JÜRGEN JUNG

SOME REFLECTIONS ON THE BASIC CONCEPTUALISATION OF A RESOURCE MODELLING LANGUAGE FOR BUSINESS PROCESS MODELLING - CONCEPTS, REQUIREMENTS AND OPEN RESEARCH QUESTIONS

June 2003

Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 35

Die Arbeitsberichte des Instituts für 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 "Arbeitsberichte des Instituts für Wirtschaftsinformatik" comprise preliminary results which will usually be revised for subsequent publications. Critical comments would be appreciated by the authors.

Anschrift der Verfasser
Address of the authors:

Dipl. Inform. Jürgen Jung
Institut für Wirtschaftsinformatik
Universität Koblenz-Landau
Universitätsstraße 1
D-56070 Koblenz

Arbeitsberichte des Instituts für Wirtschaftsinformatik
Herausgegeben von / Edited by:

Prof. Dr. Ulrich Frank
Prof. Dr. J. Felix Hampe
Prof. Dr. Klaus G. Troitzsch

Bezugsquelle / Source of Supply:

Institut für Wirtschaftsinformatik
Universität Koblenz-Landau
Universitätsstraße 1
56070 Koblenz

Tel.: 0261-287-2520
Fax: 0261-287-2521
Email: iwi@uni-koblenz.de
WWW: http://www.uni-koblenz.de/~iwi

Abstract

This research report presents an overview of some basic ideas regarding the conceptualisation of a resource description language for business process modelling. This language is an extension to an existing process modelling language — MEMO-OrgML. The basic conceptualisation comprises basic resource types and their mutual relations. It aims to satisfy the needs of different user types and abstractions. The language will be applicable by domain experts and offer domain specific resources and resource types for various domains by offering adequate abstraction on resources. A specialised resource modelling language fosters the reuse of common concepts and assures the integrity of a business process model. The current version of this report has to be regarded as a first draft regarding general concepts of a resource modelling language, requirements and —most of all— open research questions.

Contents

List of Figures

1 Introduction

This research report presents a work in progress at the IS Research Institute at the University of Koblenz (Germany). It describes the conceptualisation of a resource modelling language as an extension to an existing process modelling language developed by the research group *Enterprise Modelling* at the IS Research Institute. The resource modelling language aims to add language features for the specification of resources on a domain-specific level and offers formal criteria for a further mapping to IT-related resources and the development of a corporate information system.

In well established process modelling languages, business processes and their relationships only describe *what* has to be done. Resources assigned to processes specify *who* (human resources or autonomous machinery) has to work on the process and *what* (e.g. machinery, tools, expedients) will be needed. Modelling of resources will enrich the semantic expressiveness of conventional business process models. Additionally, the resource model might be integrated with existing object-models or other models describing an organisation. This integration[1] is a key issue for the design of an integrated business process language. It offers domain-specific concepts and ensures the integrity of a model regarding resources. The different MEMO languages base on an object-oriented approach. They are specified by a meta-model and share common abstractions in an underlying object model. Despite the importance of the elaboration of an object-oriented approach as a basis for the resource modelling language, this report only describes basic ideas for the conceptualisation of such a language. The conceptualisation mainly comprises some resource-related concepts, requirements for the modelling of resources and open research questions. It focusses on domain-specific aspects and abstracts from a formal base for the integration of resources into MEMO as long as the concepts are far from being precise enough for an adequate integration into the MEMO meta-model.

1.1 Process Modelling

The analysis, representation and management of knowledge about an organisation and its processes has always been very important [25]. A lot of work has been done on the development and evaluation of ontologies for process modelling [39, 40, 41, 42, 43, 44, 18], the specification of process modelling

[1]Especially the relationships between entities (objects) of a resource model and objects of an object model are nor discussed within this report. Nonetheless, such information might enrich the semantic expressiveness of an organisational model. But this report only focusses on the basic conceptualisation of a resource modelling language and prescinds —by now– from the formal specification of the integration of the resource modelling language into object models.

languages [8, 30, 37, 1] as well as on business process modelling methods and concepts [20, 32]. Business process models can be used for different kinds of purposes:

- Documentation of processes of an organisation to foster communication [12, 30]
- Analysis of business processes [8, 4, 36]
- Simulation of processes [2]
- Support for business process re-engineering [6, 30]
- Generation of workflow schemata [6, 30]
- Software development of process-oriented applications [12, 35, 34, 32, 6]

The documentation of an organisation's processes (as well as other organisational aspects like its structure or strategy) fosters communication with new employees or external consultants [12, 30]. Business process models represent a common medium for the communication of domain experts and novices. They offer domain level concepts[2] and enable a broader distribution of knowledge among other business-related people with different skills and knowledge of an organisation.

The analysis of business processes relies on a relatively detailed description of process models and according concepts. Depending on the analysis' purpose, a modelling language has to offer language features for the modelling of the facts which are in its scope. Analysis might for example support the detection of weaknesses in existing processes [8, 4, 36]. Appropriate language features provided by a process modelling language support the determination of media clashes[3], unnecessary processes[4] or potentials for further optimisations. Nevertheless, the potential for further optimisations relies on the degree of formal description of the business process model. Depending on identified weaknesses, a business process re-engineering might be applicable [6, 30].

Simulation supports the detection of weaknesses of a business process model [2] as well as analysis. In contrast to analysis, simulation does not rely on structural properties of a business process. Simulation generally allows the prototypical execution of previously designed processes on the basis of concrete entities. Depending on the process and the selected entities, a prototypical execution of a certain process can be started and observed by the designer. A simulation usually allows the observation of a processes

[2]In contrast to other modelling purposes and languages, the level of abstraction is very high with respect to the degree of formalisation.

[3]If different kinds of media are included.

[4]Depending on the degree of detailed descriptions of process models.

execution depending on given input parameters. It is in some extend more powerful with respect to expressiveness than a static analysis. Simulation can focus on aspects of a model which have not been considered in advance. Hence, simulation enables the detection of additional properties by the observation of the behavior of a process. It also shows unknown properties and restrictions regarding a new business process. Simulation depends on a prototypical instantiations of a process model[5]. Typical instances of objects and their values (e.g. a specific processing time or accounting information) are added to the model and the execution of a process can be observed. Depending on the objectives, business process re-engineering supports the redesign of processes with respect to weaknesses identified by an analysis or a simulation [6, 30].

Business process models might also be a preliminary stage for an information system's (IS) design. A workflow-management-system (WfMS) or a newly developed software system are alternatives for such an IS [12, 35, 34, 32, 6]. The distinction between a workflow schema and a software system is the level of coding. Workflow schemata can usually directly derived from a process-model. Process types are mapped to atomic workflows and resources are are assigned accordingly. This is because of the predefined semantics of WfMS. A workflow s only described by its processes and associated applications. Hence, the applicability of such systems is restricted to classical computer-supported processes. The extend of these capabilities is usually determined by the WfMS. In contrast to this, the development of a new information system does not depend on the limitations of a given workflow framework[6]. Such a system is usually build using low-level programming languages and domain-specific frameworks.

Resources are essential for the modelling of processes [33]. Processes and their relationships only describe *what* has to be done. Resources assigned to processes specify *who* has to work on the process and *what* will be needed. Resource are usually not available in an unlimited manner [29, 33]. Modelling resources offers the opportunity for the determination of the efficiency of a process according to economic aspects. Hence, the usage of scarce resources has to be taken into account for the analysis or simulation of processes as well as for the development of a workflow application or an information system. Bottlenecks resulting from scarce resources can be identified and supported by alternative resource which may replace the original resource in case of a failure. But, the quality of analysis, simulation, and system development depend on the conceptual power of the resource

[5]The prototypical instantiation of a process model means the mapping of a conceptual process model to a simulation model with additional properties of an instance. Such an instance might correspond to the instance of a process and an instances of associated entities as well.

[6]like a workflow management system

modelling language. Hence, such a language should offer:

- domain specific concepts to be (re-)used by domain experts
- semantically rich resource types including integrity constraints
- support for analysis and simulation
- potential for the further development of information systems [7]

Many process modelling languages have been developed in the last years. Some of them also include the specification of resources on different levels of abstraction. Petri-nets offer a formal language for the specification of processes [2]. In classical Place/Transition-Nets, resources (i.e. their states) can only be modelled by a subnet. Higher-level Petri-nets offer an extended expressiveness by using tuples as markings (instead of anonymous markings like classical Petri-nets) [2]. Oberweis developed a language based on Petri-nets but using complex relations as markings [30]. Nonetheless, these approaches offer only a few language features for the descriptions of resources. Even worse, the concept of a *resource*[8] itself usually does not exist explicitly. Also Scheer's event-driven process chains (EPC) do not include specific resources [34, 35, 36]. All entities participating on a business process have to be modelled by an entity-relationship-diagram (ERD). The current version of the Unified Modelling Language (UML) provides facilities for business process modelling [3, 26, 9]. Like in all previously presented approaches, resources are not a specific language feature and have to be added by the user.

This paper presents some basic ideas on the development of a resource specification language for business processes and relates to a current work in progress. The resource modelling language aims to satisfy the needs of different user types. The language has to be applicable by domain experts and offer domain specific resources and resource types for various domains by offering adequate abstractions on resources. This resource modelling language is an extension to an existing process modelling language — OrgML (Organisation Modelling Language) — which is part of an integrated enterprise modelling method MEMO (Multi-Perspective Enterprise MOdelling) [12].

[7] The potential for the development of information systems might be realised in the computer-supported generation of schemas for workflow management systems or the ability for requirements engineering or domain analysis for with respect to software development.

[8] And also the term *resource* as well.

1.2 MEMO and Resource Modelling

MEMO is a method for modelling organisations on different levels of abstraction and from different perspectives. MEMO has been initiated by Ulrich Frank and is the main research topic of the research group 'Enterprise Modelling' at the University of Koblenz. MEMO includes several languages for modelling static, functional and dynamic aspects of an enterprise. One of these languages is the MEMO-OrgML, which supports modelling of organisational structures and processes. Resource modelling has not been subject of the first conceptualisation of the MEMO-OrgML. Our first approach for modelling resources with MEMO-OrgML is presented here.

A resource modelling language supports the domain-specific usage of resource-types. A predefined resource modelling language might guarantee the integrity of the usage of resources in specific process types. Domain-specific circumstances might result in a restriction of the usage of specific resources. Also the usage of one resource might result in the usage of other resources. Hence, dependencies between different resources — and resource types as well — should be specified and support an adequate allocation of resources to given processes. Reuse of resource descriptions should be supported by given language features and the reusability of already defined resources. An extensive library of given resources and types supports the reuse of previously specified concepts. Furthermore, resource-specific language features for the modelling of resources allow for the reuse of resource-dependent properties of given resources.

Generally speaking, a proper conceptualisation of a resource modelling language and related resources[9] fosters a proper application and specification of resources. Integrity of the usage of resources and resource type might be assured by the reuse of given resource types. A resource modelling language should offer domain-specific concepts for the description of resources used in a business process model. Such a language implies two benefits. Firstly, it supports the reuse of already defined concepts. Resources and their corresponding properties do not have to be modelled explicitly and can, therefore, be reused. Secondly, the usage of a resource modelling language helps to ensure integrity constraints. Those constraints are given by the conceptualisation of a resource modelling language. The proper application of hereby related language features fosters the appropriateness of resource models for given business process models.

[9] This conceptualisation also comprises resource types.

