JÜRGEN JUNG

MAPPING OF BUSINESS PROCESS MODELS TO WORKFLOW SCHEMATA – AN EXAMPLE USING MEMO-ORGML AND XPDL

April 2004

Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 47

UNIVERSITÄT KOBLENZ · LANDAU

Institut für Wirtschaftsinformatik
Fachbereich Informatik
Universität Koblenz-Landau

JÜRGEN JUNG

MAPPING OF BUSINESS PROCESS MODELS TO WORKFLOW SCHEMATA – AN EXAMPLE USING MEMO-ORGML AND XPDL

April 2004

Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 47

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 Wirtschafts- und Verwaltungsinformatik
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 Wirtschafts- und Verwaltungsinformatik
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

Contents

List of Figures

1 Introduction

Business process modelling (BPM) and Workflow Management (WfM) are two popular subjects in the area of information systems research (IS research). On the one hand they both seem to be very similar but on the other hand they concern the same subject from two different points of view. BPM and WfM foster a mainly process-oriented perspective on organisations. This process-oriented view comprises activities and their relationships within and to an organisation's context. Relationships among business processes might be specified using control flow (consecutive, parallel or alternative execution), hierarchical decomposition and/or generic relationships. Relationships to the organisational context comprise the assignment of organisational units (company, department, role) and resources (tools, machinery).

Nevertheless, a more differentiating reflection on business processes and workflows seems to be appropriate[1]. Referring to several sources, they both represent different levels of abstraction on process-oriented organisations. According to Frank and van Laak a workflow mainly concentrates on the processing of digital office documents[2]. Human activities (in terms of manual processes) as well as decision making processes are left out or at least reduced to interactions with software applications. Similar discussions on conceptual distinctions can be found in the literature. Some examples are given as follows: Böhm summarises the conceptual differences between business processes and workflows as the emphasis of IT on workflows. Like other authors, he places business processes more on a conceptual level of the enterprise[3]. Junginger also mentions the fact that every kind of resource might be assigned to a business process[4] while workflows are mainly supported by IT-related resources. Stark characterises workflow by the management and support of business processes combined with IT[5].

This research paper describes a first approach towards the mapping of concepts of a given business-process-modelling-language to workflow schemata. This work outlines conceptual equivalences and differences. For the support of the mapping of business processes to workflow schemata we will especially focus on information which has to be added to business processes in order to map them to workflows. The structure of this report is given as follows: The next two sections give an overview on business process modelling (section 2) and workflow management (section 3), respectively. Extensions to the business process modelling language (section 4) as well as a prototypical implementation of a tool (section 5) will be presented afterwards. This report ends with a summary, concluding remark and an outlook to future work in section 6.

2 Business Process Modelling

This preliminary section provides an overview over general aspects and the area of application of business process modelling. Additionally, core concepts of a business process language will be presented.

[1] The notion of business process and workflow are further presented in the following sections 2 and 3.
[2] Cf. [FrLa03]
[3] Cf. [Böhm00, p. 3]
[4] Cf. [Jung01, p. 18]
[5] "Workflow promises a new solution to an age-old problem: managing and supporting business processes. What is new about workflow is the way it harnesses the power of information technology to structured work." [Sta97, p. 5]

2.1 General Overview

The analysis, representation and management of knowledge about an organisation and its processes has always been very important[6]. A lot of work has been done on the development and evaluation of ontologies for process modelling[7], the specification of process modelling languages[8] as well as on business process modelling methods and concepts[9]. Business process models can be used for different kinds of purposes:

- Documentation of processes of an organisation to foster communication[10]
- Analysis of business processes[11]
- Simulation of processes[12]
- Support for business process re-engineering[13]
- Software development of process-oriented applications[14]

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. Business process models represent a common medium for the communication of domain experts and novices. They offer domain level concepts 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 detailed description (with respect to the analysis' purpose) of process models and related 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. Appropriate language features provided by a process modelling language support the determination of media clashes, unnecessary processes 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.

2.2 Business Process Modelling with MEMO-OrgML

Multi-Perspective Enterprise Modelling (MEMO) is a method for the modelling of organisations according to different views as well as different levels of abstraction[15]. 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 (Organisation Modelling Language), which supports modelling of organisational structures and processes. Resource modelling has not been subject of the first conceptualisation of the MEMO-OrgML but will be added shortly[16].

An introductory example for a process that has been modelled using MEMO-OrgML is given in Figure 1. An order is received by the distribution department and the data will be checked directly afterwards (process No. 1 called 'Check Data'). The order will be further processed if the given data is

[6] Cf. [KoPl00]
[7] Cf. [WaWe93], [Web97] and [GrRo99].
[8] Cf. [EJLT99] and [SuOs97]
[9] Cf. [Herb97] and [Öst95].
[10] Cf. [Obe96] and [Fra99]
[11] Cf. [BeJo01], [EJLT99] and [Sche99].
[12] Cf. [Baum96]
[13] Cf. [CKO92] and [Obe96]
[14] Cf. [CKO92], [Öst95] and [Fra99]
[15] Cf. for example [Fra99].
[16] The preliminary conceptualisation can be found in [Jung03].

valid (event No. 4 called ‘Valid Data’) or aborted if the data seems to be invalid (event No. 3). Aborting an order results in sending a rejection message to the customer in process No. 9 (‘Compose Rejection Message’). Further processing of the order comprises the entering of the data into the order-management-system (process No. 2: ‘Enter Order’) the parallel execution of the processes 6, 7 and 8. Process No. 6 is a composed process which consists of one or more sub processes. The process called ‘Compose Acceptance Message’ (No. 7) is a semi-automated process executed by the distribution department. Process No. 8 is a fully automated process sending a default email-message to the customer.

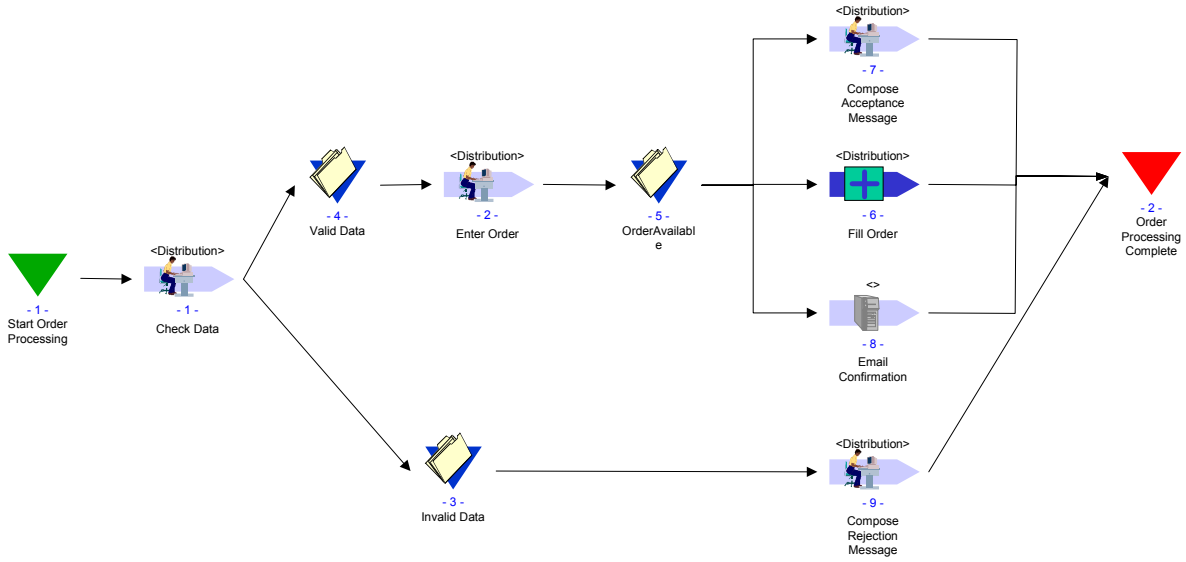

Figure 1: Process Modelled Using MEMO-OrgML

The example given above contains only an extract of the language features offered by the MEMO-OrgML. A detailed presentation of all language features will be given in the following sections. Section 2.2.1 starts with different kinds of processes and will be followed by the presentation of events in section 2.2.2 and the specification of control-flow between process elements in section 2.2.3. We will close with the discussion of additional concepts provided by the MEMO-OrgML in section 2.2.4.

