaphor has gained remarkable attention. This can be contributed to various causes. Apparently, the term "service" is overloaded. It can be applied to different subjects, e.g. to "service-orientation" as a general attitude, to the service-sector, to the services provided by an IT department, or to services provided by software. Hence, it helps with bridging the gap between different professional communities by providing a common view –at the price of ambiguity. From an academic point of view, the hype around service-orientation is sometimes annoying. That does not mean, however, to neglect "service" as a concept. It emphasises something that is essential for dealing with complexity: abstraction. If we use a service that is provided by somebody else, we do not have to care about the way the service is produced – but only about the result. Hence, we can abstract from the complexity that is related to the realisation of a service. Furthermore, the implementation (or provider) of services can be easily changed as long as the contract is satisfied – provided the contract is specified on a sufficient level of detail. The same holds true for services that are provided by software. One could argue that there is no need for an explicit concept of service, since it can be regarded as being synonymous to terms such like "task", "process" or "method"/"operation" (in the case of software systems). Nevertheless, it makes sense to provide for the explicit representation of services. This is mainly for three reasons:

- The notion of a service helps with managing complexity by emphasising abstraction. Furthermore, it puts emphasis on contracts. Thereby it contributes to stable and reliable patterns of separating concerns.
- The concept of a service is well accepted and appreciated by many. Therefore, a corresponding diagram should be accepted by many prospective users (requirement **U1**).
- A precise reconstruction of different aspects of the term – which would result in different concepts of services (e.g. those realised through business processes, through tasks, or through software) – could contribute to overcome the problems that are caused by the ambiguous use of the term.

Purpose: Support for documenting and analysing the system of services. Potential questions:

- What is the share of services that are obtained from external suppliers? **A**
- How is the average level of service satisfaction? **A**
- Are there similar services that could be gainfully combined? **P**
- Are the services defined on an appropriate level of abstraction and detail? **H**
- Where would it make sense to encapsulate software systems through software services? **H**
- Are there service contracts that need improvement? **P**
- Is there need for introducing further services? **H**

Key concepts: service, software-service, contract, various types of associations between services

Potentially integrated with: Service Management, IT Management

Example: Figure 6 represents a possible service diagram. Note that it will usually be helpful to enrich service diagrams with other representations that describe the relevant context, e.g. processes that provide a service, software systems that provide an IT service, the involved organisational units etc. Notice that the business processes that are supported by the selected service "Incident Management" could also be part of a more elaborate business process map.

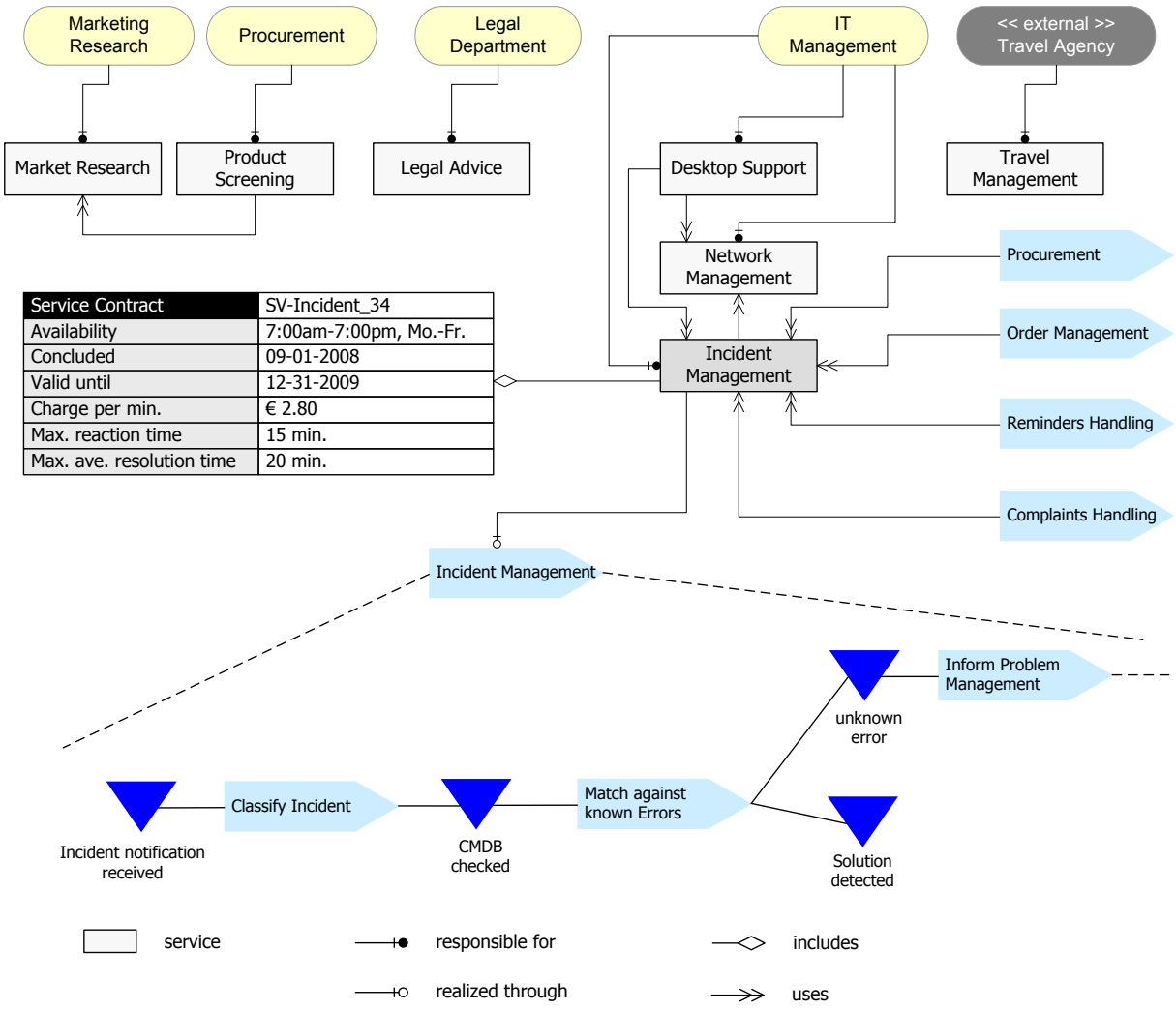

Figure 6: Example of a service diagram and selected service - associated with relevant context diagrams

Specific Requirements

Requirement SR29: It should be possible to specify different types of services.

Requirement SR30: There should be concepts that allow for defining associations between services and between services and other relevant concepts such as business processes, software systems, organisational units etc.

Requirement SR31: The OrgML should provide concepts for specifying service contracts on various levels of detail.

Challenges

Challenge C4: In order to promote reuse, i.e. modelling efficiency, and model integrity, it would be good to provide concepts that allow for expressing commonalities of a set of service types (**Requirement F4**). For this purpose, specialisation relationships would be particularly useful. However, the specification of a specialisation relationship implies a remarkable challenge: In an ideal case, specialisation does not only mean that a specialised concept includes the features of its superordinate concept. Furthermore it implies that an instance of the specialised concept can substitute an instance of the superordinate concept. A possible solution could be to relax the ideal concept of specialisation – e.g. to give up the demand for substitutability.

3.7 Decision Scenario Diagram

The more dynamic the environment of an organisation, the more important are decisions that do not merely execute formal rules. In large organisation, there will be many different kinds of decisions, which makes it a challenge to analyse the quality of decision making and how it could be improved. Firstly, this requires building categories of decisions that are alike. We call a category of similar decisions decision scenario. Secondly, decision scenarios need to be evaluated with respect to their economic relevance, e.g. their impact on an organisation's competitiveness. Thirdly, those factors that are regarded as relevant for the economics and the quality of decision making need to be considered, e.g. the required skills, information, resources as well as results provided by other decision scenarios. Decision scenarios can be associated with organisational charts or role models, with services (that provide the results of decisions), with process models (that describe the path through an organisation that a decision is supposed to take) and with resources, especially with information. Note that decision scenarios are not only relevant for an organisational perspective, but also for a strategic one. In the organisational perspective, the main focus is on how a decision scenario can be supported by an adequate organisational context.

Purpose: Analysis and evaluation of relevant decision scenarios; improving the organisational support for decision making; analysis of decision paths (participating positions), efficiency etc. Potential questions:

- What are the decision scenarios that need specific attention? **P**
- How could the time a decision takes be reduced? **P**
- Are decision makers sufficiently competent? **P**
- Is the quality of decisions a relevant issue? **P**
- Are there decision scenarios that focus conflicting goals? **A**
- How could the quality of decision be improved? **H**

Key concepts: decision scenario, skill, goal, constraint, relationships between decision scenarios

Potentially integrated with: Human Resource Management, Decision Support System

Example: Figure 7 shows a decision scenario diagram. One selected decision scenario is specified in more detail using a set of features. Also, it is associated with elements of an organisational chart and referred to the diagram types that it could make use of.