1.3 Levels of Abstraction

Many everyday terms are ambiguous with respect to their level of abstraction[10]. The term 'resource' might be interpreted as resource type (i.e. a resource class[11]) or equally likely as an instance of resource type (e.g. an existing database server in the corporate network). Furthermore, a resource class can be looked at from an intensional or an extensional point-of-view. The intensional interpretation of a class corresponds to a template for the specification of instances. A class named Person defines the attributes `firstname`, `lastname` and `dateOfBirth` for all persons. This template is used to instantiate new person-instances of the class Person. By contrast, the class extension is a set of objects with common properties. Ambiguities of interpretations have to be avoided in the context of business process modelling for formal analysis, simulation or software development. Furthermore, additional aspects of resource classification in a process modelling language have to be taken into account. Such aspects are illustrated by example of a PDA[12] in Figure 1[13]. The resource modelling language concepts are at present described by OML-diagrams[14] regarding to a future meta-model. Instances of types in the meta-model are resource types such as the PDA in Figure 1[15]. The resource type PDA may be classified from a technical or economic point of view. Technical aspects are model, processor speed, and size of the main memory. These aspects determine the restrictions of usage for such a resource type in a definite context. Economic properties of a PDA resource type address cost of operation and maintenance of a specific instance in a business process. Those aspects are described at different levels of abstraction by the same resource type.

An initialised resource type 'Compaq iPaq H3660' is an instance of the resource type *PDA* and can in turn also have instances (like an iPaq with the serial number 4775348 at the bottom of Figure 1). Consequently, we have a multi-level type-instance-relationship for the description of PDAs in a resource model. This relationship, however, is hard to handle in the context of business process modelling. Common object-oriented programming lan-

[10]The terms might also be interpreted accordingly. Different perspectives and individual backgrounds can lead to various interpretations of the same term. Not only the levels of abstractions result in ambiguities but also the personal context of a developer.

[11]The terms 'type' and 'class' are used synonymously within this paper.

[12]The abbreviation stands for Personal Digital Assistant and describes a pocket-sized mobile computer.

[13]Meta-models in MEMO are described by the MEMO-OML —the MEMO Object Modelling Language. The notation of the object models used within this report does not reflect the original OML notation. The notation is rather leant against the UML notation but corresponds to concepts of the OML.

[14]using the UML notation

[15]A similar illustration can be found in [16].

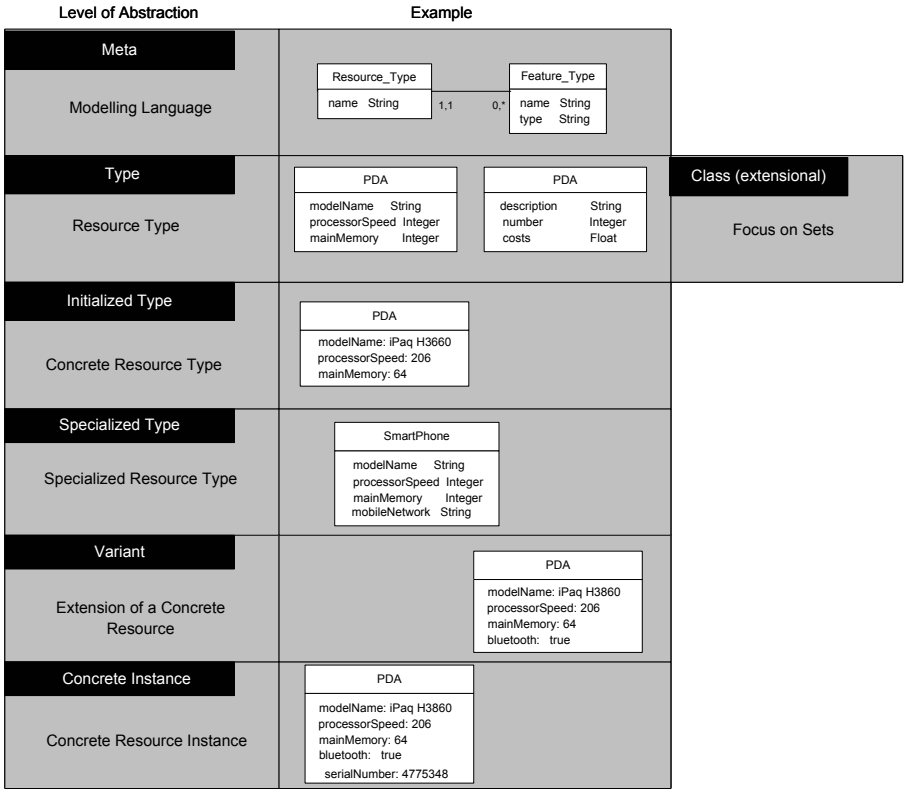

Figure 1: Levels of Abstraction

guages only offer a flat class-instance-relationship. Therefore, type-instance relationships of resource models can hardly be mapped directly to the programming language's concepts.

To make things even more complex it is not clear whether special PDAs (e.g. a SmartPhone) are subtypes or instances of the PDA resource type. On the one hand, they may be regarded as subtypes because they have an additional features compared to standard PDAs. This feature is the build-in mobile telecommunication facility. On the other hand, a SmartPhone is of the resource type PDA because it might be a variant of an existing PDA with an additional mobile telecommunication facility (built-in or attached).

Conceptual modelling usually prescinds from concrete objects and changeable aspects. Hence, only a processes type is favored in conceptual models. Despite this fact, modelling of instances might be appropriate in certain situations:

- The modelling of —anonymous but separately identifiable— instances is used to formulate relationships between resources in different pro-

cesses. Such a relationship may be of the kind: The resource (instance) used in process A is also required in process B. Such a resource might be referred to by its unique identification but remains anonymous on a conceptual level.

- A prototypical instance in a process model reflects representative property-values of a resource type. Such representative values may depend on average values, user defined parameters or other user-defined aspects[16]. Costs for the usage of a resource are usually attached to the concrete resource instance and differ between instances. Hence, costs can hardly be assigned to resource types in conceptual modelling. But average costs of a certain resource type might be assigned to a prototypical instance which is used for formal analysis and simulation.
- Concrete instances might be required for the illustration of the current state of an organisation's information system. The first step of business process reengineering is the modelling of existing business processes. This model includes processes and resources as they are at the moment. This might also include specific resources like special servers, relevant machinery, or technical engineers, which are hard to compensate.

Depending on the modelling purpose, the modelling of types and instances might be supported by a resource modelling language. A user does not have to specify instantiated resources but he might get the opportunity to do so. The specification of concrete resources can result in a huge set of language features but it might support the adequate description of instances —if needed. Whenever the precise description of resources is required, a language should offer adequate language features. Such a language should also support different perspectives (technical and economic). Furthermore, different levels of abstraction have to be available with respect to type-instance-relationships. We will present our first approach for resource modelling in the following section. This approach will cover many of the discussed aspects.

2 Resource-Types

The foundation of the resource modelling language is the meta-model for the specification of resources and their types. The current version of the core of this language is presented in figure 2. The purpose of this meta-model is the

[16]Additional —user-defined— attributes might be added to the simulation. Such attributes reflect additional properties required for the simulation but which are not an inherent part of the domain model itself. Examples for such properties (and associated values) are estimated costs or average computing time.

description of abstract resources on a conceptual level. Resources are seen from the perspective of a domain expert, who deals with existing resource types assigned to business processes. Accounting aspects are excluded and will be discussed in section 4. The core resource modelling language aims to offer domain-specific resource types like machinery, raw material, staff qualification, and their respective types.

2.1 Basic Meta-Model of Resource Types

At the top level of the class hierarchy of the meta-model we distinguish between compound and elementary resources types. A compound resource type is an abstract resource type, which is composed of other abstract resource types. Hence, a compound resource type may consist of several elementary or compound resource types. Elementary resource types are specialised to human, physical, and intangible resource types. A human resource type corresponds to an organisational unit or a role filled by a person. Physical resource types comprise all tangible objects used within a business process. The classification of these resource types correlates to their economic differentiation between operational and consumptionable resources(cf. [7]). Media don't seem to fit in this classification. Consequently, they are put into a different category. In our first approach, subtypes of physical resource types are resource types, which

- are used for the completion of a process and are still available afterwards (`OperationalResourceType`)
- are consumed by the completion of a process (`ConsumptionableResourceType`), or
- store information (`MediumType`).

Operational resource types are physical resource types, which are used within a process and are still available after its completion. Examples are machinery, tools, and vehicles. They are all used for processing but remain available. In contrast, consumable resource types are a prerequisite for a process and are transformed during the execution of a process. Raw and operational material as well as spare parts are used by a business process and are transformed to a (partial) product. They (individually) will not be available for other processes. Information containing media does not fit the differentiation between operational and consumable resource type. A medium or its information might be out-of-date after a process or it might be a prerequisite for subsequent processes. Intangible resource types represent all resource types, which are neither physical nor human. Two subtypes have

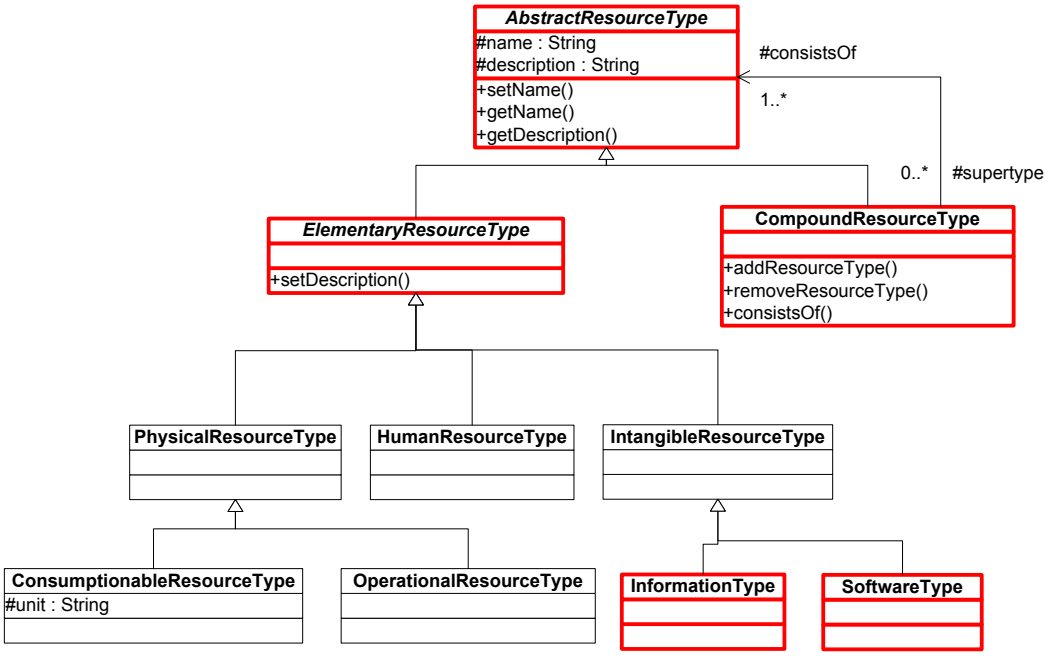

Figure 2: Basic Resource Types

been identified up to now - information and software[17]. They can not exist on their own, because they need a medium for representation - persistent or for transmission.

Open Research Questions: The basic classification as presented above has to be evaluated in future projects. It is mainly correlated to classical differentiations found in the German literature on resources[18]. Such a division into resources which can be used as a tool within a process or are consumed by the execution of a process seems to be inappropriate for media. A medium might be a reusable storage for information (e.g. a tape) or a one-way representation for data (e.g. a hardcopy). Future research has to provide experiences on the appropriateness of such a distinction.

[17]Further specifications of *information* and *software* might be put into concrete shapes by other components of MEMO – say the object model. An object model migght formally describe the structure and semantics of information. Again, these aspects are left out by the hereby described resource modelling language.

[18]In fact, many authors discuss so called *Produktionsfaktoren* (production factors) which mainly correspond to our notion of resources.

Besides this, resources might be distinguished by their activeness within a process. Hence, they might be active or passive. Active resources are autonomously capable for the achievement of the goals of a process. Examples for active resources are humans or autonomous machinery. They react on triggers and can coordinate the execution of a process depending on previously defined rules. Passive resources are not able to act without the coordination of an active resource. They are only tools, which are used by active resources. The distinction between those kinds of activity seem to be ambiguous. Is a running (email-)server process active[19] or passive[20]? Is an employee active[21] or passive[22]? Such a classification only allows the concrete application to a given organisation. The general applicability has to be examined in different projects and environments.