2.2.1 Processes

The general specification of a process in MEMO-OrgML is shown in Figure 2. Every process can be assigned to an organisational unit, which is annotated on top of the graphical notation of a process. Examples for organisational units are a whole organisation, department, business unit or a role. Additionally, a process can be identified by a unique number (‘number’ in Figure 2) and described by a meaningful qualifier.

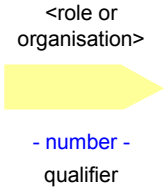

Figure 2: Process

There are several different types of processes in MEMO-OrgML which are classified using different kinds of aspects.

2.2.1.1 Elementary Process Types

Elementary processes in MEMO-OrgML are classified by the types of resources required for their execution. Processes can be executed manually, automatically or semi-automated (refer to the graphical notation in Figure 3). Manual processes are exclusively performed by human resources without any IT-support. In contrast to this, automated resources are solely executed by computing machinery without any support of human resources. Semi-automated processes refer to a support by human and technological resources. Whether semi or fully automated, the focus is on IT resources.

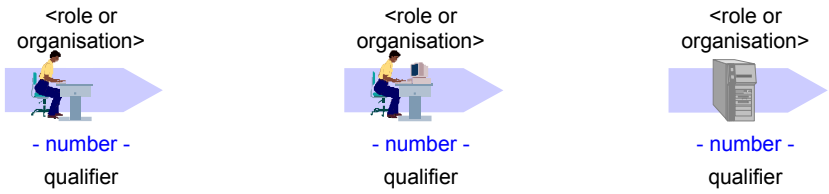

Figure 3: Types of Processes

The processes presented so far are usually subject to a specific start and end. A manual process for the installation of a controller for a central heating in a building can only begin after an order is entered and will end with the completion of the order. Nevertheless, there are some tasks without a specific start and end. Those processes usually run continuously. An example for such a process is the quality assurance. Tasks related to quality assurance might not directly relate to specific sub processes in software development. Furthermore, quality assurance is orthogonal to a software development method because it has to be guaranteed at every stage of a software development process. Hence, quality assurance has to be done continuously, keeping every task regarding software-development in mind. The graphical notation for continuously running processes is displayed in Figure 4.

Figure 4: Continuously Running Process

2.2.1.2 Aggregated Process Type

An aggregated process is composed of other elementary or aggregated processes. Equally like general processes, aggregated processes can be specified by assigning an organisational unit and annotating a unique number and a descriptive name (qualifier). The graphical notation for an aggregated process is given in Figure 5. It is important to note that an aggregated process is mainly specified by its sub processes. Also its sub processes are assigned to an organisational unit and specified according to necessary resources. Furthermore, an essential part for the specification of an aggregated process is the control flow between its sub processes. Hence, every aggregated process has to be specified by a process model containing its sub processes and their control flow.

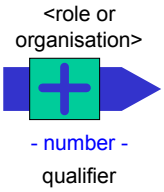

Figure 5: Aggregated Process

2.2.1.3 External Processes

External processes are generally executed by external organisations. Examples for external processes are those which are executed by an autonomous partner or a subcontractor. In MEMO-OrgML, we differentiate between different kinds of partners regarding external processes. The main focus is on the independency of the given organisation. According to the graphical notation given in Figure 6, there are types of partners (from left to right): contractor, autonomous institution and partner.

Figure 6: External Processes

A contractor is an external organisation, which is bound to the terms and conditions of the given organisation. It is an independent organisation but depends on the requirements given by the principal. An example for a contractor is the supply industry in the automotive sector. Every supplier depends on a car manufacturer specifying the technical data and contracts. In contrast to this, an autonomous institution is very independent from the given organisation. An autonomous organisation is free in the specification of contracts and cannot be forced by other organisations. External partners are a special kind of contractor, which are more independent than a contractor but less free than an autonomous organisation.

2.2.2 Events

Events represent special states during the execution of business processes. An event is a momentous symptom and not a period of time. The most important types of events in MEMO-OrgML are given in Figure 7 (from left to right): Start event, stop event, an event indicating the change of a state and an event for an incoming message. For every business process there is a dedicated starting event as well as a final state. Those events form the boundary between the modelled process and the organisation's context. In contrast to a general event indicating the change of an internal state, the incoming information event corresponds to an increase of information within the process system.

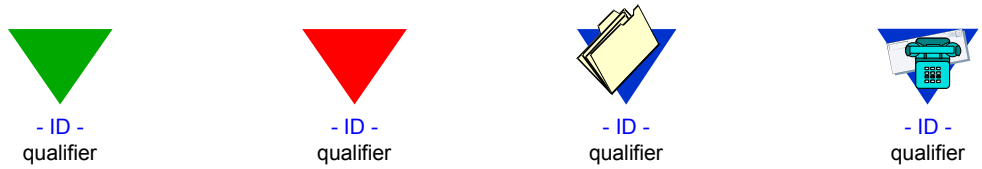

Figure 7: General Events

The event types presented so far cover general aspects of process models like start, stop, status changes and incoming news. Nevertheless, they ignore timely aspects like points of time and periods (their graphical notation is given in Figure 8). A point of time corresponds to a well-defined timestamp and a period to an interval given by a well-defined end. Hence, a

moment can be specified by its absolute specification. The end of a period might only be described by a timestamp relative to a given or imaginary starting point.

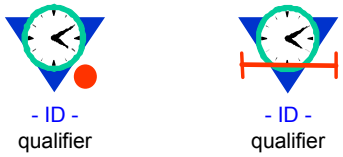

Figure 8: Time-related Events

2.2.3 Control Flow

Control flow specifies flow-related relationships between processes. Those relationships might be determined by logical or temporal constraints on the execution of business processes. We mainly distinguish between sequence, concurrency and alternative.

2.2.3.1 Sequence

A sequence in business process modelling usually corresponds to the consecutive execution of processes. Hence, the termination of one process results in the instantiation of exactly one following process. Figure 9 shows an example for a sequential execution regarding to events. The interpretation is as follows: If event No. 1 (data available) occurs, process No. 1 (Check Data) can be started. The termination of this process fires event No. 2.

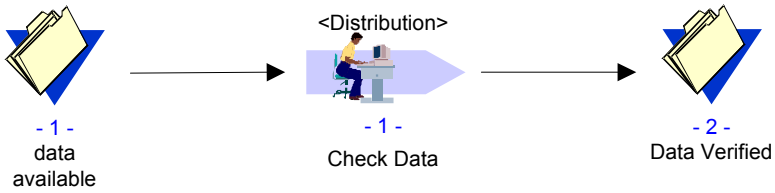

Figure 9: Example for a Sequence

2.2.3.2 Alternative

An alternative is as general rule interpreted as a fork of control flow. This means that after the execution of one process called A either a process B or a process C is initiated. Alternative execution of business processes. In other terms there can be only one successor of process A within a concrete instantiation of the whole process, namely B or C in the example. In sense of mathematical logic this represents an exclusive-or (XOR) relationship between following processes. This is not restricted to only two processes. Several succeeding processes can be involved in such a relation. In addition to the fork of process flow there is also an equivalent join. The graphical notation for an alternative is given in Figure 10. After the process No. 1 is completed either the events 3 or 4 will be fired. If event No. 3 (data is invalid) is fired, the execution continuous with process No. 2 (enter order)