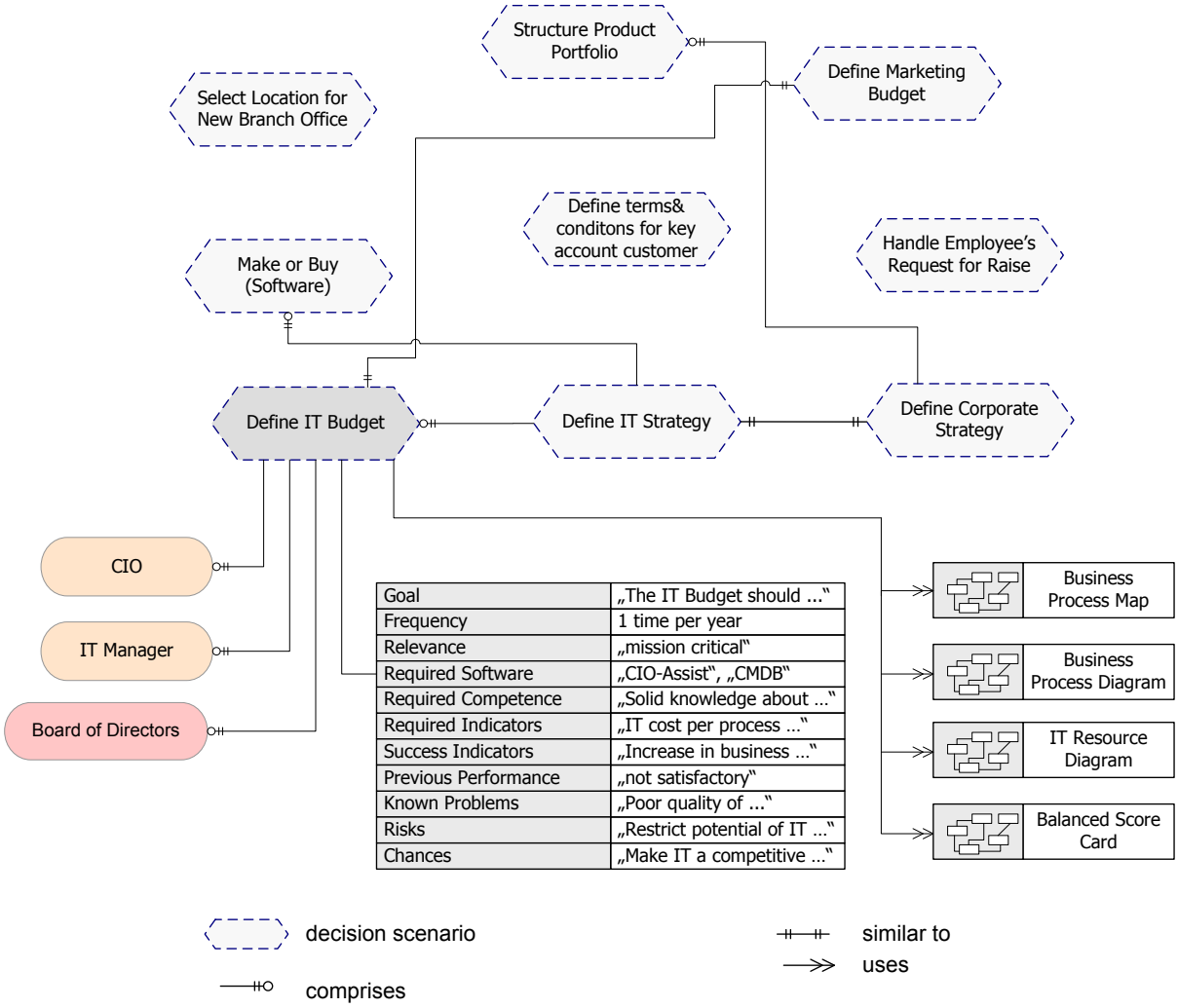

Figure 7: Example of a decision scenario diagram and exemplary description of selected decision scenario

Specific Requirements

Business process diagrams serve as a key instrument to guide a plethora of analyses. Therefore, the requirements that are related to corresponding concepts of the OrgML and their visualisation are manifold.

Requirement SR32: The OrgML should provide concepts that allow for specifying those features of a decision scenario that are needed for analysing and improving its performance. They include quality (perceived and measured), execution time, resources (required, actually available) and associations to other decision scenarios.

Requirement SR33: In order to describe the path, a decision is supposed to take within an organisation, decision scenarios require an appropriate combination with business process models.

Challenges

Challenge C5: Similar to **Challenge C4**, the specification of a specialisation relationship between decision scenarios marks a critical issue. Analysing a larger number of decision scenarios and their commonalities might help with developing an appropriate solution.

3.8 Integrating Process Diagrams with Organisational Charts or Role Models

Analysing and re-organising a business process requires accounting for its organisational context: Who is in charge of performing a process? Are the organisational resources (units, positions) assigned to a process adequate for accomplishing the process goals? Who is the (internal) customer of a process? etc.

Purpose: support the analysis of the organisational effectiveness and efficiency of a business process type. Potential questions include:

- Who is in charge of performing a process? **A**
- Are the organisational resources (units, positions) assigned to a process adequate for accomplishing the process goals? **P**
- Are there conflicting assignments of organisational units to a process? **H**
- Who is the (internal) customer of a process? **A**
- Is there need for adapting the organisation structure to improve process support? **H**

Key concepts: process, event, organisational unit, role, committee

Potentially integrated with: ERP, IT Management, WFMS

Example: Figure 8 shows a business process diagram that is associated with elements of an organisational chart.

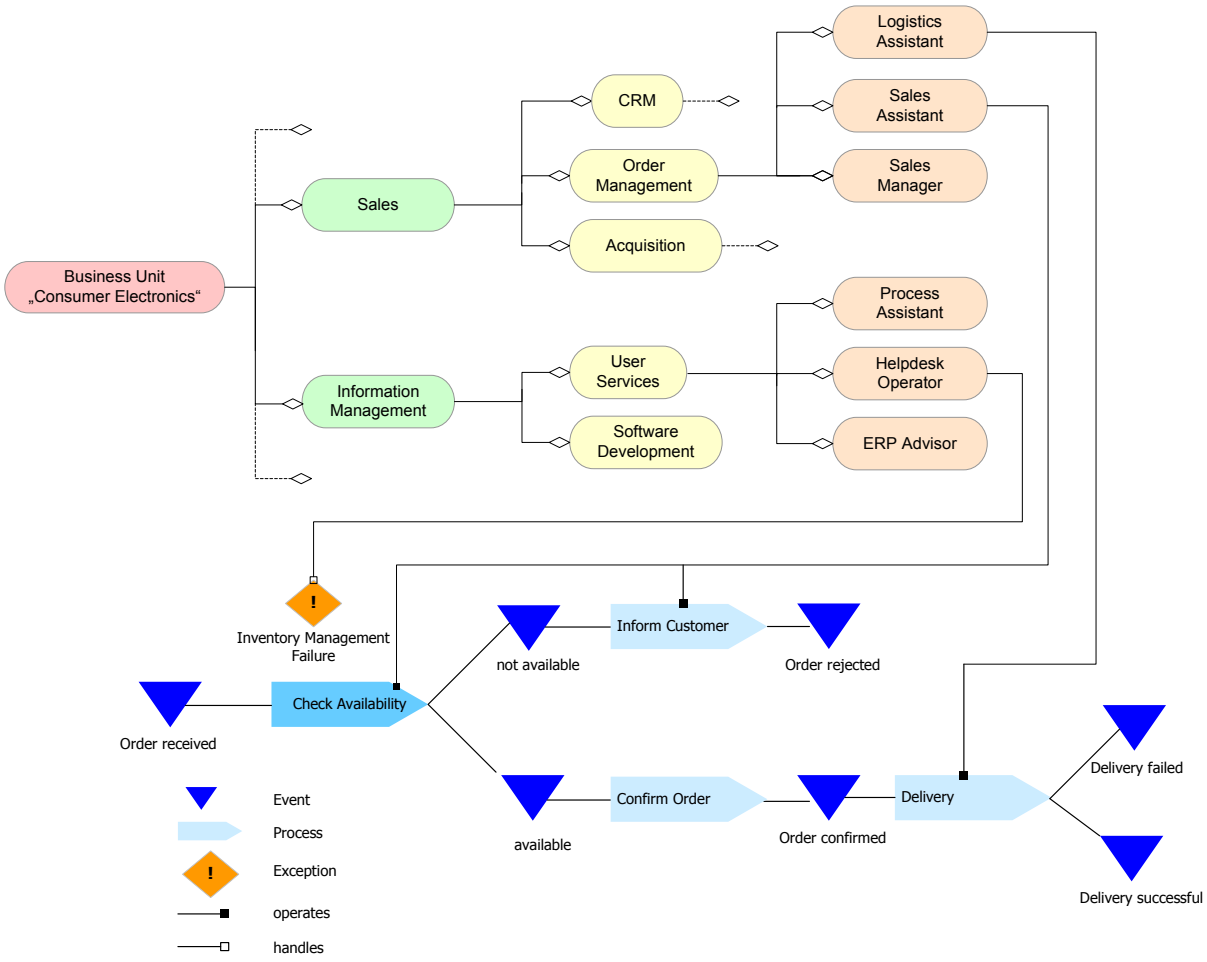

Figure 8: Example of business process diagram enhanced with elements of an organisational chart assigned to selected process

Specific Requirements

Requirement SR34: Different types of associations should allow for assigning organisational units, roles and committees to business process models, e.g. “in charge of”, “supports”, “provides technical support” etc.

Requirement SR35: It should be possible to conveniently express constraints on instances of organisational units, roles and committees assigned to processes. If, e.g., the position “Sales Assistant” is assigned to more than one process within a business process, it may be important to express that it should always be the same position instance (i.e. the same employee) who is assigned.

Requirement SR36: Referring to an organisational chart maybe regarded as helpful in some cases. In many cases, it will add to a diagrams complexity and distract from the main focus, i.e. the business process diagram. Hence, the notation should allow for assigning organisational units without representing the corresponding organisational chart. It should also allow for clearly distinguishing between different kinds of assignments.

Requirement SR37: It should be possible to express relevant constraints on the assignment of organisational units or roles/committees to processes. For example: If organisational units and

roles were assigned the tasks they are supposed to perform and all processes were decomposed into tasks, then a constraint could be applied that only those organisational units etc. can be in charge of performing a process that cover all corresponding tasks.

3.9 Business Process Association Diagram or Business Process Map

Business process diagrams serve to model one business process type only. Sometimes there is need for looking at a group of business process types or even all business process types of a company at the same time. Business process association diagrams allow for representing several business process types and the relationships that exist between them. They are sometimes referred to as "business process maps". Note that relationships between business process types may recommend referring to additional models, e.g. resource models. The business process types included in a business process map can be decomposed into decomposition diagrams, which allows for analysing commonalities of business process types on a more detailed level.

Purpose: Representation of various business process types and various types of relationships between them, such as support, competition, dependence – in order to foster analysis and (re-) design of business process systems, i.e. sets of interrelated business process types. Potential questions include:

- Are there process types that compete for the same resources? **A**
- Are there process types that target conflicting goals? **P**
- What are the most important processes? **P**
- Is there need for reducing the number of processes? **H**
- Are there any problematic inter-process dependencies? **P**
- Is there need for improving inter-process coordination? **P**

Key concepts: business process, goal, various types of relationships

Example: Figure 9 shows a simplified process association diagram. It refers to organisational units, thereby supporting the analysis of organisational conflicts.