Generally speaking, the hereby given conceptualisation only reflects some basic thought, basing on a small fraction of literature. It has to be evaluated in practice and requires a general inspection regarding current publications. So far, the basic conceptualisation reflects our first ideas and will be improved by further research. Such research will clarify the adequateness of the focus on classical economic features or the power of additional concepts related to information systems.

2.2 Information Types

Information types correspond to different kinds of information within an organisation [5]. Information types might be specific document types like order, assembly list, and invoice [23]. An information type —respectively the structure of the information it describes— might for example reflect guidelines, regulations, and directions for business processes. All these kinds of information are covered by `InformationType` in the meta-model in figure 3.

An information type is defined by its structure, as described by the following examples:

- An invoice consists of a date, sender, receiver, and several positions.
- A guideline is structured by its document, the corresponding applicability, and its contents.

An information type's structure is encapsulated by the attribute `structureDefinition`[23].

[19] It can handle incoming emails on its own.
[20] It requires the assistance of a human operator in case of a failure.
[21] He can act on his own.
[22] The role of sick employee needs to be filled by a substitute employee.
[23] Regarding the current stage, such a definition might be more or less abstract. The definition

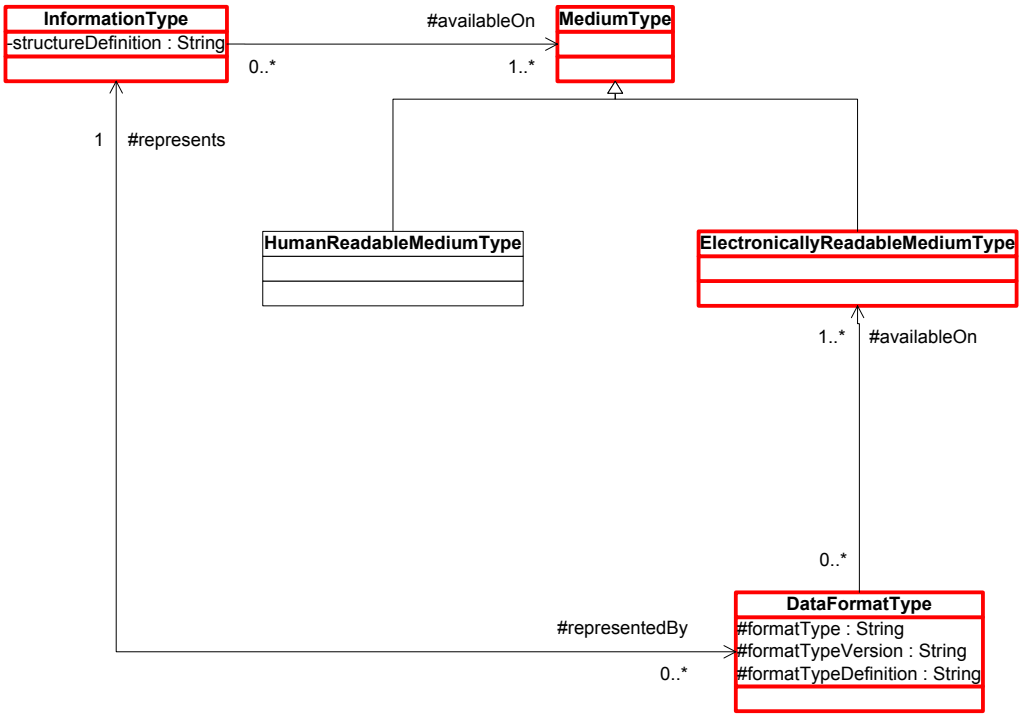

Figure 3: Information Resource Types

This definition is independent of a concrete data definition. Every information might be structured by a formal data definition but need not to be. Consequently, the structure definition of an information type is independent of a formal data definition. Such a data definition might be an XML-DTD, an SQL table definition or another structure definition language. If there is a formal structure definition, it might be assigned to the information type[24]. There has to be drawn a clear boundary between an information's format and its medium. The medium format specifies the structure a document whereas the medium contains such information. Therefore, a medium is a

can be made using natural language or being given in a formal language. Examples for useful formal languages are the Extended Bacchus-Naur-Form (EBNF) or IT-related document definition languages like —for example— XML (eXtensible Markup Language) and SGML (Structured Generalised Markup Language).

[24]With respect to a pure object-oriented approach, the specification of documents using an object-oriented description might be suitable. Such an approach will be fostered by the hereby designed resource modelling language. Consequently, features can be mapped to an object-oriented model —specified by the OML. Nevertheless, we would like to emphasise the usage of general document-related languages, such as SGML or XML.

carrier for information. We distinguish between human and electronically readable media. A human readable medium corresponds to a representation for human readers, like a paper-based hardcopy or a screen output. Electronically readable media are those kinds of media, which allow a computerised processing of information. Storage of information on an electronically readable medium is only allowed if a data-format for this information exists.

Open Research Questions: The distinction between *software* and *information* is motivated by the role of such an *intangible* resource. Software represents an active kind of an intangible resource. It acts as an active resource with respect to the execution of a program. Information is therefore a passive intangible resource. Information might be used as a regulation for the further processing of data. It is not quite clear at the moment whether the distinction between software and information is of any use. Further research has to be done.

2.3 Media Types

Information types represent a non-physical type of resource. Every information type has to be stored on a medium representing the according information. We would like to distinguish between human and electronically media in this paper[25]. Regarding media clashes, the distinction between human and electronically readable media is of a very importance. Human readable media correspond to all documents which offer an adequate representation of information to a human reader. Examples for such kinds of documents are a printed invoice, a paper-based product catalog or any kind of paper-based documents. These documents are presented in a way that they will be readable by humans and support further processing by humans. Electronically readable media are those which are based on an IT-system and allow a further processing by an information system.

This distinction is only a first attempt for the representation of different kinds of media in the context of the examination and detection of media clashes. Information itself is not restricted to humans or information systems (i.e. its readability or interpretability). Different usages by humans or computers only depend on the representation of information. Printed documents are easy to read by an employee but cannot be interpreted by a computer[26]. The distinction between human and electronically readable me-

[25]This distinction is our first approach regarding the handling of media clashes. With respect to a closer notion to media clashes, such a clash refers to the change of one medium to another. But, a media clash might not only refer to the change of a medium. It might also reflect to the transition between different data format types.

[26]A hardcopy might be digitalised and recognised by OCR-software but this results in an

dia might be not very convincing, because the set of electronic media and human readable documents[27] are certainly not disjoint. Some media are only readable by humans and others only by computer-based information systems. A fraction of media represents a basis for human or computerised processing. The projection of data on a display bases on electronically information but results in a human readable representation (a computer can only read and interpret such information only by additional technological facilities). Also the digitalisation of printed documents may foster a computerised processing. But, the transformation of information from one medium to another results in effort to be done by an employee.

Generally speaking, an information type does not change its readability with respect to human readers or information systems. All kind of readability might be reduced to the media containing an information. The medium types determines the kind of reader: human readable media are restricted to the processing by humans and electronically readable media to information systems. Examples for human readable media are every kind of representation which maps formal structures of information types to a representation adequate to a human readable medium. Human readable media might be a paper-based hardcopy or the on-screen representation of an information type. Electronically readable media correspond to an information systems containing a structured representation of information. Both representations base on the same type of information and its corresponding structure as presented in paragraph 3. The same kind of information might be represented on a human or electronically readable medium. Differences usually depend on the kind of media. Main goal of the differentiation of human and electronically readable media is the recognition of media clashes.

The conceptualisation of media types for the resource modelling language for the MEMO-OrgML is displayed in figure 4. The class `HumanReadableMediumType` comprises all media readable by humans but not by automated information systems. This includes media-types like printed documents. `ElectronicallyMediumType` comprises all kinds of media which are readable by information system components. We hereby separate persistent and transient media. Persistent media act as a storage for information types. Information is stored on the medium and will be available at any time for discovery. A transient medium acts as a transportation channel between different persistent media. In general, a transient medium is acts as an transportation channel between persistent media.

Again, the modelling of media mainly addresses the representation and

additional process for the scanning and revision of a document. Such a task can only be done by a human resource and implies additional expenditure of human capital. The abbreviation OCR stands for *Optical Character Recognition*.

[27] The term *documents* only refers to the representation of information.

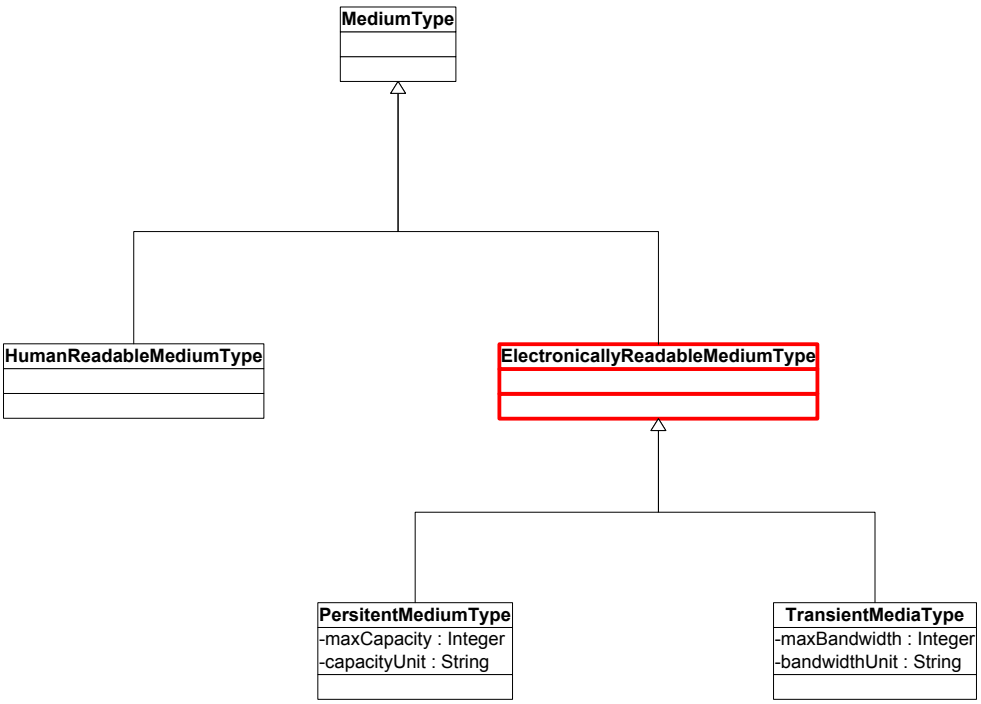

Figure 4: Media Resource Types

detection of media clashes. Such a media clash is not characterised by a transformation of information but by the change of different kind of media.

Open Research Questions: Our notion of media clashes bases on the transition between different kinds of media. Nevertheless, media clashes might also correlate to semantic clashes between different data format types. Despite the fact, that media clashes are basing on changes between different media, similar clashes might also be identified between different format representations. Focussing on human and electronically readable media covers only one aspect of media clashes. Additionally, frictions in format types reflect the same kind of clashes. Hence, media clashes might not be reduced to given media but might also contain data formats. We currently only address the detection of media clashes with respect to different media types[28] and therefore omit semantic clashes caused by different data formats. Language features for the explicit modelling of human and electronically readable media might support this. Nevertheless, this approach

[28]For example hard-disk in contrast to paper-based documents.

has two intrinsic disadvantages:

- The distinction between human and electronically readable media is often not clear. Paper-based documents are rather readable by a human but might also be digitalised and transformed into an electronic document. Current OCR-technologies[29] improve the quality of the electronic document by converting the digital bitmap into an modifiable text.
- Semantic differences between different document or format types are not covered by this approach. They usually result in equal problems like clashes based on media.

The current conceptualisation of the resource modelling language contains only the explicit modelling of media. This first —but restricted— conceptualisation will be evaluated in future research. Results of this research will point out the advantages and/or the inadequateness of the general distinction between human and electronically media.