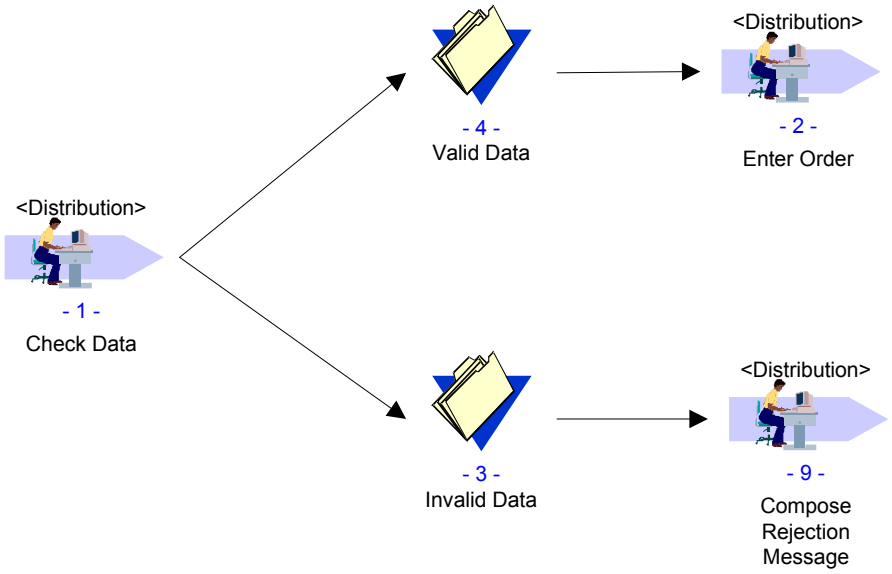

Figure 10: Alternative Execution

2.2.3.3 Concurrency

Many processes can be executed in parallel or concurrently. Generally speaking, the parallel execution is identical with the execution at the same time. In contrast to this, concurrency only means, that processes might be executed independently from others. But there is no such differentiation in MEMO-OrgML. There is only one language feature for concurrency, neglecting the fact of simultaneous execution. An example for the concurrent execution of processes is given in Figure 11. Event No. 5 (OrderAvailable) fires the concurrent execution of the two processes Fill Order and Email Confirmation (AND-split). The synchronisation (AND-join) of these two parallel paths results in event No. 2 called Order Processing Complete. Alternatively an OR-join[17] is possible if only one parallel branch has to terminate in order to fire event No. 2.

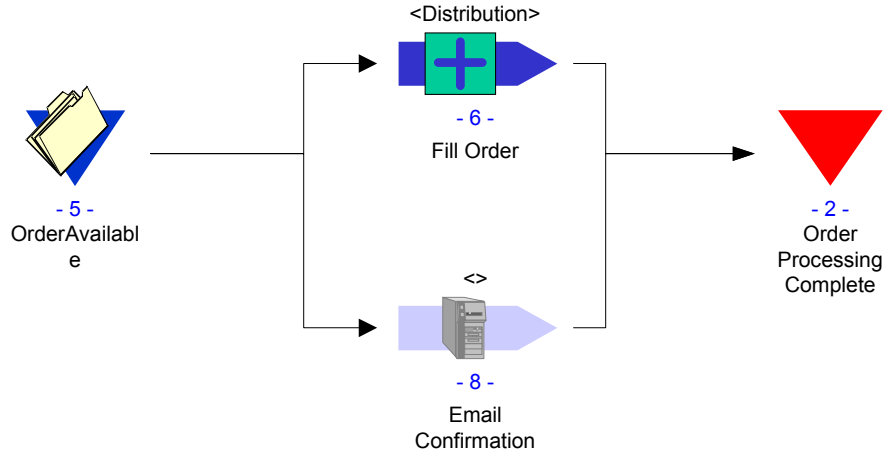

Figure 11: Parallel execution

2.2.4 Additional Concepts

In addition to the process-oriented concepts, exceptions, notes and constraints are provided in the MEMO-OrgML (their corresponding graphical notation is shown in Figure 12). An exception corresponds to an event which indicates an unusual behaviour during the execution

[17] The graphical notation for an Or-join is not given in this document.

of a process. Examples for such kinds of exception are the failure of an IT resource, the breakdown of a truck or the illness-related absence of a human resource. Those kinds of exception are usually hard to integrate as regular events with business process models using common control flow. It is not always clear what has to be done in case of an exception. Hence, the exception assigned to a business process indicates the appearance of a faulty event. A procedure for handling every kind of exception has to be described or at least a global exception handling routine.

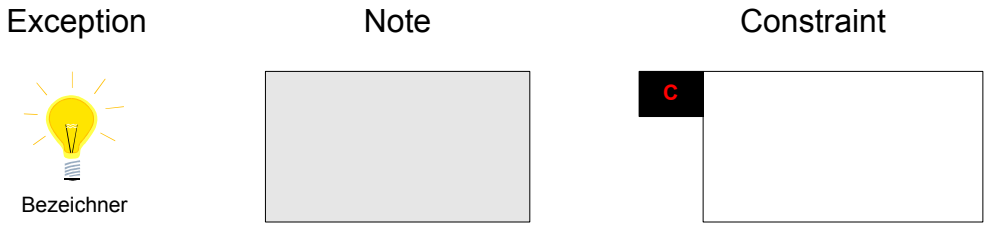

Figure 12: Additional Concepts

A note is a textual description for special aspects of a business process model and is intended for human readers. Automatic processing of notes by computerised IS-systems is not planned. A given note might be related to a process model, a single model element or any extract from the model. In contrast to this, a constraint serves to the specification of formal conditions. A constraint usually refers to information objects used in the business process. Consequently, a constraint can only be formulated formally if business documents are described in a formal manner.

2.3 Other Business Modelling Concepts

In addition to the process-oriented concepts given in section 2.2 there are two other kinds of language features for modelling process-oriented organisations. Organisational units correspond to departments, divisions or roles which are assigned to a business process as a responsible actor. Resources are actors or tools which are required for the execution of a business process.

2.3.1 Organisational Units

The static structure of organisations can be described by an organisational chart. Such a chart shows an organisation by its sub-units and their respective relationships. The meta-model for the modelling of organisational charts is given in Figure 13 a). An abstract organisation unit might either be a position or an organisational unit. Every organisational unit is a composition of other abstract organisational units. A position is an elementary description of the responsibilities of an employee. An example for an organisational chart is given in Figure 13 b): An imaginary company consists of three departments for procurement, production and distribution.

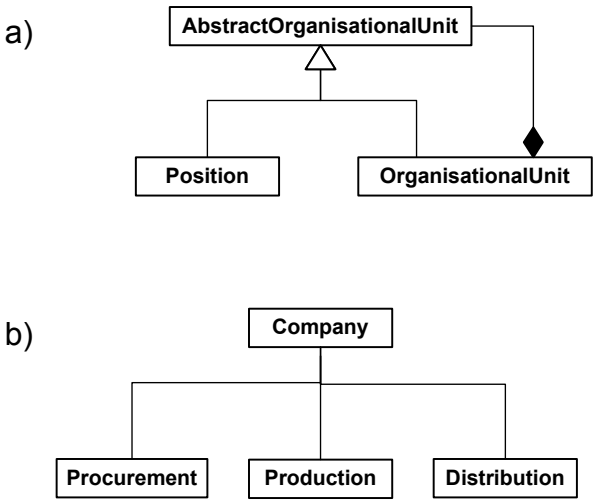

Figure 13: Specification and example of organisational units

Organisational units and roles are elements assigned to business processes and refer to human actors which are responsible for the execution of a business process. Organisational units and positions as described here can be assigned to business processes as described in section 2.2.1. Roles are not necessarily defined in an organisational chart but can be assigned to business processes.

2.3.2 General Resources

Resources are essential for modelling processes[18]. Processes and their relationships mainly describe dynamic aspects and the order of events. Resources assigned to processes additionally specify subjects and objects of business processes. Resources are usually not available in an unlimited amount[19]. 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. As the resource-modelling-language for MEMO-OrgML is currently under development, no graphical notation will be given here but a short introduction into the underlying meta-model. An excerpt from this meta-model is shown in Figure 14. The class AbstractResource is the root of the generalisation hierarchy on resources. Every resource has a name (name : String), a textual description (description : String) and a list of resource attributes (attributes[0..*]:ResourceAttribute). Every resource attribute is a name-type pair for the specification of user-defined attributes on resources.