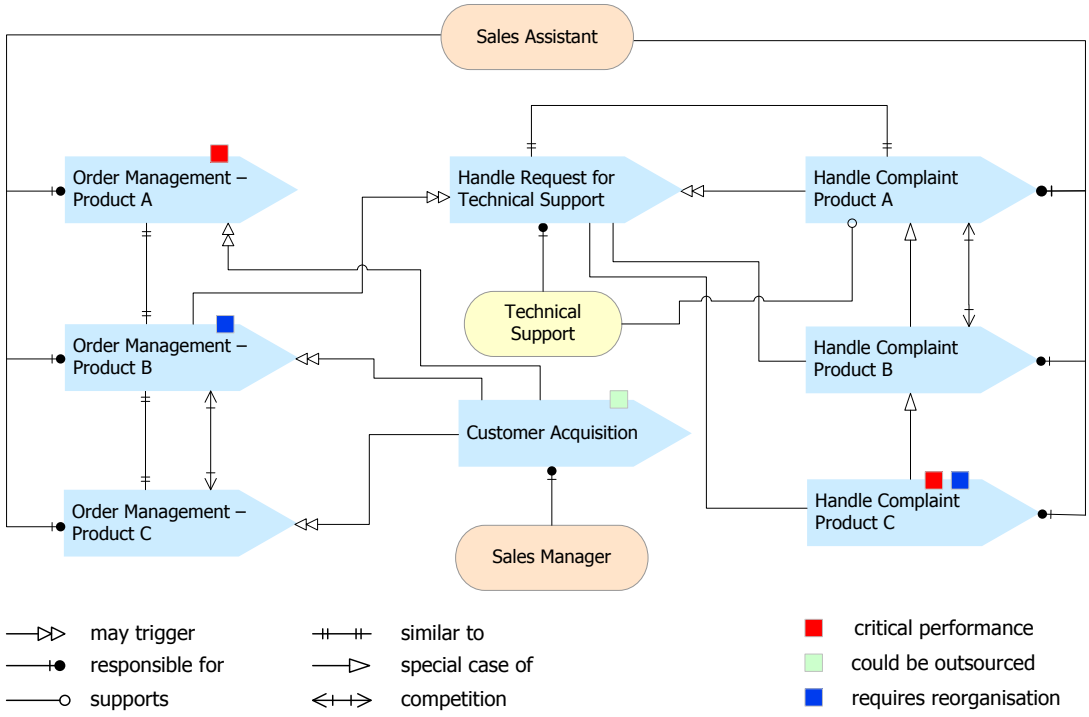

Figure 9: Illustration of business process association diagram

Specific Requirements

Requirement SR38: In addition to a set of predefined relationship types, it should be possible to define further relationship types (corresponds to **Requirement A2**).

Challenges

Challenge C6: Again, specialisation relationships mark a challenge. The example in Figure 9 gives an impression of the benefit that could be generated by specialisation: If all features that are specified for a superordinate process type (e.g.: supported by "Technical Support" for "Handle Complaint Product A") would be inherited to the specialised process types, this would not only contribute to a model's clarity, but also foster its maintainability. At the same time, it seems extremely unlikely for most cases that the quest for substitutability could be satisfied. Probably, the only option will be to aim at a relaxed concept of specialisation.

Note that there may be cases where it is useful to combine business process association diagrams with business process decomposition diagram.

3.10 Project Template or Project Diagram

The specification of business process types is based on the assumption that all corresponding process instances follow the same pattern. Projects are supposed to be different. In common definitions of a project, it is emphasised that every project is of its own kind. At first sight, this would imply that a distinction of project instance and project type would not make sense. It seems that such a conception of project is true and wrong at the same time. It is true because a project targets at least in part *terra incognita*. It is wrong, because there is usually an idea or a conception of how to organise and run a

project. It is possible to distinguish projects of different kinds. Also, there are process models – on different levels of abstraction and detail – that are supposed to guide the execution of projects of a certain kind. Hence, there is a distinction between project instances and more generic conceptions that are supposed to apply to a range of project instances. Note that we avoid using the term "project type" in order not to imply that a project instance is determined by its type. Instead, we speak of a project template or a project model that guides the specification of a particular project execution plan. While it is not intended to provide support for the management of particular project instances – this is subject of a plethora of specialised tools – it makes sense to include project templates within an enterprise model. This is for various reasons:

- It seems that project-oriented organisations are of growing importance. Also, the distinction between business processes that need to be frequently adapted to changing requirements (sometimes referred to as "ad hoc processes") and projects is blurred.
- With respect to the similarities of the execution of a project and a business process, the reuse of concepts to model business processes does not only promise to benefit from the corresponding analytical power, it should also contribute to models that appear comprehensible to those who are familiar with business process models.
- Different from project management methods and tools, the focus would be more on economic aspects of projects within an organisation and on designing an appropriate, supportive context (organisational units, roles, resources, information system).
- Finally, MEMO itself is a method, which is supposed to be deployed in projects. Hence, modelling corresponding kinds of projects would be part of the method description. Depending on the level of abstraction, there would be more or less room for individual adaptations or – in other words – for *method engineering*.

Purpose: Representation of project categories in order to analyse their organisation, the resources they require as well as the support through information systems. Furthermore, project templates serve as a blueprint for planning future project instances of the corresponding category, thereby reducing project planning cost, contributing to exploit previous experience and to common project execution standards. Related questions include:

- Is there need for providing project management with better IT support? **P**
- Are the organisational resources (units, positions) assigned to a project adequate for accomplishing the process goals? **P**
- Are control structures and information flow defined with a project execution plan adequate? **H**
- Are there conflicting assignments of organisational units to a project? **H**
- What is the appropriate project template for a particular project? **H**

Key concepts: project template, milestone, project phase, various types of relationships

Potentially integrated with: ERP, IT Management, Project Management, Knowledge Management

Example: Figure 9 shows a simplified project execution plan that expands a project category with a project map. Note that it is not to be confused with the plan of a particular project. Instead it is intend-

ed to serve as a blueprint for all projects of a certain category. Hence, its level of abstraction depends on the predictable variance of the targeted category of projects. In the example, associating a project phase with an organisational chart is done using a keyword ("BP Analyst"). Alternatively, it could also be accomplished by assigning a graphical symbol as in Figure 8. The same holds true for assigning diagram types.

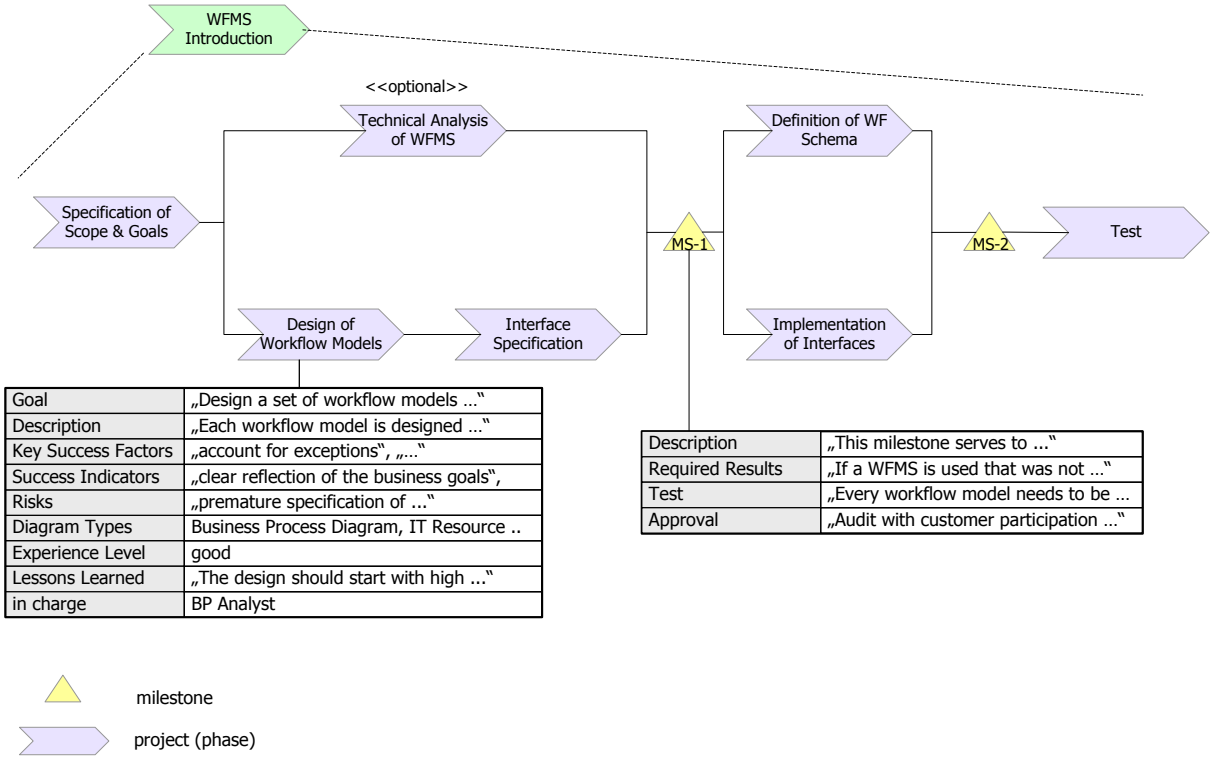

Figure 10: Example of a project template

Specific Requirements

Requirement SR39: As far as possible, concepts for business process modelling should be reused. This does not only foster the maintenance of the language, it also allows for reusing associations – e.g. to organisational units, resources, classes etc. – defined for business process diagrams.

Requirement SR40: There is need to account for concepts that are specific for project planning, e.g. to express problems, risks, challenges as well as accomplishments.

Requirement SR41: With respect to the high level of abstraction to be expected for some project execution plans, they should be supplemented with guidelines for "instantiating" project instances.

3.11 Project Association Diagram or Project Map

For organisations that generate a substantial share of their revenues through projects, planning and managing projects is a crucial issue. This includes the professional documentation of previous experience, taking advantage of synergies between projects and preventing friction caused by the competi-

tion for resources. A "ballpark view" of the project categories an organisation covers, including relevant relationships between these categories would help address these issues in a systematic way. Similar to a business process association diagram, a project association diagram (or project map) serves as a starting point for analysing and (re-) organising the project categories of a firm.

Purpose: Documentation and analysis of relationships between project categories in order to improve the allocation of resources and to support the reuse of project-specific knowledge. Related questions include:

- How relevant are the various project categories for the organisation's competitiveness?
- Are there project categories that are similar to a selected one? **A**
- What are the resources required by the various project categories? **A**
- Which conflicts exist between project categories? **A**
- Is there need for reorganising the project categories? **H**

Key concepts: project, various types of relationships

In addition to project execution plans, it can be useful to provide for project decomposition diagrams or project maps – analogous to process decomposition diagrams and process maps.

Potentially integrated with: ERP, IT Management, Project Management, Knowledge Management

Example: Figure 11 shows a project association diagram with a structured description of a project category. It also illustrates how characteristics of project categories that are pivotal for certain kinds of analysis could be visualised.