Also the distinction between persistent and transient media reflects only an experimental approach for the classification of media. The basic idea behind such an approach is, that persistent media represent a durable storage for information. In contrast to this transient media do not store information permanently, but hold them for a short period of time. Information is only contained on a transient medium as long at is delivered to its destination (equally a persistent storage). Examples for transient media are transportation channels like local area network connections or long range data transmission infrastructures. Information is only contained in such a medium for the purpose of delivering it from its origin to its destination. Part of such a transportation process might also be the buffering of data on some nodes which are part of the transportation system. The basic idea behind the reflections on network channels[30] is, that they can be regarded as media but they do not store information permanently. Similar concepts can be found beyond information systems: Goods (physical products) are moved from a source (e.g. a manufacturer) to a specific destination (e.g. a wholesaler). Those goods are actually not stored[31] but they have a given location and the transportation process usually takes time. Regarding to both —data transmission over a network connection or the forwarding of physical products— the transportation process tainted with the consumption of

[29] Optical Character Recognition
[30] and equally like transient media
[31] like in a warehouse

time[32], costs[33] and different locations. Therefore, we treat transportation channels in our first approach as medium —especially a transient medium. Further research will prove our assumptions or constitute them as useless.

2.4 Software Types

Software is a special kind of an intangible resource[34]. Like information, software is not a physical entity and depends on a containing physical medium . Such a medium can be physical data medium (e.g. floppy disk, CD-ROM), a hard disk drive of a computer on which the software is installed, or the main memory of a computer on which the software is running. A software-type is an abstract classification of different kinds of software[35].

Compared to information, software supports the execution of tasks. An information-type mainly consists of a static structure description. Information is an instance of an information-type containing a specific content satisfying the structure definition of its type. Functional aspects are not described by an information-type but by a software-type. Software is some kind of an active intangible resource which supports the processing of information and business tasks.

The basic meta-model of the software-type and its associated resource-types is presented in figure 5. A `SoftwareType` is able to process electronically readable document-types (i.e. information-types of a specific `DataFormatType`) and relies on an electronically readable medium. The data-format-type consists of a character-string expressing the name of the format-type, the version of the format-type, and the structural definition of the format-type. The first two attributes identify the kind of the data format and the last one contains the structure-definition basing on the kind of the data format. For example a text-document of a specific format-type of is characterised by its `formatType` (Microsoft Word), `formatTypeVersion` (Word 97), and its `formatTypeDefinition` (structure of Word documents). Hence, a `SoftwareType` is associated with every available format-type[36] it

[32]Data transmission is very fast in contrast to transport of physical object. But nonetheless, there is some amount of time spent for the transportation.

[33]The transportation of information and goods as well is correlated to some amount of money.

[34]refer paragraph 2

[35]Examples for software-types are database management system, text-processor, application-server, or data-warehousing-software.

[36]A modeller should always associate only known and appropriate format types with a software package. Approriate format types corresponds to alls formats which fits the semantic level of an application. As a worst case, a text editor might be used for the editing of any file —neglecting the data type. Nevertheless, we emphasize the usage of a specialized editors for the modification of different data objects.

Figure 5: Software Types

is able to process[37].

Software depends on the availability of a data medium. This relation is modelled by the association between `SoftwareType` and `ElectronicallyMediumType` in figure 5. Note, that this association only addresses the availability of software on a medium. The installation of software on a computer and the availability as active resource (i.e. the software running on a computer) is not covered by this specification. This aspect will be covered in future versions of the resource modelling language.

Open Research Questions: The classification of software as an intangible resource seems to be odd in the context of knowledge-management. Intangible resources are usually not describable by a formal language; but, software is. Classical intangible resources are knowledge, qualification or

[37]This rather vague example bases on the assumption, that only the textual information of MS-Word document is of a certain relevance. Embedded figures and spread-sheets are left out. They only represent information (or just their representation) originally not included in the document —but associated with it.

human experience. Those aspects are hard to describe with any given language. In contrast to this, software is written in a programming language. But, software itself depends on a storage medium[38] and can only be executed by a given computer platform[39]. From our point-of-view, it is not clear wether software is intangible or not. Ongoing evaluation of the conceptualisation of the resource modelling language and its practical application will show advantages and limitations of this approach.

3 Resources

Resource-types represent general properties of resources which are needed for the execution of business process. Examples for general resource-types are trucks, computers, a DBMS, and consultants. These resource-type do not consider specific products manufactured or distributed by a dedicated company or a specific incarnation of a product type. A resource represents a dedicated instance of a specific resource type. A truck produced by a specific manufacturer is a vehicle resource of a truck type. A DBMS distributed by a software manufacturer is of the type DBMS.

An example for different levels of abstractions of a database-management-system is presented in figure 6. A database-management-system is a general resource-type (`DBMS:SoftwareType`). Such a type only expresses the need for a software-type for the management of persistent data. A concrete resource is represented by IBM's database-management-system DB/2 (`DB2:Software`). The resource type of this DBMS is `SoftwareType`. DB/2 is an instance of a DBMS distributed by IBM. A specific instance of DB/2 running on a dedicated machine is represented by `myDatabase`.

With respect to conceptual modelling concrete instances of specific types (like the single installation of a specific DB/2-license) will be neglected. We will only discuss resources and resource-types. Resource-types have already been considered in paragraph 2. Hence, this section will deal with resources.

3.1 Basic Resources

The conceptualisation of basic resources according to resources types (as discussed in section 2) is presented in figure 7. The hierarchy of resources is equivalent to the one of resource types in figure 2.

Resource types specify properties of kinds of resources and abstract from

[38] For example a persistent medium like a hard-disk or a CD-ROM.

[39] A given software-package can only be executed on a specified operating system running on a given processor architecture.

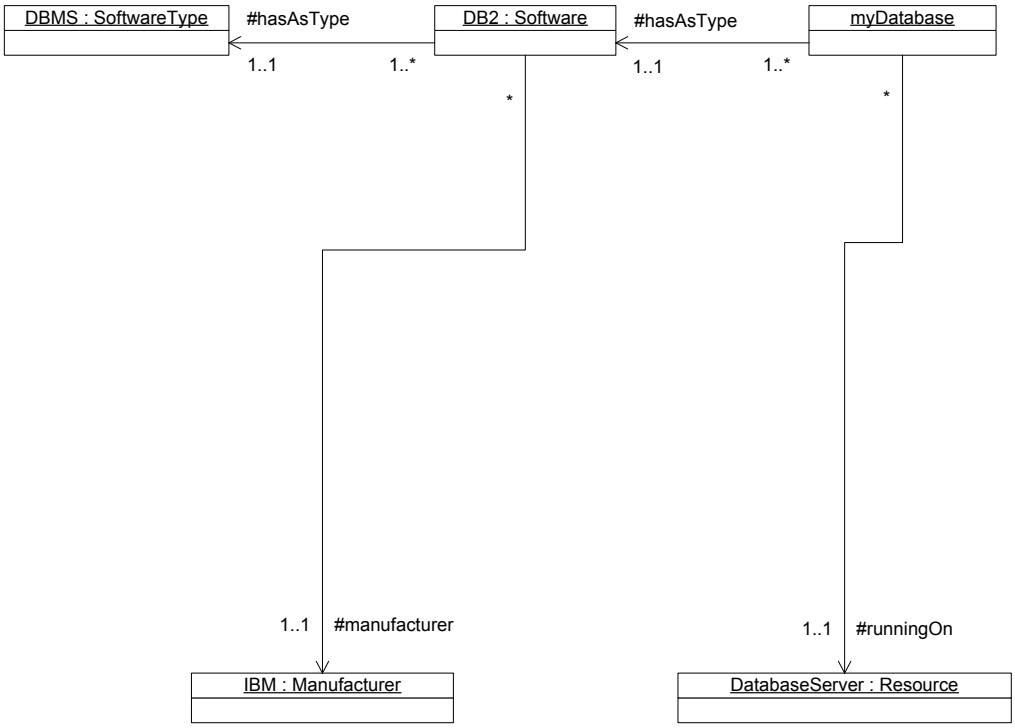

Figure 6: Type, Resource and Instance

concrete resources. In addition to this, resources represent concrete products (in a manner of resources).

3.2 Resources and Types

Regarding to different levels of abstraction, resources can be viewed from several points of view. A resource might be a seen as a resource type (e.g. database server), a resource (Oracle) or a concrete resource instance (a specific Oracle DBMS license or installation). The relationship between resources and their corresponding types is established by a `hasAsType`-relationship. Examples for intangible resources are given in Figure 8. An installed program with an assigned license number is related to a software type, represented by the name and version of the software package by the `hasAsType`-relation. Also information such as the instance of an invoice is related to the invoice type.

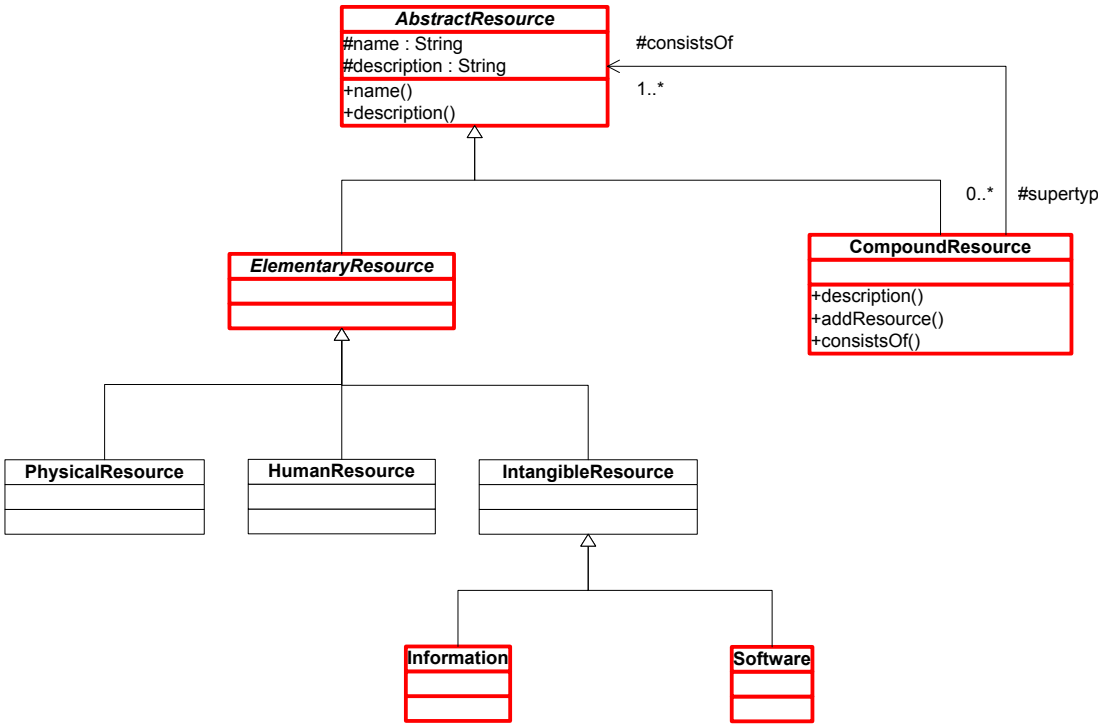

Figure 7: Basic Resources

Open Research Questions: The semantics of the `hasAsType`-relationship has to be specified by the resource modelling language. The need for this specification has already be identified as a conclusion from the different levels of abstraction presented in section 1.3. But, the way of specifying such a relation has not yet been elaborated. On the one hand, such a relation offers the flexibility for the description of resources on different levels of abstractions[40]. On the other hand, it raises the complexity of a resource modelling language by the provision of several interrelated concepts. It has to be checked wether a suitable complexity will be needed and how it can be hidden from the user of the language. An accurate specification of the language or specially designed user interfaces seem to be a possibility for the encapsulation of specific complex language features.

[40]Such levels might comprise resource types, resources and resource instances as outlined in section 1.3

Figure 8: Resources and corresponding Resource Types

3.3 Variants

A variant of a resource represents a variation from the original resource. It mainly conforms to the original resource but adds some extensions and a slight difference in its name.