[18] cf. [PSO99]
[19] cf. [Nübe01] and [PSO99]

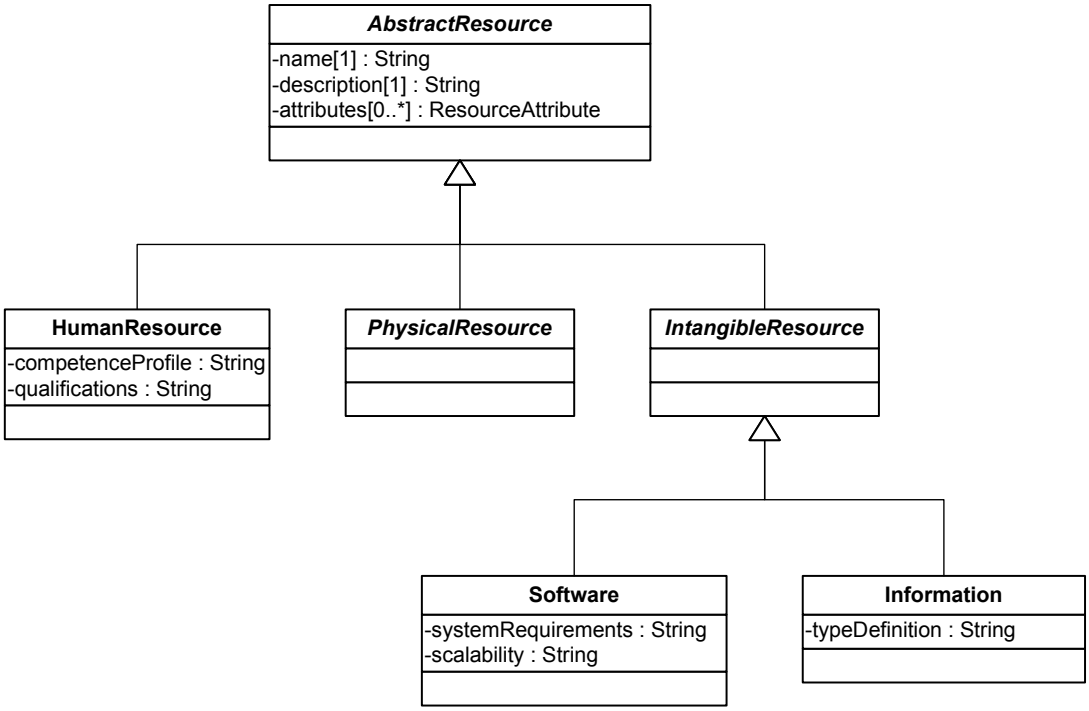

Figure 14: Excerpt from the resources' meta-model

Within the context of workflow-modelling, we generally distinguish between human, physical and intangible resources. A human resource is an abstraction on different perspectives on staff. Examples for such perspectives are concrete employees, roles filled by employees or business-oriented functions. Physical resources comprise all tangible objects used within a business process. Examples for physical resources are production plants, raw material or computer hardware. In contrast to this, intangible resources do not have a physical manifestation. Examples for intangible resources are data, information, software or even knowledge.

Human resources are an abstraction on persons, employees, roles or other staff-related perspectives. They might be associated with concrete persons or employees of an organisation as well as abstract organisational units in an organisational chart. Hence, a human resource can be characterised by different aspects. A human resource …

- … can play an active role
- … may be responsible for the execution of e-business processes
- … needs some qualification and competences for its job

The type HumanResource is a subtype of AbstractResource and has the two attributes qualification and competenceProfile, both of type String (cf. Figure 14). The qualification is an objectively describable criterion for the capabilities of a human resource. Usually, the qualification certificate is issued by an established educational body. The competence of a human resource reflects personal skill of human beings. Hence, a competence profile corresponds to personal strengths.

Physical resources comprise all tangible objects used within a business process and are neither human nor intangible. According to Heinen[20] - in the context of industrial production - it can be differentiated between non-consumable resources (German: *Potentialfaktoren*) and consumable resources (*Repetierfaktoren*). Non-consumable resources are not used up during a manufacturing process and are still available afterwards whereas consumable resources are either becoming a part of the resulting product or are being used up and therefore are not

[20] see [Hei88], p. 242

available anymore[21]. In the paper at hand we abstract from physical resources, because these are not relevant in our context of workflow applications and the perspectives we present on it.
Intangible resources are resources without a physical manifestation. Software in terms of a set of programs that run on a computer hardware is a key resource in the process of supporting workflows. The meta-type Software has two attributes: systemRequirements and scalability both of type String. The system requirements are modelled as text and describe the environment for the execution of a software system (i.e. processor architecture, minimum main memory or operating system). Scalability corresponds to the ability of supporting growing numbers of clients. The meta-type Information was created to represent information or knowledge that is relevant within workflows. It has attributes name (a symbolic reference to an information instance) and typeDefinition of type String which describes how the information is structured. Examples for information are certain customer data or enterprise knowledge of some kind.

3 Workflow Modelling

This section provides an overview on workflow management concepts. It covers the basic terminology, major standards for workflow schema interchange and presents a standard for schema definition of the WfMC – XPDL.

3.1 General Overview

Workflow management is an important technology in the area of IS Research and focuses on the support and management of electronically supported processes. This section provides a short introduction into workflow management by giving an overview on common terminology and the description of the standards of the Workflow Management Coalition.

3.1.1 Workflow Management Coalition

The *Workflow Management Coalition* (WfMC) was founded in 1993 and is an alliance of companies and organisations dealing with workflow management. The mission of the WfMC is the foundation of a common terminology regarding workflow management and the establishment of standardised interfaces. These interfaces comprise the definition, execution and management of workflows as well as references to external documents and applications[22]. The conceptualisation of the interfaces is given by the WfMC's reference model depicted in Figure 15. Core of the reference model[23] are the workflow enactment services using one or more workflow engines for the execution of workflows. A workflow engine is a software managing workflows regarding to given workflow definitions.

[21] see [SS01], pp. 89-90
[22] Cf. [Jung01, pp. 126] and the references given there
[23] The WfMC reference model is specified in [Holl95].

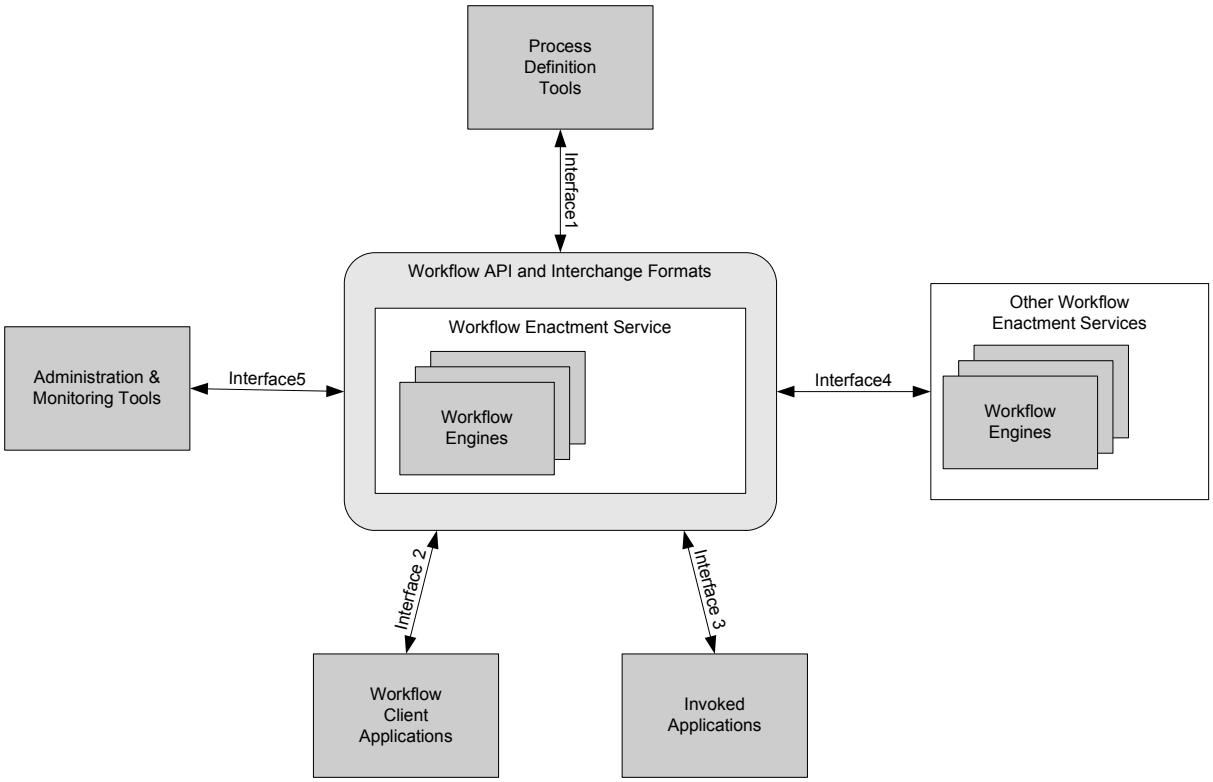

Figure 15: Workflow Reference Model of the WfMC[24]