Goal	„Introduction of WFMS in ...“
Description	„To support the effective ...“
Key Success Factors	„availability of process experts“, „...“
Success Indicators	„cost reduction per instance“, „ ...“
Risks	„premature specification of ...“
Cost Drivers	„Lack of information on ...“, „...“
Experience Level	satisfactory
Lessons Learned	„The design should not be restricted ...“
Previous Projects	AG-04-3, BR-05-1, BR-06-7, ...
Staff Availability	challenge
Instantiation	„To specify an execution plan for ...“
Org. Unit	Business Information Systems
Responsible	Senior Project Manager

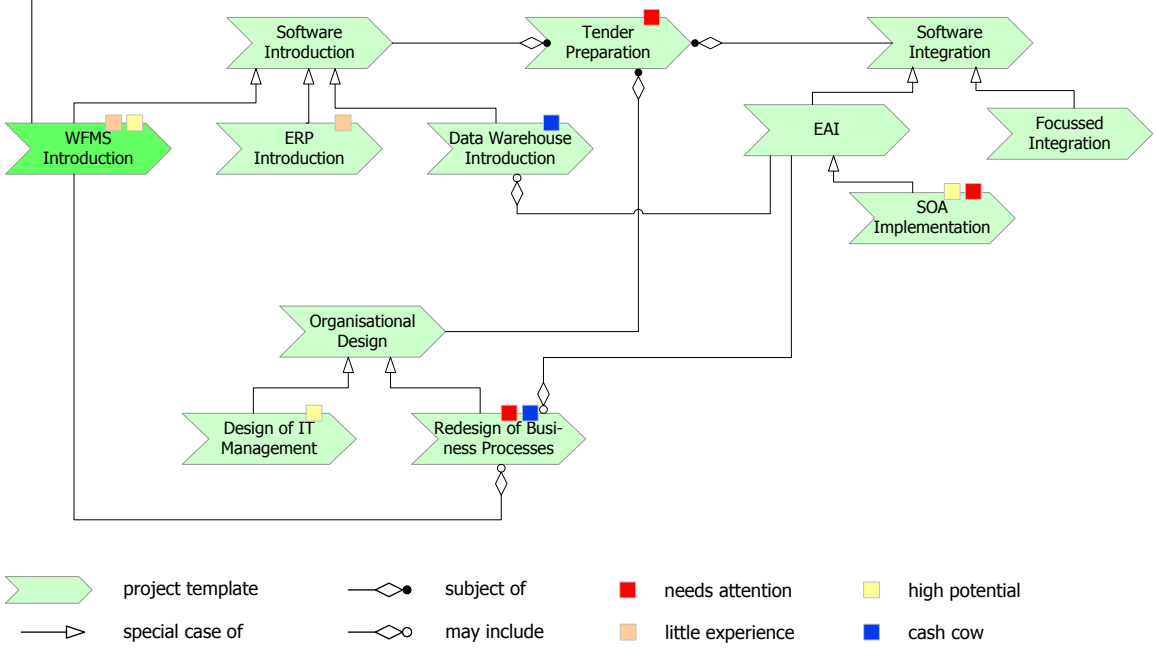

Figure 11: Example of a project association diagram

Specific Requirements

Requirement SR42: It should be possible to associate projects or project phases with concepts of other diagram types that help with analysing and (re-) designing the project categories of a firm.

Challenges

Challenge C7: As with business processes, specialisation relationships offer clear benefits – and impose challenges concerning the specification of their semantics. Probably, the substitutability constraint cannot be satisfied. Therefore, the only feasible solution might be to do with a relaxed conception of specialisation.

4 Integration with other MEMO Diagram Types

An enterprise is composed of various models specified in different languages. The actual range of models depends on the purposes, an enterprise model is supposed to serve. Hence, providing for the integration of the MEMO OrgML with other MEMO modelling languages promotes is value, because it allows for enhancing OrgML models with context – thereby supporting a wide range of analysis and design tasks. The following examples give an impression of this potential and the corresponding requirements.

4.1 Integration with Class Diagram

Business information systems are supposed to support business processes. For this purpose, they should provide information required within a business process or store information that is produced. Furthermore, they should support process management by controlling the sequence of actions within a process. Business process models support the analysis and eventually reorganisation of business processes. This includes the analysis of the information required and produced within a business process. With respect to systems design, this information can be described in more detail by referring to an object model (or class diagram respectively), which can be refined step by step according to the requirements discovered through analysing business process models. Focusing on business process models for analysing the requirements an information system should satisfy comes with clear advantages. Firstly, it is a proven heuristics to focus on processes – or functions respectively – to identify the information need. Secondly, process models provide a representation of the action system most prospective users and system developers will understand. Last, but not least, associating these models provides a foundation for generating software – including context-specific user-interfaces.

Diagram Types: Business process diagram, class diagram (MEMO-OML or UML)

Shared concepts: class, method/operation, (object-) event (lifecycle) etc.

Purpose: Support for analysis and design of process-oriented business information systems. An elaborate model of this kind serves as a foundation for generating implementation level documents, e.g. workflow schemata or context-specific user-interface specifications. Related questions include:

- What are the classes required within a business process? **A**
- Where are instances of these classes created or released within the process? **A**
- Which methods of these classes are used? **A**
- Do the classes referenced in the process represent the same or different instances? **A**
- Is there need for specifying transactions? **H**

Potentially integrated with: Software Development Environment, DBMS, WFMS

Example: The scenario depicted in Figure 12 shows a business process model that includes references to a class diagram. Note that references can be defined on different levels of detail. For instance, a

class can be associated to a (sub) process to indicate that this class is required somehow within this process. If a more precise description is necessary, a process could be associated with those methods of a class that are required within the process – as it is shown in the example. In order not to violate the demand for encapsulation, attributes are not accounted for. Furthermore, it could be referred to the instantiation of an object or to its release.

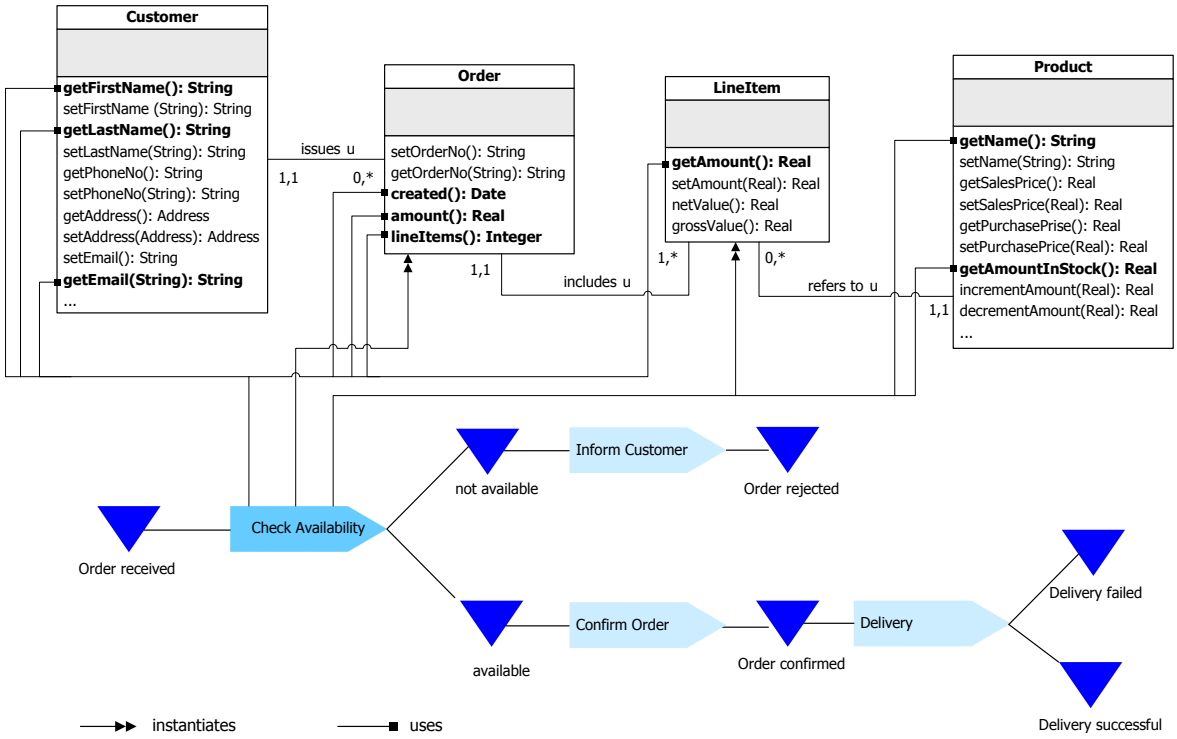

Figure 12: Integrating a business process model with an object model

Specific Requirements

Requirement SR43: Concepts within associated models that are referred to, need to be integrated into the MEMO-OrgML. This includes, for instance, concepts such as "Class" or "Operation". Furthermore, there are concepts required that allow for differentiated associations, e.g. "instantiate", "release". In order to provide for the integration both with the MEMO OML and the UML, mappings to corresponding concepts in both languages need to be defined.

Referring to a model of the organisational structure or a role model allows for identifying potential groups of users and corresponding requirements and privileges.

4.2 Integration with IT Resource Diagram

Business processes are pivotal for the effective and efficient use of IT. The integration of business process diagrams with IT resource diagrams (Kirchner 2008) is aimed at supporting various purposes that are related to this issue.

Diagram Types: Business process diagram, IT resource diagram (MEMO-ITML), possible supplemented by organisational chart

Shared concepts: software system, platform, application, service/function etc.

Purpose: There are various purposes to be addressed:

Realisation of process-oriented information systems: Often, it will be no option to design systems from scratch as outlined above by associating business process diagrams with class diagrams. Instead, existing systems need to be used, which might require to improve their integration. To analyse the demand for integrating existing application systems and to foster the realisation of integrated architectures, each business process type can be analysed with respect to the specific data and functions (or services) it requires from existing application systems. Based on the result of such an analysis, a new multi-tier architecture can be designed that encapsulates existing systems and contributes to a higher level of flexibility. Related questions include:

- What are the specific functions of application systems required within a business process? **A**
- Is there need to share data between these systems – hence: is there need for data integration? **P**
- How could the use of an application system be reconstructed by a set of services? **H**

Support of IT Management: Managing IT requires providing appropriate services to the business, e.g. care for highly available systems, fast and reliable trouble shooting, user helpdesk etc. Assigning these services – as well as the IT resources required to realise these – to business processes helps with defining their priority and with evaluating their business value. Possible Questions:

- What is the business value generated by an application system's contribution to a business process? **P**
- Which IT management services are required to support a business process? **P**
- Which IT failures can impact the performance of a business process and how can they be avoided? **P**

IT-Business-Alignment: Managing IT as an asset recommends aligning it to the business. Due to the complexity of both, the business and IT systems, there is need for focusing on relevant aspects, hence: for avoiding distraction through too much detail. Furthermore, it is required to foster communication and cooperation between various stakeholders, such as IT users, IT experts, and top management. Models of IT resources that are associated with business process models help to gain a better understanding of how IT contributes to the business. In general, linking IT resources and the business through business processes serves to foster alignment, which includes a better mutual appreciation of business people and IT professionals. Possible Questions:

- Do the functions provided by IT adequately account for the requirements of business processes? **H**
- Does the IT architecture allow for convenient and safe adaptations to changing business processes? **H**

Potentially integrated with: IT Management, Software Development Environment, WFMS

Example: The scenario depicted in Figure 13 is related to the evaluation of the business value of IT. It shows a business process model that includes references to an IT resource model and to an organisa-

tional chart. It also includes references to models of IT services, each of which can be expanded to a model of an IT service process. Similar to the previous scenario, associations can be defined on different levels of detail.