Example: The original resource is a scanner-model produced by a specific manufacturer (let us call it Hewlett-Packard). The same scanner type might be distributed with an additional equipment for the digitalisation of dias under a modified product name. Hence, both products base on the same scanner type and differ only in the additional equipment and the slightly modified product name. Most of the technical specification like resolution, scanning speed, physical dimension are equal to both products because they rely on the same basic product type. Differences are only caused by an additional feature (the equipment for the digitalisation of slides, which might also be attached to the original product as an option) causing an extended scope of delivery and a modified product name.

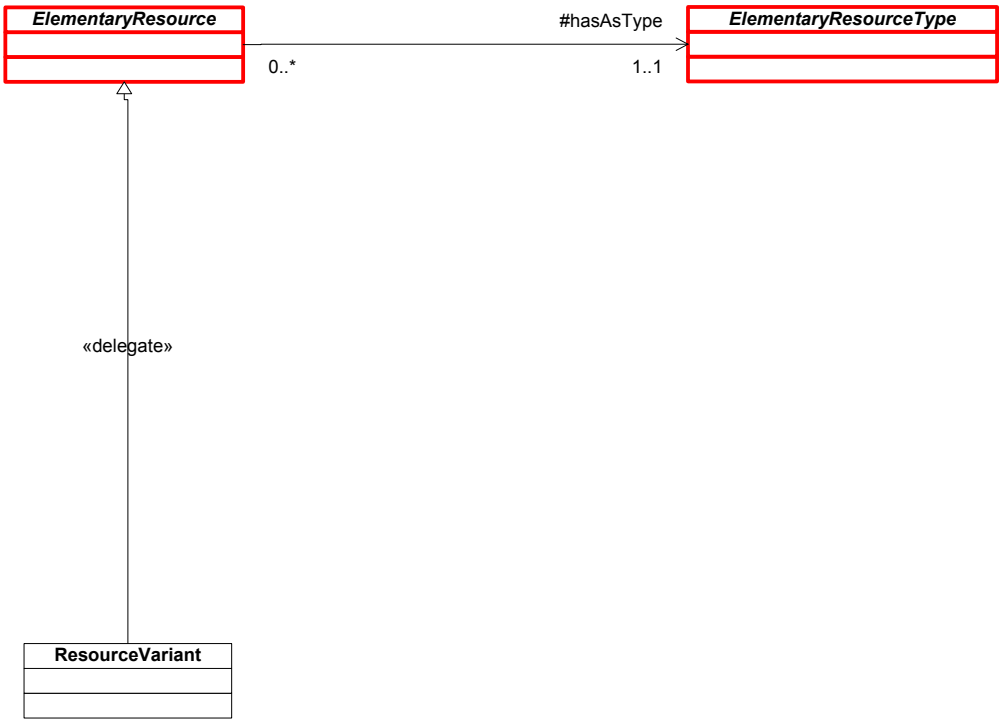

Figure 9: Variant of a Resource

The consideration of variant as an extension to already available resources is presented in figure 9. A specific elementar resource (`ElementarResource`) has an associated type (`ElementarResourceType`) and might have some variants. Such a variant (`ResourceVariant`) might add some features to the elementar resource and change its product name. Generally speaking, a variant adds some additional information to an existing resource. Most information is adopted by a variant and extended by additional information. Usually the product name is modified with respect to the changes.

Such an extension of a resource leads to the interpretation that a variant is equal to a known resource type (with respect to some additions). Most properties of the resource and the variant are the same.

4 Accounting of Resources

From an economic point-of-view, resources are usually associated with costs of their corresponding usage within a business process. The way in which

resources are allocated to tasks is very important to the efficiency of a workflow. Hence, a resource modelling language has to consider economic aspects of the resource usage. Regarding different business processes and resource types, costs can be determined differently. A differentiation of costs for any kind of resources is given in figure 10.

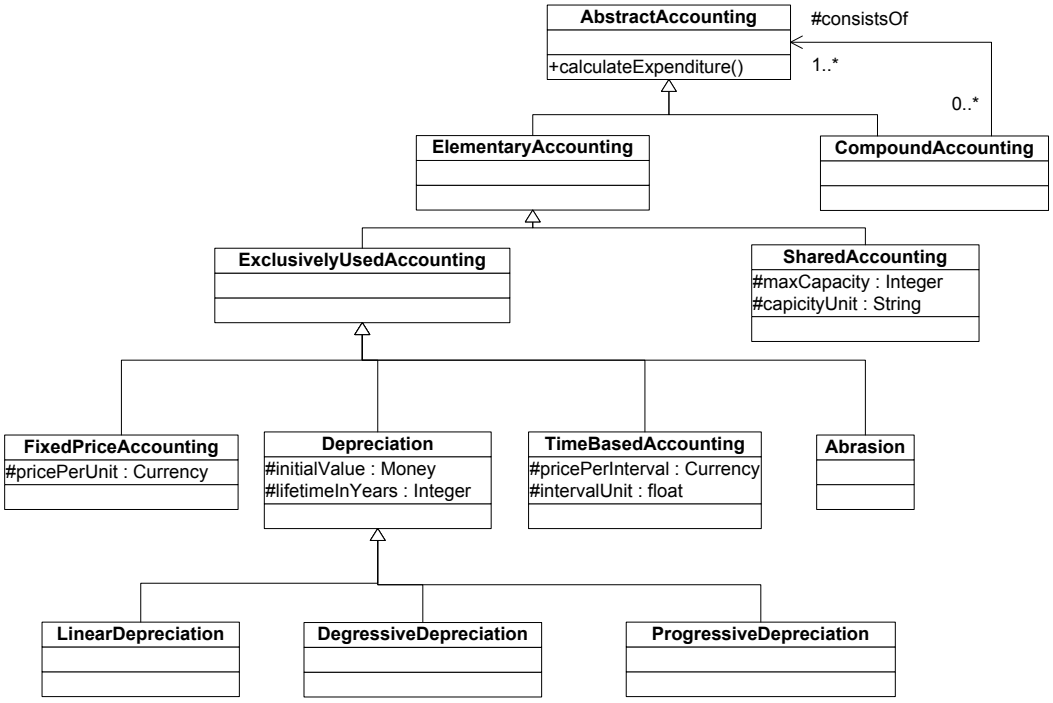

Figure 10: Basic Accounting

A resource can at any given time be used within several processes or just one. If it is shared between processes its availability is restricted by a maximum capacity; represented by a capacity unit and an upper bound of numbers of that unit. The usage of such resources in different processes has to be specified by the amount of units required for that specific process. A database server is an example for a shared resource. The work load of a database server is often specified by a maximum number of simultaneous transactions. Thus, the capacity unit is 'transaction' and the maximal capacity is the highest number of transactions at a given point of time.

Costs of exclusively used resources can be interpreted differently - depending on the kind of resource use (see 11). Resources are accounted for on a fixed price per unit or on a time base. Another possibility is the assignment

of abrasion or depreciation for the use of a resource. Fixed prices usually form the calculation base for spare parts as well as operational and raw material. Some staff and machinery may also be calculated for by fixed price. Additionally, those resource types are characterised by their abrasion or depreciation, while used for a process. Hence, the kind of accounting is orthogonal to the type of a resource.

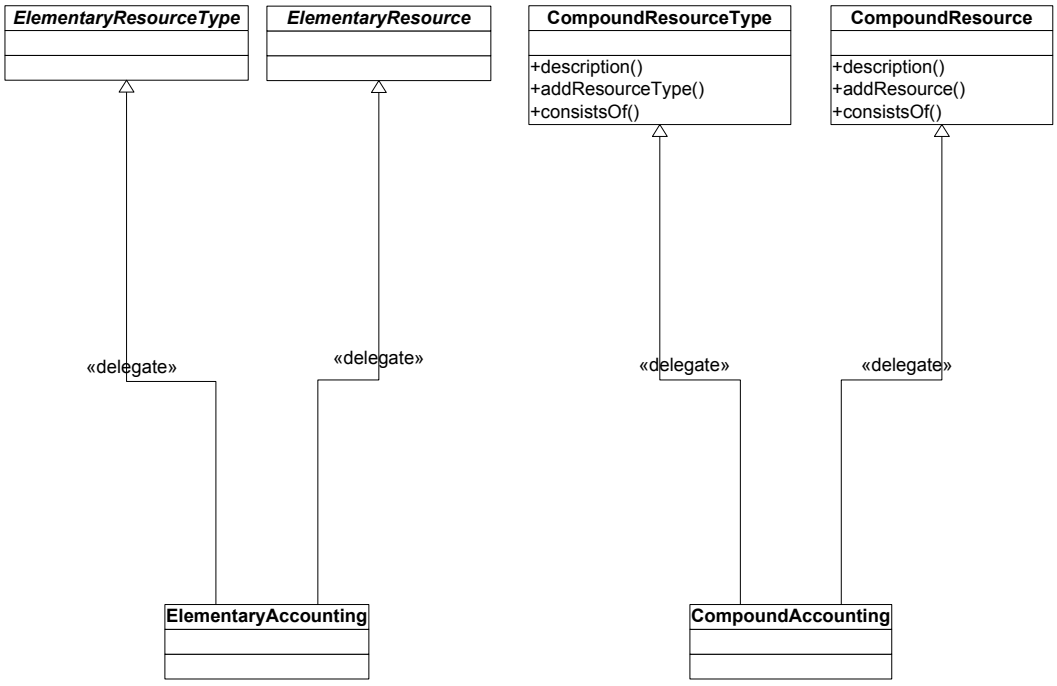

Figure 11: Assignment of Accounting Information to Resources and Types

As stated in section 1.3, resources can be seen from different points-of-view. The basic framework for the specification of resources and resource types follows a conceptional point-of-view and economic features were deferred for the moment. The core problem lies in different kinds of classification of resources. Two major typing schemes discussed in this paper are the conceptual and the economic classification. There are other kinds of classification like availability, quality, or responsibility. Hence, we should consider a multiple classification of resources and resource types. Additionally, the economic classification of a resource might change in time or between different business processes. A fixed-price calculated resource of one specific process might be accounted on a time base in another process. In consequence, the classification from an economic point-of-view can be dynamic.

Odell discusses multiple and dynamic classification in [31].

Different perspectives for the classification of resources in the resource modelling language are realised by a concept called *delegation*[41]. Taivalsaari defines delegation as follows:

> **Delegation** is a general-purpose *sharing* mechanism. It requires the presence of the [...] self-reference, and operates by forwarding messages from one object to another *without changing the self-reference*. [38, page 66]

Taivalsaari's point-of-view is obviously influenced by object-oriented programming languages. He explains delegation as forwarding messages and refers to the self-reference. But, the core of its definition addresses a general sharing mechanism of different concepts (abstractions) which are conceptually unified in one object. This object might fill different roles, depending on the context.

The primary classification of resources bases on a domain expert perspective as presented in Basic Meta-Model of the Resource Specification Framework. Other perspectives - such as accounting - are attached to resources by delegation (see Figure 11). Delegation means, that a resource might satisfy a certain perspective by playing a special role. The role of an elementary resource type (ElementaryResourceType in Figure 11) is represented by an elementary accounting. This elementary accounting encapsulates all economic properties of a resource and replaces the resource in every economic context. All non-economic aspects of a resource within an economic context are represented by the resource (type) itself. Hence, the elementary accounting represents a resource whenever accounting properties are requested and delegates all other inquiries to the resource itself.

5 Interfaces to other MEMO-Languages

The major objective of this paper is the specification of a conceptually adequate resource specification language. Hence, the core business lies in a semantically adequate evaluation of resource types and their mapping to an appropriate language. Nonetheless, this language will be integrated into the existing modelling language MEMO-OrgML. Therefore, the resource modelling language has to consider existing language features of the existing organisation modelling language. These aspects will be considered by

[41] The concept of delegation is presented in [14] from a conceptual point of view. The authors also propose a prototypical implementation in Smalltalk.

the definition of the assignment of resources to organisational units (see paragraph 5.1) and processes (paragraph 5.2).