The five different interface definitions correspond to the integration of external aspects:

- *Interface 1* specifies the exchange of workflow models between external modelling tools and a workflow management system. External tools might be graphical editors for workflow definition or just a textual editor. Nevertheless, some general purpose process modelling tools supporting the WfMC standard can be used for the specification of workflows[25].
- *Interface 2* describes the communication between a WfMS and workflow client applications. Workflow client applications are applications directly correlated with the workflow engine. They usually implement basic functionality of workflow applications like notification and data transfer[26].
- *Interface 3* addresses the need of integration of external applications. Usually, the needed functionality might not be completed by the WfMS. Hence, there has to be an interface to other applications already running in the enterprise[27]. Examples for such kind of applications are business related software and special software tools.
- *Interface 4*: Goal of interface 4 is the integration of other workflow management systems. The specification comprises the invocation of remote activities, data transfer as well as synchronisation aspects between different workflow enactment services[28].
- *Interface 5* describes the communication between the workflow enactment services and external monitoring and administration tools[29].

Interface 1 is the most relevant specification for the purpose of mapping business process models to workflows. It concentrates on the specification of different types of workflows

[24] Source: [Holl95, p. 20]
[25] Cf. [Gad01, p. 48]
[26] Cf. [Holl95, pp. 31]
[27] Cf. [Holl95, pp. 35]
[28] Cf. [Jung01, p. 126] and [Holl95, pp. 41]
[29] Cf. [Jung01, p. 126].

(processes) as well as associated organisational units and applications. The system-specific integration of applications is done using interface 2/3[30].

3.1.2 Workflow Specification Languages

WPDL[31] is the first attempt of the WfMC to specify of a standard for the interchange of workflow definitions. Being a standard for exchanging models, it does not comprise a graphical notation. Meanwhile, WPDL has been replaced by XPDL[32], an XML-based document definition for workflows. The basic conceptualisation of the XPDL is represented by the meta-model shown in Figure 16. This meta-model generally comprises static entities (e.g. data or applications) as well as dynamic concepts (processes). Static entities are represented by the meta-types

- Workflow Relevant Data,
- Workflow Participant Specification and
- Workflow Application Declaration.

Workflow-relevant data is initialised, created, read from external applications and used during the execution of workflows[33]. It might be produced by an activity within a workflow or extracted from an external data source (like an enterprise information system). The creation of a new data entity or the digitalisation of a document might be sources for the creation of data in the context of a workflow. Examples for external data sources are corporate databases containing relevant data for an enterprise. These data sources are represented by the meta-type System and Environmental Data in Figure 16. The workflow participant specification describes the resources which perform the given workflow processes[34]. This specification does not necessarily correspond to a human or a single person. It actually represents an abstract resource or a role which can be filled by one or more humans as well as an automated machine. Nevertheless, the specification of a workflow participant corresponds to a resource available in an organisation or an entity in an organisational chart (Resource Repository or Organisational Model). The workflow application declaration provides the description of software applications needed for the execution of a workflow process[35]. Those applications are usually invoked by the workflow engine and workflow-relevant data has to be passed as a parameter. Examples for workflow applications are internal applications as well as external applications like corporate information systems or common office applications. Internal applications are usually provided as part of a workflow management system or can be developed using a proprietary[36] development environment or language.

[30] Interfaces 2 and 3 are combined to one interface definition by now (cf. [WfMC98] and [Jung01, p. 127]).
[31] Workflow Process Definition Language
[32] XML Process Definition Language (cf. [Nori02])
[33] Cf. [Nori02, p. 10].
[34] Cf. [Nori02, p. 9].
[35] Cf. [Nori02, pp. 9].
[36] By the term ‘proprietary’ we mean an environment or language which is part of the WfMS.

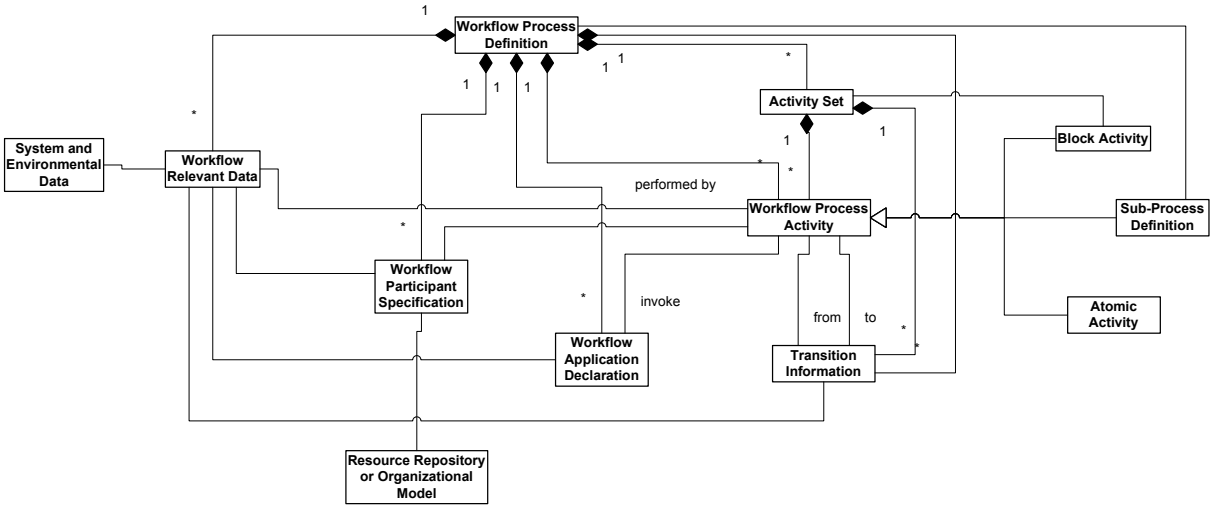

Figure 16: Meta-model of the WfMC[37]

A workflow process definition is an aggregation of static entities (data, applications, participants) as well as the description of the system's dynamic behaviour. Dynamic aspects of the meta-model are represented by the entity-types Transition Information as well as Workflow Process Activity and its concrete subtypes

- Block Activity,
- Atomic Activity and
- Sub-Process Definition.

An activity is a given unit of work which will be executed by a participant using computer applications[38] and relevant data. Additionally, every activity is characterised by a start- and end-time as well as the fact whether it can be executed automatically by the WfMS or by a workflow participant. The transition information specifies the control flow between activities[39]. It consists of a starting activity, an end-activity and a condition under which the transition is made. An atomic activity is an indivisible unit of work which has to be done at one go. A sub-process definition allows the embedding of another workflow process definition. A block activity consists of a set of other activities (type Activity Set). The semantics of an activity set is similar to the one of a macro. If an activity set is called during the execution of a workflow process, the activities contained in the set are copied into the calling process definition[40].

The concepts given here will be further discussed later in this research report. Different types of processes and transitions will be presented in the following section 3.2. Additional concepts like applications or participants are subject to section 3.3.

3.2 Basic workflow concepts

Main concepts for the description of the dynamic aspects of a workflow system are activities and transitions. Activities correspond to defined units of work which can be atomic or consist of a set of activities. The control flow between activities is specified by transitions. Hence, a transition relation between two activities defines the ordering of these activities. An activity can be started if its preceding activities (connected by transitions) have been terminated. Transitions, activities and static entities (i.e. IT-related resources) are grouped by a so called workflow process definitions.

[37] Source: [Nori02, p. 12]
[38] Cf. [Nori02, p. 8]
[39] Cf. [Nori02, p. 9]
[40] Cf. [Mato03, p. 13]

3.2.1 Workflow Process Definition

A **workflow process definition** groups all elements necessary for the execution of a workflow. As shown in the meta-model in Figure 16 these elements comprise dynamic (activities and transitions) and static aspects (data, applications and participants). Additional attributes are a unique identifier, a name and two headers. The *process header* comprises the creation date, a textual description and different time-related properties (e.g. estimated duration of a process' execution) of a workflow process. The *redefinable header* consists of information about the author of the process definition, a country key, its publication status, responsible participants and a version number.