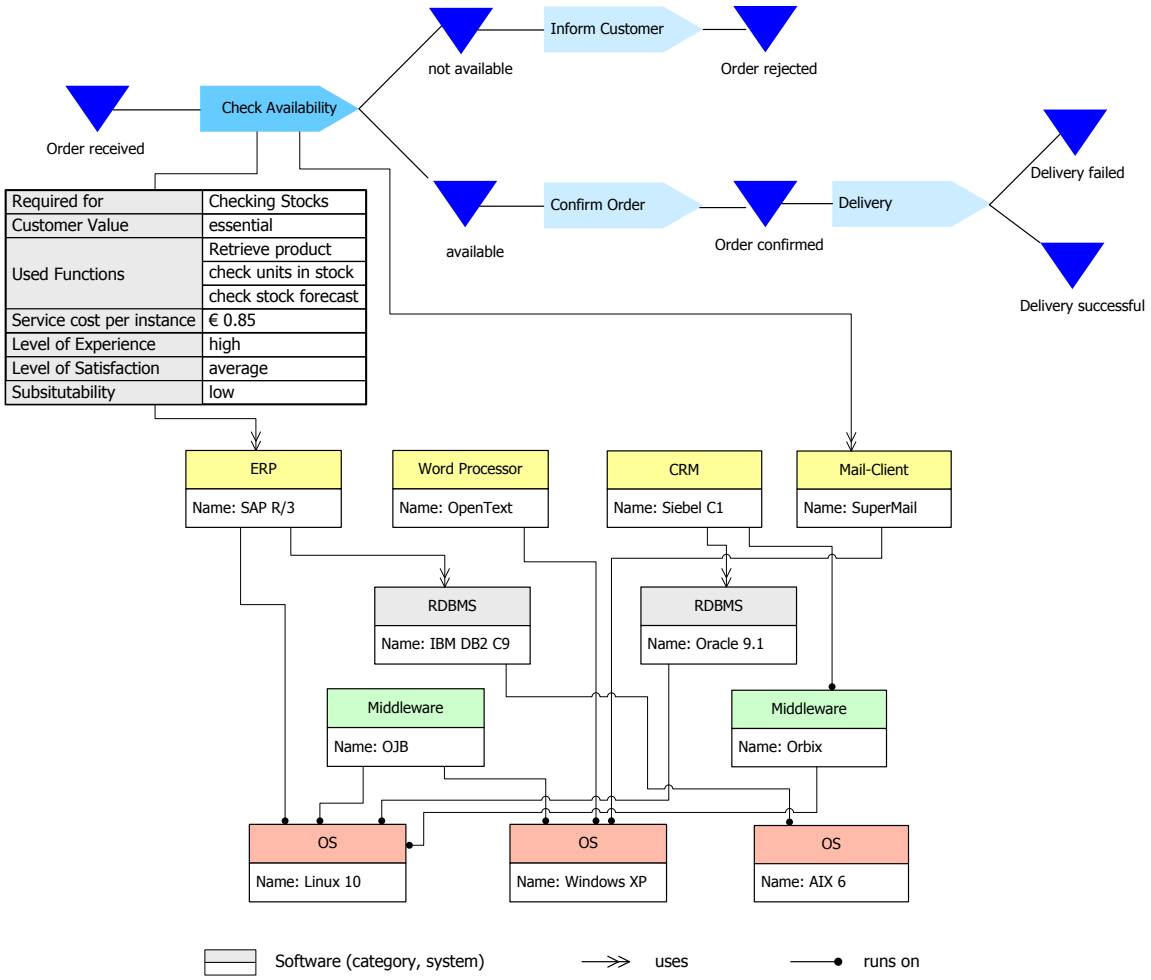

Figure 13: Integrating a business process model with an IT resource model

Specific Requirements

Requirement SR44: Concepts within the IT resource modelling language, e.g. to describe hardware, system software, networks, applications etc. need to be included in the OrgML in order to allow for defining references.

Challenges

Challenge C8: Modelling IT resources faces the challenge to define the level of abstraction and detail used for describing IT artefacts. For instance, should a particular software system be modelled as a type or as an instance, e.g. of the type "Word Processor". While this is not a direct challenge to the specification of the MEMO OrgML, it is, nevertheless, affected by it: The MEMO OrgML needs to include those concepts of the corresponding language for IT modelling that are referenced from a business process model (or an organisational chart respectively).

4.3 Integration with Extensive Enterprise Model

This final scenario serves to illustrate the entire scope of deploying OrgML models. An extensive enterprise model may include a wide range of diagram types. The basic idea is to provide a multi-perspective representation of an enterprise that supports a variety of analysis and design tasks as well as their coordination. It serves as a common reference for cooperating actors/teams. Hence, the languages used for creating enterprise models can be regarded as a foundation for tailoring a variety of modelling methods. In other words: They are a foundation for *method engineering*. In addition to that, an enterprise model does not only include descriptions of an information system's semantics (like a schema), but also descriptions of the concepts that constitute the relevant action systems. Hence, an enterprise model delivers a description of an information system as well as its pragmatics. In other words: It can serve as a foundation for self-reflexive information systems. A self-reflexive information system does not only include a description of itself but also of its context and its purpose.

Organisation models, especially business process models, play a key role within enterprise models. They provide abstractions that most stakeholders, no matter what professional background they have, are familiar with. Therefore the can serve as a common interface between other, more specialised models. In addition to that, they serve as the starting point for many types of analysis.

Diagram Types: Business process diagram, organisational chart, class diagram, IT resource diagram, resource diagram, project execution plan, manufacturing process diagram, strategy diagram, value chain diagram etc.

Purpose: An enterprise model itself serves a plethora of purposes regarding the representation of knowledge about a business, and manifold specific analysis and design tasks. In this context, organisation models are aimed at providing a starting point for analysis as well as a common reference that helps with bridging gaps between more specific perspectives. All questions listed above can be addressed by an enterprise model. In general, an enterprise model provides a good foundation for analysing how certain aspects of an organisation are linked to others or influenced by them.

Example: Figure 14 shows the composition of an enterprise model from models created in various MEMO modelling languages (the language designation is shown in inverse print in the small box on top of each box that represents a model). In most cases, it will not be required to use all possible diagram types. The associations between different models are illustrated by a few selected examples. To associate two models, it is required that the corresponding modelling languages share a concept that serves as an interface. In the example associations, this common concept is attached to the model that refers to the associated one. The concept that is used for establishing this reference – hence the concept, both associated languages need to have in common – is displayed in the box the edge representing the association starts from. For instance: To express that an operation of a certain class is required within a business process, a concept like "Operation" – which is part of an object modelling language – needs to be integrated into the OrgML. In general, it is possible for all diagram types to use more than one instance. This may apply to differentiating between representing the actual state and a target state. Also, it can help with reducing complexity to restrict the scope of a model to a part on an organi-

sation, e.g. to a department, only. Some diagram types have a narrower focus by nature, especially business process models and project templates.

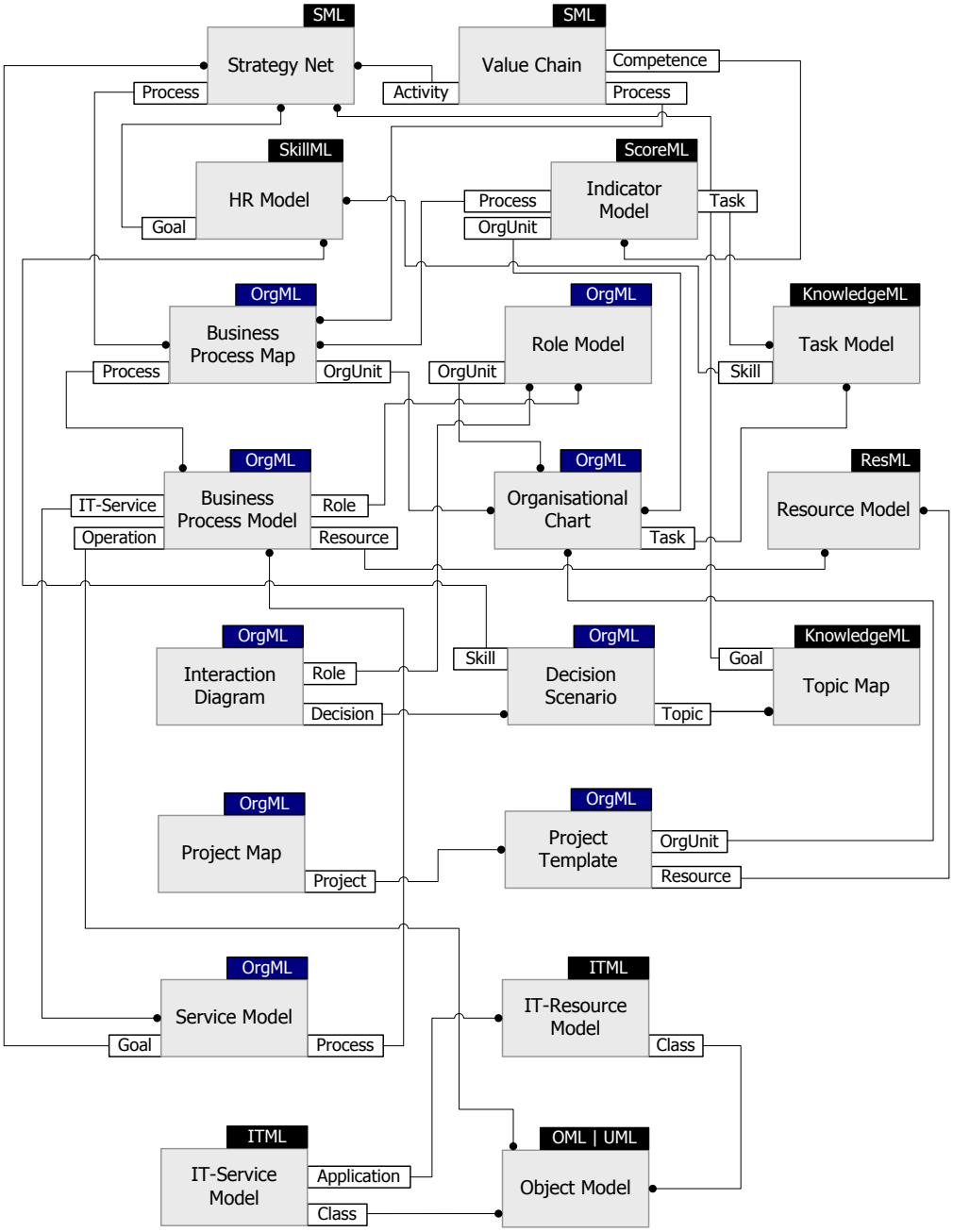

Figure 14: Possible Models of a MEMO-Enterprise Model and Exemplary Associations

Figure 14 demonstrates the outstanding importance of the OrgML within the family of MEMO modelling languages. It supports a relatively large number of diagram types. Furthermore, many analysis scenarios recommend to start with process-oriented abstractions such as process maps, business process models or project maps. The various languages have evolved in different research projects, which resulted in overlapping foci. For instance, both the ITML (IT Modelling Language, Kirchner 2008) and the ResML (Resource Modelling Language, Jung 2007) serve to model resources. Also, the various languages of KnowledgeMEMO (Schauer 2008), which serve to model – among others – tasks, goals

and human resources, focus on aspects that are in part covered by other languages already – e.g. by the OrgML.