5.1 Human Resources

Organisational structures can be expressed by well-established organisational charts. Such a chart contains organisational units and their relationships. There are —at least— two different types of relationships:

- aggregation
- authority

An *aggregation relationship* describes the composition of organisational units to a composed unit. Hence, all subunits of a composed organisation unit are parts of it. An organisational unit which cannot be decomposed is called a position. A position might be filled by a specific employee. From this point-of-view, positions represent abstract members[42] of an organisation and an organisational unit is some kind of grouping of these members. Superior aggregated units are transitively composed of other aggregated units (e.g. groups, departments, and divisions). *Authority relationships* define superior-subordinate-relationships. Not the aggregation of organisational units but superior units are modelled. Superior units are authorised to give orders to subsidiaries and have generally to be involved in decision making processes. Following such an approach for organisational charts, organisational units with subsidiaries can be represented by persons, too. Additionally, both approaches can be used simultaneously within one organisational chart.

Obviously, organisational units represent human resources for the tasks an organisation has to do. Those tasks can be mapped to processes of a business process model. Organisational units (especially those filled by staff member) can be associated with resources the same way. The specification of organisational structures is usually not a core feature of resource modelling. Hence, a resource modelling language may not include an organisation structure language (like organisational charts). But it should support the usage of such a language by adequate interfaces. The MEMO-OrgML already includes a language for organisation structures. Consequently, the resource modelling language will not include its own organisation structure modelling language but an interface to the existing one.

[42]The term 'abstract member' is used to emphasise, that a position must not be directly correlated to a specific person. In fact, a position keeps constant over time while the person filling a position might change.

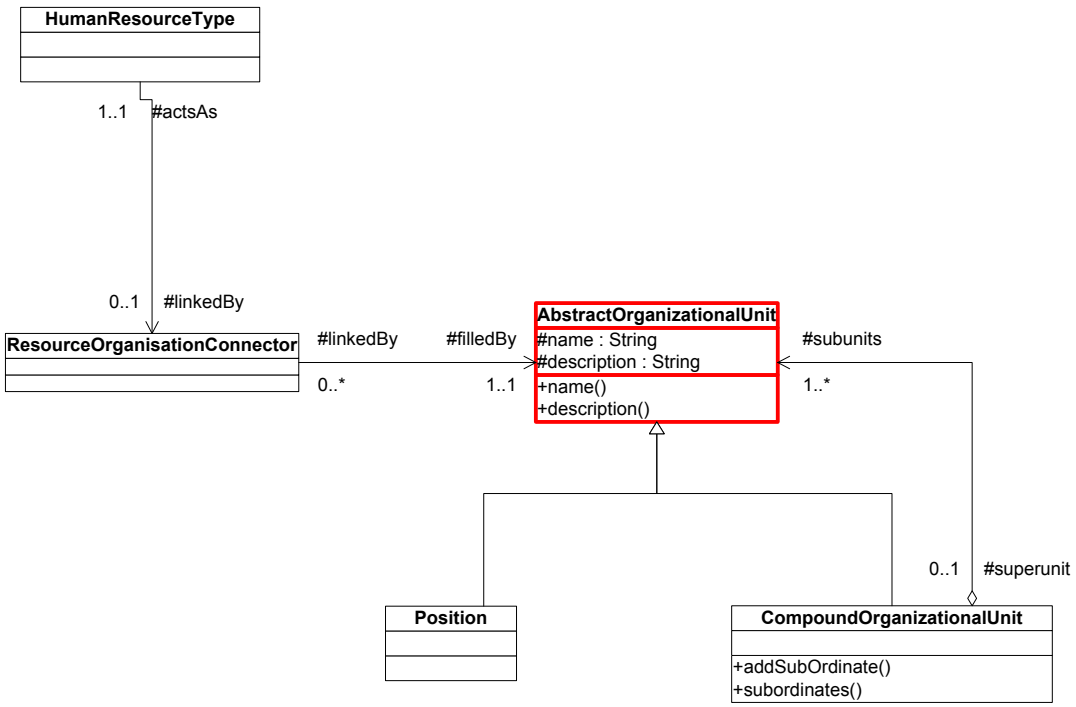

Figure 12: Human Resources assigned to Organisational Units

A first approach to the integration of the resource modelling language and organisation modelling language of MEMO-OrgML is presented in figure 12. Organisation structures are regarded on a very abstract level. An organisational unit (`AbstractOrganisationalUnit`) is either a composed unit (`CompoundOrganisationalUnit`) or a position. Each composed unit consists of one or several other organisational units. A position does not have any subunits. A human resource type is assigned to a member of the organisation by a special linking object of the type (`ResourceOrganisationConnector`). This connector represents an interface between the concept of `HumanResourceType` of the resource modelling language and an associated organisational unit.

Open Research Questions: This very first approach for the mapping of resources to organisational units (and vice versa) only indicates the future integration of resource and organisation modelling in MEMO-OrgML. As far as the language for the description of organisation structures has not been specified in detail, we assume the simple conceptualisation as presented in 12. Further changes will not affect the resource modelling language but only the class `Role`.

5.2 Allocation of Resources

A single business process only describes what has to be done on a business object (a physical product or an information object). Additionally the assignment of resources to a process specifies the subject (*who or what* will work on an object) and what additional objects (i.e. passive resources) are needed. But the assignment of resources or resource types to a business process is more than a simple attachment of a resource to a process. Furthermore, the extend of the usage of a resource in a corresponding process has to be defined.

One approach for the allocation of resources to processes[43] is described by Fowler's *Resource Allocation Analysis Pattern* [10]. This pattern (refer to figure 13) introduces an abstraction called *Resource Allocation* that specifies the amount of a resource's usage in a business process. This abstraction is represented by the class `ResourceAllocation` in figure 13. A resource allocation is differentiated by a general and a specific resource allocation. A general resource allocation may only be applied to a resource type and a specific resource allocation to a resource. Specific resource allocations are specialised to `TemporalResource` and `Consumable`. Consumable resources are used up by the execution of a process. They will consequently not be available after the process' termination. Generally speaking, consumable resources are consumed by a process. Temporal resources are used for the execution of a process but will be available afterwards (except of attrition). Resource types are specialised to `ConsumableType` and AssetType— accordingly. A temporal resource allocation only corresponds to an asset which is of the type `AssetType`. A holding corresponds to a consumable allocation as well.

The resource allocation pattern by Fowler promises a valuable approach for the specification of a resource's usage within a process. It separates aspects of a specific usage from the concrete resource itself. Also the annotation of an adequate quantity and the specification of periods for time-based allocated resources introduces a valuable approach.

But some aspects of this pattern are not convincingly elaborated. The differentiation between resource and resource types only addresses the usage and the booking of a resource. It is not clear at the moment whether this differentiation fits to our perspectives on instances and types. Quantities for the resource allocation are only discussed on abstract level by Fowler. He does not specify concrete types of quantities for different kinds of usages of resources. The usage of a resource is only restrictively reducible to quantity-based and time-based allocations. Additional factors as attrition or depreciation (refer to 4) have to taken into account. Hence, basic ideas of

[43] Other approaches have to be identified and evaluated in future.

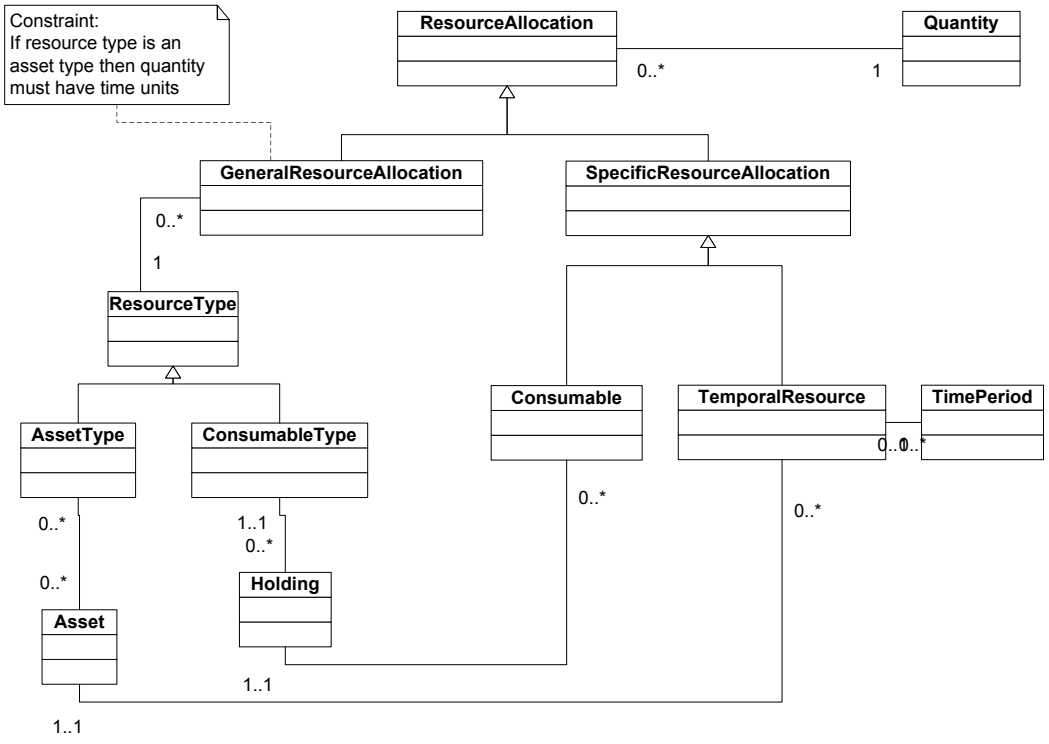

Figure 13: Resource Allocation Analysis Pattern by Fowler

the resource allocation pattern seem to be a valuable foundation of resources in the MEMO process-modelling language. But additional observations on this pattern and the evaluation of other resource allocation techniques will be needed.

6 Support Types

Additional supporting types, which are not used in the context of resource modelling only, are presented in this section. Those types are used for the specification of resources and resource types in the context of this research report. A financial amount[44] (refer to section 4) describes the amount of money spent for the usage of a resource for the execution of a specific process. Also weights and measures (as discussed in section 6.2) are provided in the resource modelling language on a semantic adequate level of abstraction. Furthermore, those types are not only relevant for resources

[44]represented by a finacial value

and resource types. Financial values might also be valuable for other kinds of applications within different contexts. A money-representing type like the financial-value-type is also applicable to product catalogs, accounting systems, or data-warehousing applications. Also Weights and measures are relevant in a huge set of software systems. Hence, those types are designed to be used in the resource modelling language but their design is as general as possible that they can be (re-)used in other contexts, too.

6.1 Financial Values

A financial value[45] describes – in the context of this resource modelling language – the amount of money spent on the usage of a specific resource for the execution of a certain (business) process. The kind of accounting of a resource (type) assigned to a process is specified by an accounting type as presented in section 4. Independent of the accounting type, one important parameter of the accounting calculation is of the kind of financial value. The result of an accounting calculation is of the type `FinancialValue`, too.

A financial value is characterised by an amount of money and the currency of this money. From an object-oriented point-of-view, a financial value consists of an amount, a currency and a set of methods applicable to a financial value. The methods define operations, which are provided by a financial value. Those operations include arithmetic (add and subtract) and boolean (isEqual, isSmaller, and isBigger) operations as well as conversion services (conversion to another currency). A proposal for the conceptualisation of a financial value type is displayed in the class diagram in figure 14.

Every financial value is characterised by a monetary amount and a currency. The amount is of the type `Decimal` with a precision of two digits after the decimal point. The currency of a financial value is assigned by an associated object of the class `Currency`. This annotation of the currency defines the real monetary value of an amount.

Some desirable functions on financial values address arithmetic operations, boolean comparison, and conversion. The class diagram in figure 14 displays some examples of such functions. The methods `add` and `subtract` support the addition and subtraction of one financial value to/from another. Especially different currencies have to be taken into account by such an addition/subtraction. Hence, amounts of financial values of different currencies have to be converted to one common currency. Regarding the methods `add` and `subtract`, the common currency and the currency of the result is the one of the receiver of the message.