An **activity set** is a set of activities and transitions. All transitions contained in this set can only start from activities within this set and end in activities within this set. In other words, there are no transitions leaving an activity set or coming from outside. Properties of an activity set are a list of activities, a list of transitions and a unique identifier.

3.2.2 Workflow Process Activities

As shown in Figure 16, there are different types of activities within a workflow process definition. An atomic activity is an indivisible unit of work executed under the control of a WfMS. Such an activity can be executed automatically or by a human participant and usually works on workflow-relevant data. In contrast to this, block and route activities do not refer to workflow-relevant data. A block activity executes an activity set and has no own behaviour. Invoking an activity set means the start of the first activity in the set. The execution terminates with the last activity in the activity set (cf. Figure 17). A route activity is an activity with no behaviour. It only serves as a dummy activity for cascading transition conditions[41].

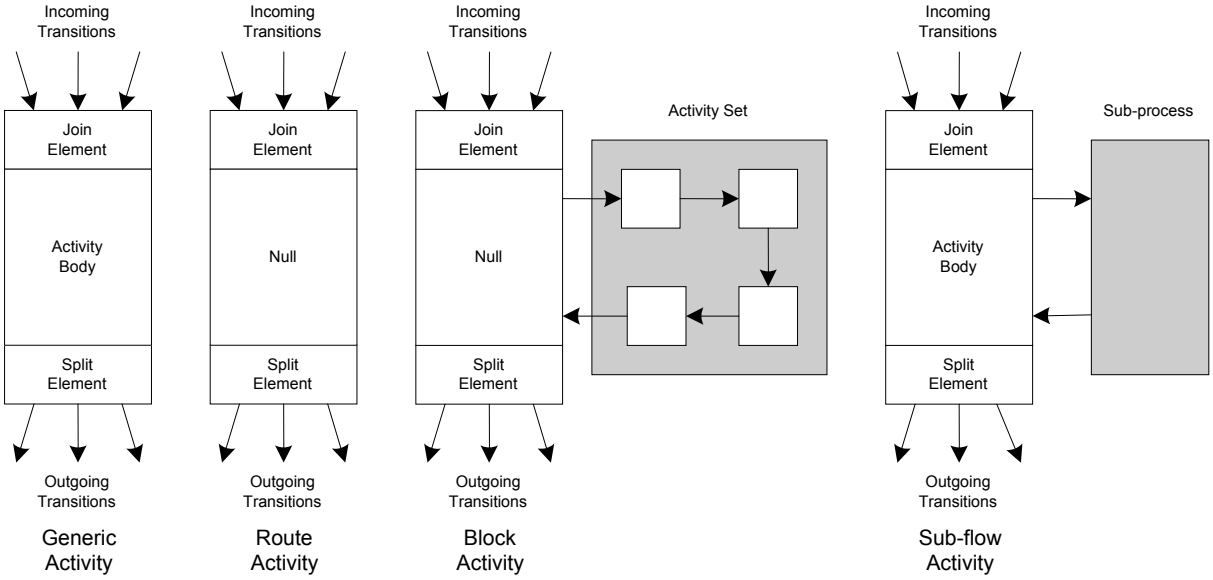

Figure 17: Different kinds of Activities in XPDL[42]

According to XPDL there is only one general XML-element for activities called 'Activity'. Specific elements for route, block or sub-flow activities are missing. The differentiation between different types of activities is done by the annotation of alternative attributes. Those attributes are named Route, Implementation[43] and BlockActivity. Activities can additionally be specified according to their level of automation (automatic or manual) as well as their

[41] Please refer to section 3.2.3.

[42] Source: [Nori02, p.30]

[43] The WfMC uses three different names for the same concept: An *atomic activity* (cf. [Nori02] , p. 12) is also called generic activity (cf. [Nori02], p. 30) and implementation (cf. [Nori02], p. 31).

implementation alternatives (no implementation, tool or subflow). An automatic activity can be fully controlled by the workflow engine using internal and external applications. Manual activities require the involvement of a human being. Activities corresponding to the no-implementation alternative cannot be supported by an WfMS. These are usually manual tasks which can be executed without the support of a WfMS. A tool supported implementation implies the support of a software application. Such applications have to be assigned to this kind of activity[44]. If the implementation type is set to *subflow*, the execution has to be delegated to another workflow process definition. Parameters can be passed to such a sub-flow activity and the synchronisation can be specified with respect to a synchronous or asynchronous execution. Synchronous execution requires the calling process to wait for the termination of the called process. After its termination the called process might pass output values to the calling process. During an asynchronous execution the calling process has not to wait for the termination of the called process and output values are not possible.

Atomic, tool-supported activities might be executed by a human actor. Such a human resource corresponds to the XPDL-type participants. Human resources and participants can be associated with staff members[45]. This association is not part of the XPDL specification and has to be implemented by a concrete WfMS-implementation. XPDL only describes on an abstract level the participants of a workflow correlating them to typical roles. In contrast to this a WfMS manages users of a system which can in turn fill a specific role. The mapping of the roles provided in an XPDL-description to roles given in a WfS[46] has to be done by the WfS itself. If an XPDL-participant cannot be mapped to a WfS-role, a default role has to be applied. A participant without any correspondence in the WfMS might – for example – be assigned to a default role like an administrator.

Additional information for activities are deadlines and simulation information. A deadline is the expiration of a given period of time. A deadline might for example be a milestone (given a project management context) or specific appointment. The occurrence of a deadline can be handled synchronously (the current activity is interrupted by the deadline) or asynchronously (the handling of the deadline has to be done parallel to the currently running activity). Simulation information extends the model by giving specific data for the simulation of models. Examples for specific data are average costs, expected duration and average waiting time.

As shown in Figure 17, every activity is a join-point for several incoming transitions (join element) and specifies the type of splitting for outgoing transitions (split element). Both – join and split – can refer to a parallel or an alternative execution of workflows. An alternative split (XOR) represents a fork specifying that exactly one of the given alternatives can be executed. An alternative join corresponds to the synchronisation of an alternative split[47]. The parallel execution of activities is started by an AND-split and ended by an AND-join. Rules for the construction of workflow descriptions regarding parallel and alternative connectors (splits and joins) are classified by so called conformance classes. A conformance class specifies criteria for the construction of diagrams of activities. A NON-BLOCKED conformance class implies no formal properties of a diagram regarding the relationships between splits and joins. If the conformance class is set to LOOP-BLOCKED, the graph build by the activities and transitions is a directed acyclic graph[48]. A FULL-BLOCKED graph implies that every AND-split has exactly one AND-join, every XOR-split exactly one XOR-join and vice versa. Additionally every path starting from the split will reach the corresponding join.

---

[44] The assignment of applications to workflows will be presented and further discussed in section 3.3.2.

[45] This will be subject to section 3.3.1.

[46] Workflow System

[47] We assume that every alternative split has an equivalent alternative join. Hence, every path beginning at a given alternative split will end at one – and exactly one – alternative join.

[48] DAG = Directed Acyclic Graph

3.2.3 Transitions

A transition is the partly specification of the control flow between activities. As shown in section 3.2.2, the information whether incoming transitions of an activity are disjoint (alternative) or conjoint (parallel) is assigned to the activity. Additional information about a transition is assigned to the so called *transition information*[49]. Basic elements of such a transition are its name (i.e. a character string), a textual description and a condition. While a description usually consists of a natural language description a condition should be a (semi-)formal specification of the circumstances enabling or disabling a transition. It therefore has to be represented by a Boolean expression. Additionally, a starting and an ending node are assigned to transition information.

Consequently, every transition is characterised by exactly one source activity (*from*), exactly one destination activity (*to*) and a Boolean expression representing a firing condition. There are four different kinds of conditions in XPDL:

- CONDITION: A transition can fire if its condition is evaluated to *true*.
- OTHERWISE: Indicates a default transition which will fire if no other transition's condition evaluates to *true*.
- EXCEPTION: An exception is a special transition indicating an abnormal behaviour. An exception-condition can trigger the rising of a special condition.
- DEFAULTEXCEPTION: A default exception is triggered if all other exception conditions are evaluated to *false*.