Specific Requirements

Requirement SR45: Integrating OrgML models with other models used within an enterprise model requires including all concepts of the corresponding modelling languages that are used for inter-model associations.

Requirement SR46: To cope with modifications of existing modelling languages and the creation of further languages, it might be required to provide versatile linking mechanisms on a low semantic level.

Challenges

Challenge C9: The plethora of MEMO languages and diagram types respectively is suited to cause confusion. There are two approaches to counter this problem. Firstly, the overlaps between the different languages need to be analysed in order to reorganise and consolidate the set of languages. Secondly, prospective users should be guided with configuring an enterprise model – and corresponding process models – according to the needs of specific analysis or design scenarios. In other words: There is need to enhance MEMO with guidelines for *method engineering*.

Challenge C10: Enterprise models facilitate a plethora of different relationship types between their elements. They also allow for many kinds of analysis. With respect to the semantics of the language this implies the challenge to define a set of relationship types that satisfies a wide range of possible use cases. At the same time, there should be concepts that allow for the specification of further, customised relationships. With respect to the concrete syntax, specific symbols for marking relationship types, as those used in Figure 11, can contribute to the expressiveness of a diagram. However, with a growing number of relationship types, one will eventually run out of symbols. Also, a large number of different symbols will result in models that are hardly comprehensible anymore. It could be a possible solution to define only a few graphical symbols for those relationship types that are frequently used and to deploy self-explanatory terms for marking less relevant as well as user-defined relationship types. This is the case, too, for the visualisation of specific analysis features as it is illustrated in Figure 11.

5 Dictionary of Requirements

The first part of our analysis was aimed at requirements that should be accounted for DSML in general – at least if they are intended for being used for enterprise modelling. The following dictionary presents the complete list of all requirements.

ID	Requirement
F1	The specification of a modelling language should include a precise and complete specification of its syntax. In an ideal case, this will be a formal specification. In any case, the syntax specification should allow a human to clearly decide whether a specific model is syntactically correct or not.
F2	In order to support the implementation of corresponding modelling tools, the specification language should correspond to languages used for software design.
F3	The rules defining the semantics of a modelling language should be suited to clearly guide prospective users with the construction of appropriate models and their adequate interpretation. These rules should be formalised, if this does not compromise the intended meaning.
F4	The modelling language should feature concepts that foster a high level of abstraction to support model integrity and reuse.
U1	The concepts of a modelling language should correspond to concepts prospective users are familiar with. That recommends reconstructing existing terminology. Furthermore, it recommends using graphical symbols that are suited to illustrate the corresponding concepts' meaning.
U2	To overcome the conflict between convenience of use and simplicity, a language should provide a core of basic concepts that are sufficient for creating simple models.
U3	The modelling language should allow for building models on various levels of detail and abstraction. A modeller should not be forced to specify detail he does not need.
A1	A modelling language should provide domain specific concepts as long as they are regularly used and their semantics is invariant within the scope of the language's application.
A2	The concepts of a language should allow for modelling at a level of detail that is sufficient for all foreseeable applications. To cover further possible applications, it should provide extension mechanisms.
A3	A modelling language should provide concepts that allow for clearly distinguishing different levels of abstraction within a model.
A4	There should be a clear mapping of the language concepts to the concepts of relevant target representations. In an ideal case, all information required by the target representations can be extracted from the model. That requires that the concepts of the language allow for express-

	ing all concepts of relevant target representations.

Table 1: Generic Requirements

ID	Requirement
OM1	Requirement OM1: The language should include concepts to describe organisational structures and business processes on various levels of detail. It should also feature concepts that support specific analysis and **design** tasks **(corresponds** to **Requirement U1**).
OM2	Requirement OM2: The modelling language should be aimed at reconstructing the technical language used in organisation analysis and design. This requirement is **a specialisation** of **the** generic **Requirement U1.**
OM3	Requirement OM3: The OrgML should include concepts of other modelling languages to support references to respective models. This requirement applies especially to other languages used for enterprise modelling.
OM4	Requirement OM4: The OrgML should support all of those known control structures that are relevant for the purpose of the language.
OM5	Requirement OM5: The OrgMLshould allow for detailed references to models used for systems analysis and design. It should provide mappings to relevant workflow specification languages.
OM6	Requirement OM6: The OrgML should include concepts that allow for assigning model elements to different organisations/enterprises and to express relevant patterns of communication and cooperation.
OM7	Requirement OM7: While the specification of the language should avoid conceptual redundancy, reoccurring modelling patterns should be represented by specific model elements – if they promise to improve productivity and readability.
OM8	Requirement OM8: The OrgML should include concepts that allow for specifying business process models which include information required for running simulations.

Table 2: General Requirements for Organisation Modelling

ID	Requirement
SR1	Requirement SR1: To allow for elaborate analyses, it should be possible to describe organisational units in a differentiated way. This includes concepts to describe formal qualification, skills, tasks, responsibilities etc.

SR2	Requirement SR2: An organisational chart represents organisational units. These may be types or instances. Therefore, the OrgML should provide concepts that allow for both, defining organisational units as types and as instances.
SR3	Requirement SR3: Sometimes, certain assertions do not only apply to one organisational unit (or role or committee), but to more. It may be, for example, that various organisational units are assigned to a certain region. To elegantly specify such commonalities, there is need for concepts that allow for expressing abstractions over a set of organisational units (or roles and committees respectively).
SR4	With the increasing spread of cross-organisational networks, joint ventures etc., it becomes more and more important to account for modelling structures that include more than one legal institution. Therefore, the OrgML should provide concepts that allow to distinguish between different organisations.
SR5	Requirement SR5: While some organisational units will usually occur only once within a particular enterprise, others – this is typically the case for positions – can exist in multiple instances of a certain type. Firstly, there is need to allow for differentiating between multiplicity constraints ("There must not be more than one marketing department.", "There must be one and exactly one head of the marketing department.") and actual numbers ("The current headcount of the marketing department is 26."). The notation should support a clear differentiation of these two meanings.
SR6	Requirement SR6: Assigning employees to roles can be restricted to certain constraints, e.g. to the position an employee fills or to other roles he fills or must not fill. They may also be related to specific features of an employee, e.g. his skills. The OrgML should provide concepts that allow for expressing such constraints conveniently. It is a specific challenge to account for features of employees, since they are not within the direct scope of the language.
SR7	Requirement SR7: There may be rules, too, that define the preconditions for joining a committee – as well as the conditions that apply to terminating a membership. They may be related to roles, organisational units or other aspects. There should be concepts that allow for expressing these rules on an appropriate level of detail (in some cases the specific complexity of corresponding regulations would exceed the scope of an organisational model).
SR8	Requirement SR8: An interaction diagram should allow for representing also other types of interaction. For example, interactions could be enriched by referring to occasions, resources, tasks, subjects, communication media etc. Furthermore, the OrgML should provide concepts that allow for representing cross-organisational interactions. To adequately represent these various kinds of interactions, it will be required to make use of different kinds of diagrams/tables.
SR9	Requirement SR9: It should be possible to supplement the set of predefined analysis features ("critical performance" etc.) with additional user defined features.
SR10	Requirement SR10: It should be possible to assign every process within a decomposition diagram to one or more business process types, it is part of.
SR11	RequirementSR11: There should be specific graphical notations for business process types and decomposable processes to distinguish them from other processes.

SR12	Requirement SR12: There should be concepts that allow for describing features required for various kinds of analysis to be performed on decomposition diagrams – e.g. those indicated by the questions outlined above. This includes assigning the resources that are required by every process in order to analyse potential conflicts.
SR13	Requirement SR13: There should be concepts that allow for assigning probabilities to alternative paths of executing a business process. This requirement corresponds to OM6, since it is a prerequisite for running simulations.
SR14	Requirement SR14: Many business process types are characterised through a default flow of control. In certain, rare constellations, alternative flows of control need to be activated. However, modelling all possible constellations can result in all too complex representations that are difficult to understand and that distract from the essential flow of control. Therefore, the OrgML should provide the concept of an exception which is used to model unusual flows of control only. This requirement corresponds to F3, because an exception **is an** abstraction. It also corresponds to Requirement OM7, because it allows for representing reoccurring modelling patterns in a more readable fashion.
SR15	Requirement SR15: Business processes are usually information intensive processes. Hence, for analysing and improving their efficiency, the flow of information is of pivotal relevance. There are numerous ways how information can be transmitted through a business process. Therefore, the OrgML should provide concepts that allow for the differentiated description of the information flow.
SR16	Requirement SR16: Business processes are pivotal for an organisation's competitiveness. Therefore it is important to evaluate and – if necessary – improve their performance. This requires concepts that allow for describing the performance of a process – e.g. by comparing its actual performance against a reference performance.
SR17	Requirement SR17: Often, it will be important to distinguish different kinds of processes, e.g. a process type that is automated from one that is only partially automated. The modelling language should provide relevant types of processes together with a self-explanatory notation. Furthermore, it should allow for defining further **process** types **(corresponds** to **Requirement A2**).
SR18	Requirement SR18: It should be possible to express process invariants (similar to class invariants known from object-oriented modelling): If, e.g., a specific resource type is required for all processes of a business process, this could be expressed through an invariant; thus contributing to modelling convenience and model **integrity (**corresponds **to requirements** Requirement **F1andRequirement** OM7).
SR19	Requirement SR19: There should be concepts for representing data or objects that correspond to concepts used in systems design. Thereby, friction between business process analysis and systems design could be avoided. In an ideal case, documents used for systems design such as object models could be generated from the representation of information in business process models.
SR20	Requirement SR20: It should be possible to represent different physical media information is stored on, such as traditional media (paper, micro fiche etc.) and various kinds of digital media. Accounting for different physical media is pivotal for analysing the efficiency of information allocation within a business process.