[45] A financial value represents a certain amount of money of a specific currency.

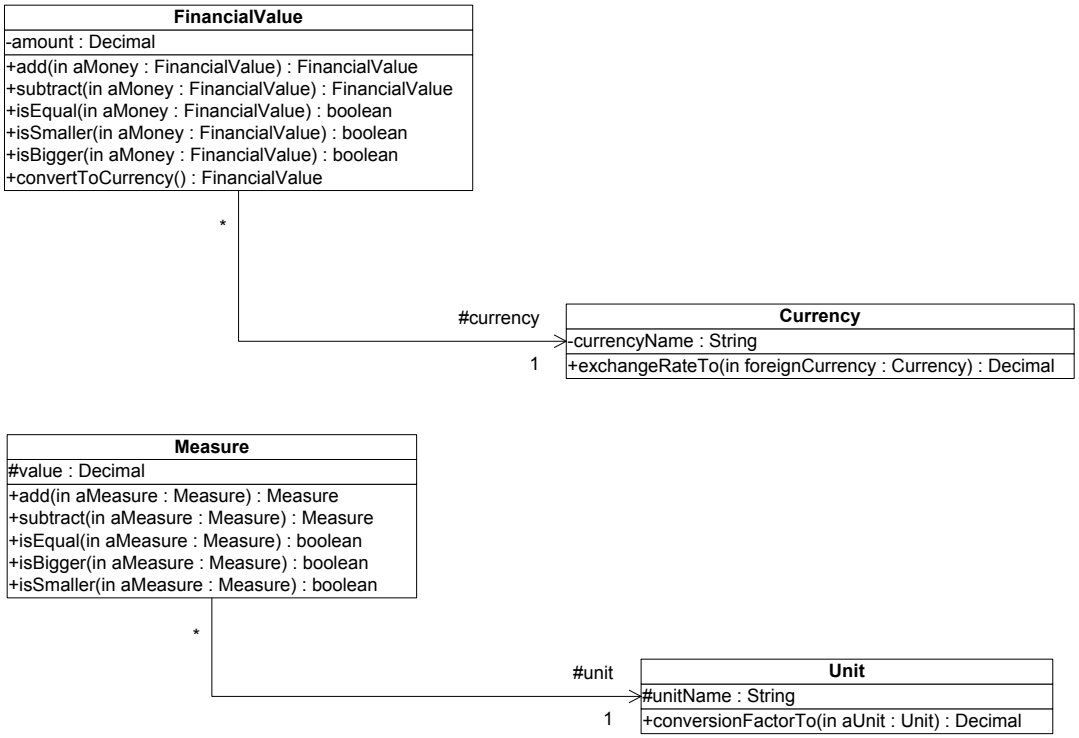

Figure 14: Support Types: Finance and Measure

Example: `(23.00 Euro).add(7.50 A$)`

The parameter `7.50 A$` is converted to the corresponding financial value in Euro and is then added to `23.00 Euro`. The currency of the resulting financial value is `Euro`, too.

The same conditions hold for boolean operations. These operations include comparisons, whether a financial value is equal, smaller or larger than another. Those operations return a result of the kind of `boolean` but use the same conversion scheme as the arithmetic operations. If the currency of the parameter is different from the currency of the receiver, the amount of the parameter is converted to the one of the receiver.

A conversion of the amount of a financial value depending on different currencies is done by the method `convertToCurrency` of a financial value object. This method determines the exchange rate for the conversion from one currency value to another by using the `exchangeRateTo`-method of the currency-object of the receiver.

6.2 Quantities

The semantic power and flexibility the class `FinancialValue` in connection with the `Currency`-class seems to emphasise the assignment of this concept to other quantities. Measures (and also weights[46]) consist of a value and an associate unit. As a currency specifies the financial value of an amount, a unit-type might stand for the dimension of a special measure-object. A first conceptualisation of this fact is presented in figure 14.

The value of a measure represents a general value corresponding to its unit. Arithmetic and boolean operations might also be applicable to measures. But not all kinds of measures are comparable. A distance in meters is comparable to a distance in kilometers or miles because they are all specified by distance units. A rectangle is also comparable to the product of two distances and the volume of a cube is comparable to the multiplication of three distances or the multiplication of a rectangle and a distance. Generally speaking, their are some kinds of measures which are comparable to each other. But this does not hold true for all kinds of measures. Timestamps are not comparable to temporal durations. Time is usually not compatible with spatial data or weights. Hence, their are some kinds of mutual comparable measures and other incompatible dimensions. A general specification of a measure-type might reflect some common properties but excludes relevant conceptual differences. Hence, a holistic applicable specification is hard to be done; against all common properties such as the existence of a value, a unit and the need for arithmetic and boolean operations.

7 Example: Information Resources

The application of the resource modelling language is demonstrated by an example given in figure 15. An assembly list of a specific product type and its according data format types are presented. Additionally, a database server running a DBMS working on the assembly list are modelled. An assembly list is a list of all subparts of a physical product. Each subpart may either be an elementary or a complex part specified by its own assembly list. The assembly list in figure 15 is of the type `InformationResourceType` and therefore specified by a structure definition. This definition (`structureDefinition` in the rounded box labelled `AssemblyList:InformationResourceType`) is formulated in an abstract manner using the Extended Bacchus-Naur-Form (EBNF). The assembly list is stored in a corporate information system using a relational database and an XML document. The relational schema is represented by the data format

[46]The term measure stands synonymously for measures and weights within this paper.

type RDB and the document type definition expressed in XML. Both electronic incarnations of an assembly list are processed by a DBMS running on a dedicated server Server of the kind ElementarResourceType. Benefits of such a modelling of the assembly list are the documentation of document structures, association to required resources, and the specification of exchange formats.

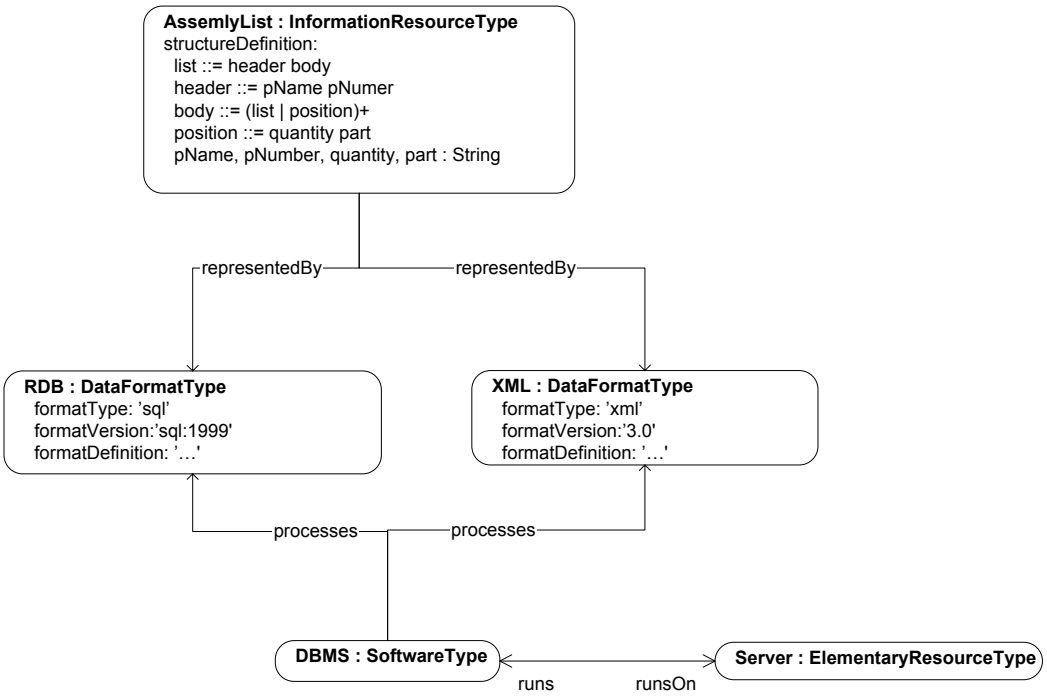

Figure 15: Example for the Usage of Resource Types

An abstract definition of the structure of an information type fosters communication on this information type. Different communication partners can relate their discussion about information types on a common definition. The specification of the assembly list might act as a reference for engineers and business administrators. It is also a specification for external partners producing subparts. The association with data formats like RDB or XML leads to a correspondence of an assembly list to computerised formats. These formats are not only used within a process but might also be used for the interaction with other information systems. Such information systems include other corporate information systems and systems of external partners as well. Data format type definitions are, thus, used as an interorganisational reference system. They specify interfaces for the exchange of data.

Associating resource types to an information type supports the allocation of resources to a process. Some processes rely on the availability of information. Information classified as an intangible resource will not be available directly. There are other resources providing the required information. These might be media containing information or an elementar resource type providing information. Information systems in a broader sense fulfil this requirement. Hence, systems allowing the access to information are an important resource. Modelling these resources specifies the way of accessing of needed information.

8 Concluding Remarks and Future Work

This research report summarises some basic ideas for the conceptualisation of resource modelling language. Some core features of this language have been described and some other features have been mentioned. A lot of open research question are pointed out and will be subjects of future research. The hereby described features of a resource modelling language will be evaluated — in cooperation with other research groups and projects — while modelling resources of different domains. Goal of this evaluation are the verification of the specified concepts and the determination of possible ambiguities in the language specification. Those ambiguities will be fixed in future versions of the resource modelling language. Furthermore, the modelled resources and resource types will enter the resource modelling language. They will be assorted into several domain-specific packages and offered to the user. We are planning to develop resource-packages for logistical processes, project management and IT-resources.

Up to now, tool-support for the modelling of resources using the resource modelling language is not available. An appropriate resource modelling tool will be developed after the end of the evaluation phase. This tool will be integrated into the process modelling framework of the MEMO-OrgML. It supports the definition of resources and resource types as well as the management of existing resources and resource packages.

References

[1] van der Aalst, W.; van Hee, K.: "Workflow Management - Models, Methods, and Systems." Cambridge (Massachusetts), London (England): MIT Press, 2002

[2] Baumgarten, B.: "Petri-Netze: Grundlagen und Anwendungen." Heidelberg, Berlin, Oxford: Spektrum Akademischer Verlag, 2. Auflage, 1996

[3] Bennett, S.; McRobb, S.; Farmer, R.: "Object-Oriented Systems Analysis and Design using UML." McGraw-Hill, 1999

[4] Bergholtz, M.; Johannesson, P.: "Validating Conceptual Models - Utilising Analysis Patterns as an Instrument for Explanation Genertion." In: Bouzeghoub, M. et al. (Eds.): NLDB 2000, LNCS 1959, Springer, 2001, pp. 325-339

[5] Chrystal, K.; Lipsay, R.: "Economics for Business and Management." Oxford University Press, 1997

[6] Curtis, B.; Kellner, M.I.; Over, J.: "Process Modelling." In: Communications of the ACM, September 1992/Vol. 35, No. 9, pp. 75-90

[7] Diederich, H.: "Allgemeine Betriebswirtschaftslehre." 7th edition, Kohlhammer, 1992

[8] Eertink, H.; Janssen, W.; Luttighuis, P.O.; Teeuw, W.; Vissers, C.: "A Business Process Design Language." In: Wing, J.; Woodcock, J.; Davies, J. (Eds.): FM '99, Vol. I, LNCS 1708, Springer, 1999, pp. 76-95

[9] Eriksson, H.-E.; Penker, M.: "Business Modeling with UML - Business Patterns at Work." Wiley, 2000

[10] Fowler M.: "Analysis Patterns - Reusable Object Models." Menlo Parc (California) et al.: Addison-Wesley, 1997

[11] Frank, U.: "The MEMO Meta-Metamodel." Research Report of the IS Research Institute, University of Koblenz, No. 9, 1998

[12] Frank, U.: "MEMO: Visual Languages for Enterprise Modelling." Research Report of the IS Research Institute, University of Koblenz, Nr. 18, 1999

[13] Frank, U.: "Organising the Corporation: Research Perspectives, Concepts and Diagrams." Research Report of the IS Research Institute, University of Koblenz, No. 25, 2001

[14] Frank, U.; Halter, S.: "Enhancing Object-Oriented Software Development with Delegation." Research Report of the IS Research Institute, University of Koblenz, No. 2, 1997

[15] Frank, U.; Jung, J.: "Process Modelling with MEMO-OrgML" To be Published as a Research Report of the IS Research Institute, University of Koblenz, 2002

[16] Frank,U.; van Laak, B.: "Anforderungen an Sprachen zur Modellierung von Geschäftsprozessen." To be Published as a Research Report of the IS Research Institute, University of Koblenz, expected 2002

[17] Fraunholz, B.; Jung, J.: "Evaluation of Mobile Applications - software-technical and human aspects." In: ...: Proceedings of the International Conference on Software Evaluation, Paris, 2002, pp. ...