Dis- and conjointness of transitions are specified by the splits and joins assigned to activities. Conditions regarding the validity of the execution of a transition are assigned to the transition information. Hence, the description of the control flow in a workflow model is split up into the nodes (activities) and arcs (transition information) of a workflow model.

3.3 Extended Concepts

Workflows are managed by a workflow management system by assigning tasks (as parts of workflow instances) to given resources. Such a resource might be either a human participant or a workflow application. A human resource usually corresponds to a role filled by a specific person in an organisation. A workflow application might be categorised into internal and external applications. An internal application is usually implemented by the WfMC itself[50] and is closely coupled to the workflow system. An external application can be characterised as an application independent from the WfMS.

Regarding the specification of resources for the execution of workflows there is one major problem. On the one hand, XPDL aims to be a language for a system-independent workflow definition interchange. Hence, a workflow model described using the XPDL should be independent from any specific workflow engine. On the other hand, the description given by an XPDL-document should be precise enough for the execution of workflows. This aspect might require the annotation of specific users or applications which are subject to a proprietary definition by a WfMS. Consequently, the XPDL-definition only provides an abstract mechanism for the specification of human resources and software-applications.

3.3.1 Workflow Participants

Regarding the XPDL-specification, workflow participants are "an abstraction level between the real performer and the activity, which has to be performed."[51] The engine has to map

[49] "The Transition Information describes possible transitions between activities and the conditions that enable or disable them (the transitions) during workflow execution. Further control and structure restrictions may be expressed in the Activity definition." [Nori02, p. 40]

[50] To be more precise: An internal workflow application is usually implemented using a programming language and environment given by the WfMS.

[51] Cf. [Nori02, p. 43].

every abstract participant to a user given in the workflow management system's environment. Every abstract participant is characterised by its unique name and type[52]. Possible types of workflow participants are:

- RESOURCE: A specific resource given in a workflow management system's environment.
- RESOURCE_SET: A set of resources.
- ROLE: A role description that directly corresponds to a role given in an organisational chart. Such a role might be a function or some kind of qualification filled by a human.
- ORGANIZATIONAL_UNIT: An arbitrary element of an organisational chart.
- HUMAN: A human being interacting with the WfMS by worklists and/or applications (i.e. a concrete human being, like 'John Miller')
- SYSTEM: A software application representing the participant of a fully automated workflow.

Those participants are assigned to activities of a workflow model using the Performer-attribute of an activity[53]. Hence, an activity keeps a reference to an abstract participant using the performer-attribute (which is rather a character string than a reference to a workflow participant). Unique identifiers of participants are used to specify an activity's performers. A workflow model describes participants on an abstract level, like organisational units or roles.

3.3.2 Workflow Application Declaration

Regarding to Junginger, workflow applications can be divided into internal and external applications[54]. An internal workflow application is implemented as part of the WfMS. They are usually implemented using a programming language given by the WfMS. In the context of XPDL, those applications are called *procedure*. An external application is an individual software package which can be used by a WfMS. Hence, an internal application is part of the WfS and an external applications is part of the corporate information system involved in a workflow.
Using XPDL, a workflow application is specified by a unique identifier, its type and a list of parameters. The name of an application is rather the unique id and does not necessarily correspond to its physical location or a concrete implementation. Like the description of workflow participants, the identification of a workflow application is only a symbolic link. The interpretation of such a symbolic link representing a workflow application depends on the WfMS at hand. There are workflow engines supporting internal applications, external applications or both[55]. Hence, the differentiation between internal and external workflow applications relies on the capabilities of the workflow engine. In general, the mapping of abstract applications (procedures or applications) to concrete applications has to be done by the WfMS

4 Mapping OrgML to XPDL

This section focuses on the information given by a business process model and its transformation to a workflow model.

4.1 Workflow Process Definitions

Every workflow process definition in XPDL generally consists of activities, transitions, applications, participants and workflow-relevant data. Hence, such a workflow process

[52] Additional attributes are a textual description and a reference to an external description of a participant.
[53] Cf. [Nori02, p. 31].
[54] Cf. [Jung01].
[55] Cf. [Jung01].

definition comprises its activities and corresponding resources[56]. Additionally, every workflow process definition consists of two different headers and a body. The two headers are the *definition header* and the *redefinable header*. The workflow *process definition header* is valid for all sub-activities and a *workflow process redefinable header* might be overridden in subflows.

Name	Description
Definition Header	• Meta-information on a process and • Instance-specific data • E.g.: Version, temporal unit, estimated duration, priority
Redefinable Header	• Meta-information on a workflow process • Properties can be overridden in subprocesses • E.g.: author, publication status
Activity Set	• Set of activities and transitions

Figure 18: Prefix of a Workflow Process Definition

4.1.1 Workflow Process Definition Header

Attributes of a definition header for workflow processes are listed in Figure 19. The creation date is assigned to a process definition during its definition and therefore represents the definition time of a workflow schema. This information can be extracted from the modelling tool supporting MEMO-OrgML and is not part of the MEMO languages' specification. The workflow process' description can be seen as the description of the top-level process of a decomposition hierarchy in MEMO-OrgML. The valid-from- and valid-to-attributes allow the specification of a period of time for the validity of a process definition. Hence, a process definition can only be used between valid from and valid to (empty string means unlimited validity). As here is no equivalent concept in MEMO-OrgML we assume an unlimited validity for all processes.

OrgML (Meta-Data)	XPDL:Definition Header
Creation Date (meta)	Created (creation date)
Process Description	Description
	Duration & Duration Unit
	Limit (vendor-specific)
	Priority
	Time Estimation
	Valid From/To
	Waiting Time
	Working Time

Figure 19: Attributes of a Workflow Process Definition Header

The other attributes only contain information on workflow instances. The duration- and limit-attribute (the limit has to be interpreted by a specific WfMS and has no meaning in the context of XPDL) contain an expected duration for the execution of the given workflow-process using a specific duration unit. The time estimation is an aggregation of waiting- and working-time as well as the duration. The waiting time corresponds to the time needed for the preparation of a process' execution and the working time correlates with the expected execution time. Those concepts are not part of the MEMO-OrgML and have to be complemented to a process model.

[56] According resources are presented in section 4.2.

4.1.2 Workflow Process Redefinable Header.

The attributes of a workflow process redefinable header are listed in Figure 20. The meta-information on the author of a model and its version can be derived from the data available in the modelling tool.

OrgML (Meta-Data)	XPDL:Redefinable Header
Modeller (meta)	Author
	Codepage
	Country key
	Publication status
Organisational Unit	Responsible(s)
Version (meta)	Version

Figure 20: Attributes of a Workflow Process Redefinable Header

The annotation of a codepage has a rather technical reason. The codepage specifies the character-set used for the presentation of texts. Country keys are specified by the ISO in the ISO 3166 standard. The publication status indicates whether a process definition under revision (UNDER_REVISION), released (RELEASED) or in use (UNDER_TEST). Responsible corresponds to an organisational unit which is responsible for the execution of a given workflow process. The responsible person can be derived from the organisational unit in the MEMO-diagram.

4.1.3 Generation of Headers

XPDL-workflow-headers contain information on a workflow process (e.g. author, version), process-information (e.g. description, responsible) and instance-related information (e.g. duration, time-estimation). This kind of information is not part of a language specification. Instead, it can be managed by a modelling tool and then be mapped directly to an XPDL-based description. Some process information (e.g. priority) is not yet available in MEMO-OrgML and has to be supplemented with a process' definition. Instance-specific information should not be included in a business process modelling language for conceptual modelling. It might only function as additional information (like a workflow-diagramm for business processes) for existing business processes on a different level of abstraction.

4.2 Resources, Information and Organisational Units

At first sight, resources seem to be easy to map to workflow participants, workflow applications and workflow-relevant data. Nevertheless, this task is hampered by some details regarding the abstractions of resources on the one hand and the concepts given in XPDL on the other hand. These details are discussed in the following subsections.

4.2.1 Human Resources

According to MEMO-OrgML, a human resource is an abstraction on persons, employees, roles or other staff-related perspectives. In XPDL, workflow participant "is an abstraction level between the real performer and the activity, which has to be performed. During run time these abstract definitions are evaluated and assigned to concrete human(s) and/or program(s)."[57] The mapping of an abstract actor (as given in an XPDL-description) to a concrete actor (e.g. the user of a WfMS) has to be done by the workflow-management-system[58].