SR21	Requirement SR21: There should be concepts that allow for expressing different levels of formal semantics, e.g. bitmap, ASCII, structured document, class. The higher the level of formal semantics, the better are the chances for processing the corresponding information automatically.
SR22	Requirement SR22: It should be possible to represent the information life cycle. This includes the creation, modification and deletion of information.
SR 23	Requirement SR 23: Sometimes, it is important to distinguish between different instances of information objects or resources in general – or to make sure that a certain instance is used. Therefore, it should be possible to assign identifiers.
SR24	Requirement SR24: In order to support the detection of media clashes, it should be possible to represent the transformation of information into a new representation. This requirement is related to Requirement SR20, Requirement SR21 and Requirement SR22.
SR25	Requirement SR25: It should be possible to differentiate between value and reference semantics of data that is transferred from one process to another. This is important with respect to the efficiency of a business process, since transferring values will usually be more costly than transferring references. It is also relevant for systems analysis and design. In general, reference semantics is preferably with respect to system integrity. However, sometimes – e.g. in offline-mode – it cannot be accomplished. Transferring copies (value semantics) requires implementing specific procedures to cater for system integrity.
SR26	Requirement SR26: The flow of information will usually include actors such as customers, suppliers or internal employees. On the one hand, it should be possible to represent the information that is requested or provided by actors. On the other hand, there should be concepts that allow for representing communication relationships between actors, e.g. cause, frequency, duration, media etc.
SR27	Requirement SR 27: To support analysing the economics of a business process, it should be possible to assign the resources that are required to execute a process.
SR28	Requirement SR 28: With respect to the economics of a business process the number (or the volume) of resources is important. Therefore, it should be possible to express this aspect.
SR29	Requirement SR29: It should be possible to specify different types of services.
SR30	Requirement SR30: There should be concepts that allow for defining associations between services and between services and other relevant concepts such as business processes, software systems, organisational units etc.
SR31	Requirement SR31: The OrgML should provide concepts for specifying service contracts on various levels of detail.
SR32	Requirement SR32: The OrgML should provide concepts that allow for specifying those features of a decision scenario that are needed for analysing and improving its performance. They include quality (perceived and measured), execution time, resources (required, actually available) and associations to other decision scenarios.
SR33	Requirement SR33: In order to describe the path, a decision is supposed to take within an organisa-

	tion, decision scenarios require an appropriate combination with business process models.
SR34	Requirement SR34: Different types of associations should allow for assigning organisational units, roles and committees to business process models, e.g. "in charge of", "supports", "provides technical support" etc.
SR35	Requirement SR35: It should be possible to conveniently express constraints on instances of organisational units, roles and committees assigned to processes. If, e.g., the position "Sales Assistant" is assigned to more than one process within a business process, it may be important to express that it should always be the same position instance (i.e. the same employee) who is assigned.
SR36	Requirement SR36: Referring to an organisational chart maybe regarded as helpful in some cases. In many cases, it will add to a diagrams complexity and distract from the main focus, i.e. the business process diagram. Hence, the notation should allow for assigning organisational units without representing the corresponding organisational chart. It should also allow for clearly distinguishing between different kinds of assignments.
SR37	Requirement SR37: It should be possible to express relevant constraints on the assignment of organisational units or roles/committees to processes. For example: If organisational units and roles were assigned the tasks they are supposed to perform and all processes were decomposed into tasks, then a constraint could be applied that only those organisational units etc. can be in charge of performing a process that cover all corresponding tasks.
SR38	Requirement SR38: In addition to a set of predefined relationship types, it should be possible to define further **relationship** types **(corresponds** to **Requirement A2**).
SR39	Requirement SR39: As far as possible, concepts for business process modelling should be reused. This does not only foster the maintenance of the language, it also allows for reusing associations – e.g. to organisational units, resources, classes etc. – defined for business process diagrams.
SR40	Requirement SR40: There is need to account for concepts that are specific for project planning, e.g. to express problems, risks, challenges as well as accomplishments.
SR41	Requirement SR41: With respect to the high level of abstraction to be expected for some project execution plans, they should be supplemented with guidelines for "instantiating" project instances.
SR42	Requirement SR42: It should be possible to associate projects or project phases with concepts of other diagram types that help with analysing and (re-) designing the project categories of a firm.
SR43	Requirement SR43:**Fehler! Verweisquelle konnte nicht gefunden werden.**
SR44	Requirement SR44: Concepts within the IT resource modelling language, e.g. to describe hardware, system software, networks, applications etc. need to be included in the OrgML in order to allow for defining references.
SR45	Requirement SR47: Integrating OrgML models with other models used within an enterprise model requires including all concepts of the corresponding modelling languages that are used for inter-model associations.

SR46	Requirement SR48: To cope with modifications of existing modelling languages and the creation of further languages, it might be required to provide versatile linking mechanisms on a low semantic level.

Table 3: Specific Requirements

6 Conclusions

The study presented in this paper aimed at a foundation for specifying DSML for organisation modelling. While organisational design comprises mainly business processes and the organisational structure, the scope to be accounted for is larger. For various kinds of analysis and design it is required to account for context. Therefore, the requirements that resulted from our investigation include concepts, too, that allow for representing specific aspects of organisational issues and others that allow for integrating organisational models with other parts of an enterprise model. The extensive list of requirements, which is summarised in the three tables above, shows that specifying a DSML for organisation modelling is a demanding an laborious task. The presented work is intended to cover a wide range of requirements related to organisation modelling in the context of enterprise modelling. Hence, an organisation modelling language does not have to address all requirements. Nevertheless, the list of requirements could then serve to describe the scope of a particular DSML. At the same time, the list is not meant to be complete. The requirements are based on the analysis of use scenarios that seem particularly interesting. Even though the list of use scenarios has evolved over a considerable time period, it can hardly be considered as complete. Therefore, future research will involve the creation of further use scenarios and – presumably – the refinement of respective requirements.

References

BÖRGER, E. 2007. Modeling workflow patterns from first principles. *In:* PARENT, V. C., SCHEWE, K.-D. & THALHEIM, B. (eds.) *Conceptual Modeling–ER 2007. Vol. 4801 of Lecture Notes in Computer Science.* Berlin: Springer.

BUNGE, M. A. 1977. *The furniture of the world*, Dordrecht and Boston: Reidel

FETTKE, P. & LOOS, P. 2003. Ontological evaluation of reference models using the Bunge-Wand-Weber-model. *Proceedings of the Ninth Americas Conference on Information Systems.* Tampa.

FRANK, U. 1998. Evaluating Modelling Languages: Relevant Issues, Epistemological Challenges and a Preliminary Research Framework. Institut für Wirtschaftsinformatik, Universität Koblenz-Landau

FRANK, U. 2011. Multi-Perspective Enterprise Modelling: Background and Terminological Foundation. *ICB Research Report, No. 46, Universität Duisburg-Essen*

FRANK, U. 2001. Organising the Corporation: Research Perspectives, Concepts and Diagrams. Institut für Wirtschaftsinformatik, Universität Koblenz-Landau

FRANK, U. 2010. Outline of a Method for Designing Domain-Specific Modelling Languages. *ICB Research Report, No. 42, Universität Duisburg-Essen*

FRANK, U. 2006. Towards a Pluralistic Conception of Research Methods in Information Systems. *ICB Research Report, No. 7, Universität Duisburg-Essen*

FRANK, U. & LAAK, B. 2003. Anforderungen an Sprachen zur Modellierung von Geschäftsprozessen. Institut für Wirtschaftsinformatik, Universität Koblenz-Landau

JUNG, J. 2007. *Entwurf einer Sprache für die Modellierung von Ressourcen im Kontext der Geschäftsprozessmodellierung,* Berlin, Logos Verlag.

JUNG, J. 2004. Mapping of Business Process Models to Workflow Schemata. An Example Using MEMO-OrgML and XPDL. Universität Koblenz-Landau.

KEEN, P. G. W. 1991. *Shaping the future: Business design through information technology,* Boston, Harvard Business School Press.

KIRCHNER, L. 2008. *Eine Methode zur Unterstützung des IT-Managements im Rahmen der Unternehmensmodellierung,* Berlin, Logos Verlag.

KIRCHNER, L. 2007. Entwurf einer Modellierungssprache zur Unterstützung der Aufgaben des IT-Managements. Grundlagen, Anforderungen und Metamodell. *ICB Research Report, No. 11, Universität Duisburg-Essen*

LORENZ, K. 1996. Sprache. *In:* MITTELSTRAS, J. (ed.) *Enzyklopädie Philosophie und Wissenschaftstheorie.* Stuttgart: Metzler.

MORGAN, G. 1986. *Images of Organization,* London, Thousand Oaks.

OPDAHL, A. L. & HENDERSON-SELLERS, B. 1999. Evaluating and Improving OO Modelling Languages Using the BWW-Model. *Proceedings of the Information Systems Foundations Workshop (Ontology, Semiotics and Practice).* Sydney.

RUSSEL, N., HOFSTEDE, A. H. M., VON DER AALST, W. M. P. & MULYAR, N. 2006. Workflow Control-Flow Patterns: A Revised View. . *BPM-Report BPM-06-22.*

SCHAUER, H. 2008. *Unternehmensmodellierung für das Wissensmanagement – Eine multi-perspektivische Methode zur ganzheitlichen Analyse und Planung,* Saarbrücken, VDM Verlag Dr. Müller.

VON DER AALST, W. M. P., HOFSTEDE, A. H. M., KIEPUSZEWSKI, B. & BARROS, A. P. 2003. Workflow Patterns. *Distributed and Parallel Databases,* 14**,** 5-51.

WEBER, R. 1997. *Ontological Foundations of Information Systems,* Melbourne, Coopers&Lybrand.

WEICK, K. E. 1979. The Social Psychology of Organizing. *In:* ADDISON-WESLEY (ed.) *Reading.* 2nd ed. ed.: Mass.