[18] Green, P.; Rosemann, M.: "An Ontological Analysis of Integrated Process Modelling." In: Jarke, M.; Oberweis, A. (Eds.): CAiSE '99, LNCS 1626, Springer, 1999, pp. 225-240

[19] Gudehus, T. : "Logistik: Grundlagen, Strategien, Anwendungen." Berlin et al.: Springer, 1999

[20] Herbst, H.: "Business Rule-Oriented Conceptual Modeling." Physica-Verlag, 1997

[21] Hiles, A.; Barnes, P.: "The Definitive Handbook of Business Continuity Management." John Wiley and Sons, 2001

[22] Jung, J.; Frank, U.: "Konzeption der Architektur eines Flottenmanagementsystems im Kundendienst." In: Grünert, T.; Sebastian, H.-J.: (Hg.): "Logistik Management - Supply Chain Management und e-Business." Stuttgart: Teubner, 2001, S. 283-292

[23] Jung, J.; Fraunholz, B.: "Resource Modelling in an object-oriented Process Modelling Language." To be Published in: Procedings of the Australian Workshop on Requirements Engineering, AWRE2002, Melbourne (Australia), 2002

[24] Jung, J; Kirchner, L.: "Logistische Prozesse im Handwerk - Begriffliche Grundlagen und Referenzmodelle" Research Report of the IS Research Institute, University of Koblenz, Nr. 29, 2001

[25] Koubarakis, M.; Plexousakis, D.: "A Formal Model for Business Process Modeling and Design." In: Wangler, B.; Bergman, L. (Eds.): CAiSE 2000, LNCS 1789, Springer, 2000, pp. 142-156

[26] Marshall, C.: "Enterprise Modeling with UML - Designing Successful Software through Business Analysis." Addison-Wesley, 2000

[27] Miller, B.B.: "Managing Information as a Resource." In: Rabin, J.; Jackowsky, E.M. (Eds.): Handbook of Information Resource Management, Marcel Dekker, 1988, pp. 3-33

[28] zur Muehlen, M.: "Resource Modeling in Workflow Applications." In: Becker, J.; zur Mühlen, M.; Rosemann, M. : Workflow Management 99.

Proceedings of the 1999 Workflow Management Conference. Muenster, Germany, November 9th 1999, pp. 137-153.

[29] Nübel, H.: "The resource renting problem subject to temporal constraints." In: OR Spektrum (2001) 23, pp. 359-381

[30] Oberweis, A.: "Modellierung und Ausführung von Workflows mit Petri-Netzen." Stuttgart, Leipzig: Teubner, 1996

[31] Odell, J. J.: "Dynamic and multiple classification." In: Journal on Object-Oriented Programming, January 1992, pp. 45-48

[32] Österle, H.: "Business Engineering: Prozess- und Systementwicklung." Springer, 1995

[33] Podorzhny, R.M.; Staudt Lerner, B.; Osterweil, L.J.: "Modeling Resources for Activity Coordination and Scheduling." In: Ciancarini, P.; Wolf, A.L. (Eds.): COORDINATION '99, LNCS 1594, Springer, 1999, pp. 307-322

[34] Scheer, A.-W.: "Architecture of Integrated Information Systems - Foundations of Enterprise-Modelling." Springer, 1992

[35] Scheer, A.-W.: "Business Process Engineering - Reference Models for Industrial Enterprises." Springer, 1998

[36] Scheer, A.-W.: "ARIS - Business Process Modeling." 2nd edition, Springer, 1999

[37] Sutton, S.M.; Osterweil, L.J.: "The Design of a Next-Generation Process Language." In: Jazayeri, M.; Schaure, H. (Eds.): Software Engineering - ESEC/FSE '97, LNCS 1301, Springer, 1997, pp. 142-158

[38] Tailvalsaari, A.: "A Critical View of Inheritance and Reusability in Object-oriented Programming." Dissertation, University of Jyväskylä, 1993

[39] Wand, Y.; Weber, R.: "A Model of Control and Audit Procedure Change in Evolving Data Processing Systems." In: The Accounting Review LCIV(2), 1989, pp. 87-107

[40] Wand, Y.; Weber, R.: "An Ontological Evaluation of Systems Analysis and design Methods." In: Falkenberg, E.D.; Lindgreen, P. (Eds.): Information Systems Concepts: An In-depth Analysis, North-Holland, 1989, pp. 79-107

[41] Wand, Y.; Weber, R.: "An Ontological Model of an Information System." In: IEEE Transactions on SoftwareEngineering, Vol. 16, No. 11, 1990, pp. 1281-1291

[42] Wand, Y; Weber, R. "Mario Bunge's Ontology as a Formal Foundation for Information System Concepts." In: Weingartner, P.; Dorn, G.J.W. (Eds.): Studies on Mario Bunge's Treatise, Rodopy, Atlanta, 1990, pp. 123-149

[43] Wand, Y.; Weber, R.: "On the Ontological exspressivness of Information Systems Analysis and Design Grammar." In: Journal of Information Systems, Vol. 3, Nr. 2, 1993, pp. 217-237

[44] Weber, R.: "Ontological Foundations of Information Systems." Coopers and Lybrand Accounting Methodology, Monograph No. 4, 1997

[45] Wenzel, J.: "Entwurf einer Modellierungssprache zur Beschreibung von Geschäftsprozessen im Rahmen der Unternehmensmodellierung." Diplomarbeit, Universität Koblenz-Landau, 1997

[46] Zickhardt, J.: "Integrierte Syntax und Semantik einer Objektmodell- und einer Geschäftsprozeßsprache: Eine EER/GRAL- Formalisierung und Semantikbeschreibung in $\mathcal{Z}$ der MEMO-Komponenten OML und PML." Diplomarbeit, Universität Koblenz-Landau, 1999

Bisherige Arbeitsberichte

Hampe, J. F.; Lehmann, S.: Konzeption eines erweiterten, integrativen Telekommunikationsdienstes. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 1**, Koblenz 1996

Frank, U.; Halter, S.: Enhancing Object-Oriented Software Development with Delegation. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 2**, Koblenz 1997

Frank, U.: Towards a Standardization of Object-Oriented Modelling Languages? Arbeitsbericht des Instituts für Wirtschaftsinformatik, **Nr. 3**, Koblenz 1997

Frank, U.: Enriching Object-Oriented Methods with Domain Specific Knowledge: Outline of a Method for Enterprise Modelling. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 4**, Koblenz 1997

Prasse, M.; Rittgen,P.: Bemerkungen zu Peter Wegners Ausführungen über Interaktion und Berechenbarkeit, Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 5**, Koblenz 1997

Frank, U.; Prasse, M.: Ein Bezugsrahmen zur Beurteilung objektorientierter Modellierungssprachen - veranschaulicht am Beispiel vom OML und UML. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 6**, Koblenz 1997

Klein, S.; Zickhardt, J.: Auktionen auf dem World Wide Web: Bezugsrahmen, Fallbeispiele und annotierte Linksammlung. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 7**, Koblenz 1997

Prasse, M.; Rittgen, P.: Why Church's Thesis still holds - Some Notes on Peter Wegner's Tracts on Interaction and Computability. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 8**, Koblenz 1997

Frank, U.: The MEMO Meta-Metamodel, Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 9**, Koblenz 1998

Frank, U.: The Memo Object Modelling Language (MEMO-OML), Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 10**, Koblenz 1998

Frank, U.: Applying the MEMO-OML: Guidelines and Examples. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 11**, Koblenz 1998

Glabbeek, R.J. van; Rittgen, P.: Scheduling Algebra. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 12**, Koblenz 1998

Klein, S.; Güler, S.; Tempelhoff, S.: Verteilte Entscheidungen im Rahmen eines Unternehmensplanspiels mit Videokonferenzunterstützung, Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 13**, Koblenz 1997

Frank, U.: Reflections on the Core of the Information Systems Discipline. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 14**, Koblenz 1998

Frank, U.: Evaluating Modelling Languages: Relevant Issues, Epistemological Challenges and a Preliminary Research Framework. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 15**, Koblenz 1998

Frank, U.: An Object-Oriented Architecture for Knowledge Management Systems. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 16**, Koblenz

Rittgen, P.: Vom Prozessmodell zum elektronischen Geschäftsprozess. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 17**, Koblenz 1999

Frank, U.: Memo: Visual Languages for Enterprise Modelling. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 18**, Koblenz 1999

Rittgen, P.: Modified EPCs and their Formal Semantics. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 19**, Koblenz 1999

Prasse, M., Rittgen, P.: Success Factors and Future Challenges for the Development of Object Orientation. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 20**, Koblenz 2000

Schönert, S.: Virtuelle Projektteams - Ein Ansatz zur Unterstützung der Kommunikationsprozesse im Rahmen standortverteilter Projektarbeit. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 21**, Koblenz 2000

Frank, U.: Vergleichende Betrachtung von Standardisierungsvorhaben zur Realisierung von Infrastrukturen für das E-Business. . Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 22**, Koblenz 2000

Jung, J.; Hampe, J.F.: Konzeption einer Architektur für ein Flottenmanagementsystem. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 23**, Koblenz 2001

Jung, J.: Konzepte objektorientierter Datenbanken - Konkretisiert am Beispiel GemStone. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 24**, Koblenz 2001

Frank, U.: Organising the Corporation: Research Perspectives, Concepts and Diagrams. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 25**, Koblenz 2001

Kirchner, L.; Jung, J.: Ein Bezugsrahmen zur Evaluierung von UML-Modellierungswerkzeugen. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 26**, Koblenz 2001

Botterweck, G.; Hampe, J.: Benutzeroberflächen für WAP-basierte Mobile Commerce Anwendungen. Arbeitsberichte des Instituts für Wirtschaftsin-

formatik, **Nr. 27**, Koblenz 2001

Jung, J.; van Laak, Bodo L.: Flottenmanagementsysteme - Grundlegende Technologien, Funktionen und Marktüberblick. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 28**, Koblenz 2001

Jung, J.; Kirchner, L.: Logistische Prozesse im Handwerk - Begriffliche Grundlagen und Referenzmodelle. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 29**, Koblenz 2001

Frank, U.: Forschung in der Wirtschaftsinformatik: Profilierung durch Kontemplation – Ein Plädoyer für den Elfenbeinturm. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 30**, Koblenz 2002

Jung, J.; Lautenbach, K.: Simulation des Einflusses von Notfällen auf die Auftragsbearbeitung in Handwerksbetrieben. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 31**, Koblenz 2002

Jung, J.: Entwicklung eines elektronischen Fahrtenbuchs - Grundlegender Entwurf, prototypische Implementierung und zukünftige Potentiale. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 32**, Koblenz 2002

van Laak, B. l.: Eine Struktur zur Beschreibung von Prozessmustern der ECOMOD-Prozessbibliothek . Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 33**, Koblenz 2002

Frank, U.; van Laak, B. L.: Anforderungen an Sprachen zur Modellierung von Geschäftsprozessen. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 34**, Koblenz 2002

Jung, J.: A Resource Modelling Language for Business Process Modelling - Basic Conceptualisation, Requirements and Open Research Questions. Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 35**, Koblenz 2002