Previously published ICB - Research Reports

2011

No 46 (December 2011)

Frank, Ulrich: "Multi-Perspective Enterprise Modelling: Background and Terminological Foundation"

No 45 (November 2011)

Frank, Ulrich; Strecker, Stefan; Heise, David; Kattenstroth, Heiko; Schauer, Carola: "Leitfaden zur Erstellung wissenschaftlicher Arbeiten in der Wirtschaftsinformatik"

No 44 (September 2010)
Berenbach, Brian; Daneva, Maya; Dörr, Jörg; Frickler, Samuel; Gervasi, Vincenzo; Glinz, Martin; Herrmann, Andrea; Krams, Benedikt; Madhavji, Nazim H.; Paech, Barbara; Schockert, Sixten; Seyff, Norbert (Eds): "17th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2011). Proceedings of the REFSQ 2011 Workshops REEW, EPICAL and RePriCo, the REFSQ 2011 Empirical Track (Empirical Live Experiment and Empirical Research Fair), and the REFSQ 2011 Doctoral Symposium"

No 43 (February 2011)

Frank, Ulrich: "The MEMO Meta Modelling Language (MML) and Lnguage Architecture – 2nd Edition"

2010

No 42 (December 2010)

Frank, Ulrich: "Outline of a Method for Designing Domain-Specific Modelling Languages"

No 41 (December 2010)

Adelsberger,Heimo; Drechsler, Andreas (Eds): "Ausgewählte Aspekte des Cloud-Computing aus einer IT-Management-Perspektive – Cloud Governance, Cloud Security und Einsatz von Cloud Computing in jungen Unternehmen"

No 40 (October 2010)
Bürsner, Simone; Dörr, Jörg; Gehlert, Andreas; Herrmann, Andrea; Herzwurm, Georg; Janzen, Dirk; Merten, Thorsten; Pietsch, Wolfram; Schmid, Klaus; Schneider, Kurt; Thurimella, Anil Kumar (Eds): "16th International Working Conference on Requirements Engineering: Foundation for Software Quality. Proceedings oft he Workshops CreaRE, PLREQ, RePriCo and RESC"

No 39 (May 2010)
Strecker, Stefan; Heise, David; Frank, Ulrich: "Entwurf einer Mentoring-Konzeption für den Studiengang M.Sc. Wirtschaftsinformatik an der Fakultät für Wirtschaftswissenschaften der Universität Duisburg-Essen"

No 38 (February 2010)
Schauer, Carola: "Wie praxisorientiert ist die Wirtschaftsinformatik? Einschätzungen von CIOs und WI-Professoren"

No 37 (January 2010)

Benavides, David; Batory, Don; Grunbacher, Paul (Eds.): "Fourth International Workshop on Variability Modelling of Software-intensive Systems"

2009

No 36 (December 2009)

Strecker, Stefan: "Ein Kommentar zur Diskussion um Begriff und Verständnis der IT-Governance - Anregungen zu einer kritischen Reflexion"

No 35 (August 2009)

Rüngeler, Irene; Tüxen, Michael; Rathgeb, Erwin P.:"Considerations on Handling Link Errors in STCP"

No 34 (June 2009)

Karastoyanova, Dimka; Kazhamiakan, Raman; Metzger, Andreas; Pistore, Marco (Eds.): "Workshop on Service Monitoring, Adaption and Beyond"

No 33 (May 2009)

Adelsberger,Heimo; Drechsler , Andreas; Bruckmann, Tobias; Kalvelage, Peter; Kinne, Sophia; Pellinger, Jan; Rosenberger, Marcel; Trepper, Tobias: „Einsatz von Social Software in Unternehmen – Studie über Umfang und Zweck der Nutzung"

No 32 (April 2009)

Barth, Manfred; Gadatsch, Andreas; Kütz, Martin; Rüding, Otto; Schauer, Hanno; Strecker, Stefan: „Leitbild IT-Controller/-in – Beitrag der Fachgruppe IT-Controlling der Gesellschaft für Informatik e. V."

No 31 (April 2009)

Frank, Ulrich; Strecker, Stefan: "Beyond ERP Systems: An Outline of Self-Referential Enterprise Systems – Requirements, Conceptual Foundation and Design Options"

No 30 (February 2009)

Schauer, Hanno; Wolff, Frank: „Kriterien guter Wissensarbeit – Ein Vorschlag aus dem Blickwinkel der Wissenschaftstheorie (Langfassung)"

No 29 (January 2009)

Benavides, David; Metzger, Andreas; Eisenecker, Ulrich (Eds.): "Third International Workshop on Variability Modelling of Software-intensive Systems"

2008

No 28 (December 2008)

Goedicke, Michael; Striewe, Michael; Balz, Moritz: „Computer Aided Assessments and Programming Exercises with JACK"

No 27 (December 2008)

Schauer, Carola: "Größe und Ausrichtung der Disziplin Wirtschaftsinformatik an Universitäten im deutschsprachigen Raum - Aktueller Status und Entwicklung seit 1992"

No 26 (September 2008)

Milen, Tilev; Bruno Müller-Clostermann: " CapSys: A Tool for Macroscopic Capacity Planning"

No 25 (August 2008)
Eicker, Stefan; Spies, Thorsten; Tschersich, Markus: "Einsatz von Multi-Touch beim Softwaredesign am Beispiel der CRC Card-Methode"

No 24 (August 2008)
Frank, Ulrich: "The MEMO Meta Modelling Language (MML) and Language Architecture – Revised Version"

No 23 (January 2008)
Sprenger, Jonas; Jung, Jürgen: "Enterprise Modelling in the Context of Manufacturing – Outline of an Approach Supporting Production Planning"

No 22 (January 2008)
Heymans, Patrick; Kang, Kyo-Chul; Metzger, Andreas, Pohl, Klaus (Eds.): "Second International Workshop on Variability Modelling of Software-intensive Systems"

2007

No 21 (September 2007)
Eicker, Stefan; Annett Nagel; Peter M. Schuler: "Flexibilität im Geschäftsprozess-management-Kreislauf"

No 20 (August 2007)
Blau, Holger; Eicker, Stefan; Spies, Thorsten: "Reifegradüberwachung von Software"

No 19 (June 2007)
Schauer, Carola: "Relevance and Success of IS Teaching and Research: An Analysis of the ‚Relevance Debate'

No 18 (May 2007)
Schauer, Carola: "Rekonstruktion der historischen Entwicklung der Wirtschaftsinformatik: Schritte der Institutionalisierung, Diskussion zum Status, Rahmenempfehlungen für die Lehre"

No 17 (May 2007)
Schauer, Carola; Schmeing, Tobias: "Development of IS Teaching in North-America: An Analysis of Model Curricula"

No 16 (May 2007)
Müller-Clostermann, Bruno; Tilev, Milen: "Using G/G/m-Models for Multi-Server and Mainframe Capacity Planning"

No 15 (April 2007)
Heise, David; Schauer, Carola; Strecker, Stefan: "Informationsquellen für IT-Professionals – Analyse und Bewertung der Fachpresse aus Sicht der Wirtschaftsinformatik"

No 14 (March 2007)
Eicker, Stefan; Hegmanns, Christian; Malich, Stefan: "Auswahl von Bewertungsmethoden für Softwarearchitekturen"

No 13 (February 2007)
Eicker, Stefan; Spies, Thorsten; Kahl, Christian: "Softwarevisualisierung im Kontext serviceorientierter Architekturen"

No 12 (February 2007)
Brenner, Freimut: "Cumulative Measures of Absorbing Joint Markov Chains and an Application to Markovian Process Algebras"

No 11 (February 2007)
Kirchner, Lutz: "Entwurf einer Modellierungssprache zur Unterstützung der Aufgaben des IT-Managements – Grundlagen, Anforderungen und Metamodell"

No 10 (February 2007)
Schauer, Carola; Strecker, Stefan: "Vergleichende Literaturstudie aktueller einführender Lehrbücher der Wirtschaftsinformatik: Bezugsrahmen und Auswertung"

No 9 (February 2007)
Strecker, Stefan; Kuckertz, Andreas; Pawlowski, Jan M.: "Überlegungen zur Qualifizierung des wissenschaftlichen Nachwuchses: Ein Diskussionsbeitrag zur (kumulativen) Habilitation"

No 8 (February 2007)
Frank, Ulrich; Strecker, Stefan; Koch, Stefan: "Open Model - Ein Vorschlag für ein Forschungsprogramm der Wirtschaftsinformatik (Langfassung)"

2006

No 7 (December 2006)
Frank, Ulrich: "Towards a Pluralistic Conception of Research Methods in Information Systems Research"

No 6 (April 2006)
Frank, Ulrich: "Evaluation von Forschung und Lehre an Universitäten – Ein Diskussionsbeitrag"

No 5 (April 2006)
Jung, Jürgen: "Supply Chains in the Context of Resource Modelling"

No 4 (February 2006)
Lange, Carola: "Development and status of the Information Systems / Wirtschaftsinformatik discipline: An interpretive evaluation of interviews with renowned researchers, Part III – Results Wirtschaftsinformatik Discipline"

2005

No 3 (December 2005)
Lange, Carola: "Development and status of the Information Systems / Wirtschaftsinformatik discipline: An interpretive evaluation of interviews with renowned researchers, Part II – Results Information Systems Discipline"

No 2 (December 2005)
Lange, Carola: "Development and status of the Information Systems / Wirtschaftsinformatik discipline: An interpretive evaluation of interviews with renowned researchers, Part I – Research Objectives and Method"

No 1 (August 2005)
Lange, Carola: „Ein Bezugsrahmen zur Beschreibung von Forschungsgegenständen und -methoden in Wirtschaftsinformatik und Information Systems"

Research Group	Core Research Topics
Prof. Dr. H. H. Adelsberger Information Systems for Production and Operations Management	E-Learning, Knowledge Management, Skill-Management, Simulation, Artificial Intelligence
Prof. Dr. P. Chamoni MIS and Management Science / Operations Research	Information Systems and Operations Research, Business Intelligence, Data Warehousing
Prof. Dr. F.-D. Dorloff Procurement, Logistics and Information Management	E-Business, E-Procurement, E-Government
Prof. Dr. K. Echtle Dependability of Computing Systems	Dependability of Computing Systems
Prof. Dr. S. Eicker Information Systems and Software Engineering	Process Models, Software-Architectures
Prof. Dr. U. Frank Information Systems and Enterprise Modelling	Enterprise Modelling, Enterprise Application Integration, IT Management, Knowledge Management
Prof. Dr. M. Goedicke Specification of Software Systems	Distributed Systems, Software Components, CSCW
Prof. Dr. V. Gruhn Software Engineering	Design of Software Processes, Software Architecture, Usability, Mobile Applications, Component-based and Generative Software Development
PD Dr. C. Klüver Computer Based Analysis of Social Complexity	Soft Computing, Modeling of Social, Cognitive, and Economic Processes, Development of Algorithms
Prof. Dr. T. Kollmann E-Business and E-Entrepreneurship	E-Business and Information Management, E-Entrepreneurship/E-Venture, Virtual Marketplaces and Mobile Commerce, Online-Marketing
Prof. Dr. B. Müller-Clostermann Systems Modelling	Performance Evaluation of Computer and Communication Systems, Modelling and Simulation
Prof. Dr. K. Pohl Software Systems Engineering	Requirements Engineering, Software Quality Assurance, Software-Architectures, Evaluation of COTS/Open Source-Components
Prof. Dr.-Ing. E. Rathgeb Computer Networking Technology	Computer Networking Technology
Prof. Dr. E. Rukzio Mobile Mensch Computer Interaktion mit Software Services	Novel Interaction Technologies, Personal Projectors, Pervasive User Interfaces, Ubiquitous Computing
Prof. Dr. R. Unland Data Management Systems and Knowledge Representation	Data Management, Artificial Intelligence, Software Engineering, Internet Based Teaching
Prof. Dr. S. Zelewski Institute of Production and Industrial Information Management	Industrial Business Processes, Innovation Management, Information Management, Economic Analyses

For more information visit us on the Web: **http://www.icb.uni-due.de**

ISSN 1860-2770 (Print)
ISSN 1866-5101 (Online)