OrgML:HumanResource	XPDL:Participant

[57] Cf. [Nori02, p. 43]
[58] It will be prescinded from a concrete WfMS within this section.

	Id
name	Name
description	Description
	Participant Type
	Extended Attributes
	External Reference
attributes	
qualification	
competenceProfile	

Figure 21: Correlation between Human Resources and Participants

As shown in Figure 21, the resource's properties name and description can be directly mapped to an XPDL-file. The properties attributes, qualification and competenceProfile will be neglected because they have no direct correspondence in XPDL. A unique identifier required by XPDL can be generated in the context of the mapping of OrgML to XPDL. Such an identifier corresponds to an object identifier in MEMO and the generated XPSL-Id has to be unique within the XPDL-definition.
The specifications of "participant-type", "extended attributes" and "external reference" are an extension to a business process model (modelled using MEMO-OrgML). Alternatives for a participant type are resources, roles, organisational units, humans and a software system[59]. Extended attributes are name-value pairs[60] and allow the annotation of system-specific information for different WfMS-products. The name is used to identify the extended attribute and the value is an information for a particular WfMS. An external reference is a reference to an external document providing the specification of a workflow-related entity. Such a document can for example be a globally available XML-DTD (specifying the structure of a workflow entity) or a web-services interface-definition (using WSDL). All these extensions have to be provided by a workflow-specific extension to business-process-models.

4.2.2 Software

The XPDL's notion of a workflow application declaration mostly corresponds to software used within a business process. A workflow application represents a software-tool required for the execution of a workflow. Every application might be invoked by the WfMS and the XPDL abstracts from concrete implementations. Consequently, applications are declared in an abstract manner, only naming them in the XPDL-definition. Every application is defined as a symbolic reference in XPDL which has to be assigned to concrete applications by the WfMS.

OrgML:Software	XPDL:Application
	Id
name	Name
description	Description
	Formal Parameters
	Extended Attributes
	External Reference
systemRequirements	
scalability	

Figure 22: Correlation between Software and Application

[59] Cf. [Nori02, p. 44]
[60] Note the difference between resource-attributes and extended attributes: A resource-attribute is a name-type-pair and an extended attribute is a name-value-pair.

The resource's (software) properties name and description can be directly mapped to an XPDL-document. The properties systemRequirements and scalability will be neglected because they have no correspondence in XPDL. XPDL requires a unique identifier, which cannot be expressed in MEMO-OrgML. However, such an identifier could either be explicitly assigned to OrgML models (which would require a small extension of the language) or generated within the automatic mapping of OrgML to XPDL. External attributes and an external reference are equally handled like the same attributes in section 4.2.1. They have to be complemented using an additional abstraction (regarding workflows). The attribute "Formal Parameters" correspond to a list of single formal parameters which are specified using the following properties:

- Id: Identifier
- Data Type: Type of the formal parameter
- Description: Textual description of the formal parameter
- Index: Position in the parameter list
- Mode
    - IN: read-only parameter
    - OUT: write-only parameter
    - INOUT: Parameters used as in- and output parameter

The Id of a formal parameter has to be unique within the namespace of a process.

4.2.3 Information

Regarding business process modelling, the description of information usually corresponds to the specification of information types which are used within a business process. In contrast to this, workflow-relevant data is associated with variables containing concrete information[61]. Such variables are generally referenced by a unique name (as identifier) and correspond to a given type.

OrgML:Information	XPDL:Data
	Id
name	Name
description	Description
typeDescription	Data Type
	Extended Attributes
	Initial Value
	Is Array
	Length

Figure 23: Correlation between Information and Data

The appropriate Id for a workflow specification might be generated automatically. Name, description and data type can directly be resolved from the corresponding attributes of the MEMO-process model (cf. Figure 23). Extended attributes have to be handled the same way as extended attributes of human resources and applications. Additionally, an initial value might be assigned to a variable. The maximum length and the property of being a collection can be determined by the attributes Is Array (the variable is a multi-valued type) and Length (upper bound of a sequence).

4.3 Processes

MEMO-OrgML supports several different kinds of process types, which have to be mapped to appropriate (i.e. similar) concepts given in the XPDL. Regarding MEMO-OrgML, there are

[61] The terms 'data' and 'information' are used synonymously within this report.

concepts like composed, manual, automated and semi-automated processes. In XPDL, there are generic and block activities. Conceptual relationships between MEMO-OrgML-processes and XPDL-activities are subject to this section.

4.3.1 Manual Processes

Manual processes (MEMO-OrgML) are executed only by human resources without any IT-support. Hence, those processes seem to be irrelevant for the execution of a workflow schema. Nevertheless, there are several different alternatives regarding the mapping of manual processes to XPDL-schemas.

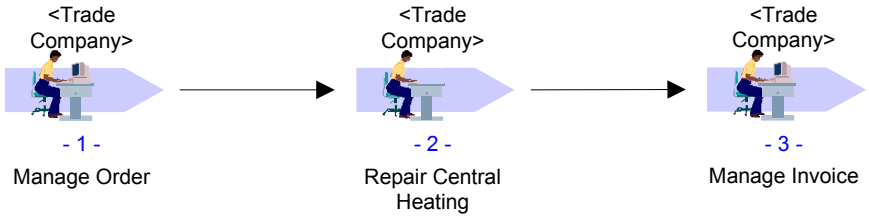

Figure 24: Example for manual processes

Alternative 1: Manual Processes are not mapped to workflow activities
This alternative ignores every manual process in the context of workflow management. Consequently, every manual process has no counterpart in the XPDL-based specification. The basic assumption is the fact, that a manual process only has to be executed by a human being and no IT-support is involved (say: a WfMS will not be needed). With respect to the example given in Figure 24, only the processes No. 1 and three are mapped to the XPDL-based specification and the process No. 2 will be dropped.

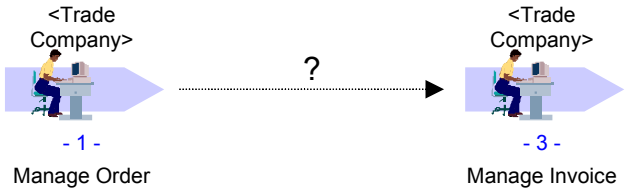

Figure 25: Example for alternative 1

As shown in Figure 24 the execution of a virtual order management process consists of the management of an incoming order, the execution of the order and the management of the invoice. In the original process in Figure 24 the termination of process No. 2 triggers the beginning of process No. 3. According to the simple mapping strategy (ignoring manual processes) information on the change of state will be neglected. The process model given in Figure 25 leaves out the manual process No. 2 in Figure 24.

Alternative 2: Mapping of manual processes to route activities
As stated in section 3.2.2, a route activity is an activity without an implementation. It has neither a performer nor an application. It also has no effect on the workflow or workflow-relevant data (as well as application data). Keeping these restrictions in mind a manual process can be mapped to a route-activity if its execution has no effect on the succeeding processes and the human resource has no equivalent in the WfMS. This alternative is tainted with the fact that the rout-activity's original purpose results from the modelling of cascading splits and joins.

Alternative 3: Mapping of manual Processes to workflow activities
Alternative No. 2 comprises the mapping of manual processes to workflow activities. Every manual process will be mapped to an activity with a human actor and no applications. Consequently, manual processes correspond to some kind of dummy-activities in a workflow-

schema. Such a dummy-activity only serves for the recognition of the completion of a manual process. The termination of a manual process might trigger the start of a following process. As an effect, every manual activity has to be to be confirmed in the WfMS.

If alternative 1 has been selected there will be no mapping of a manual process to a workflow activity. All manual processes as well as their associated transitions will be lost in the workflow model. Choosing alternative 2 will at least keep the information about the existence of a manual process in the business process model. But the human resource responsible for the execution of a manual process can not be associated with a route activity. The last alternative allows the mapping of the process itself and according participants (human resources) to the workflow model. The mapping corresponding to alternative 3 is shown in Figure 26.

OrgML:Manual Process	XPDL:Generic Activity
Id	Id
name	Name
description	Description
Organisational Unit	Performer
	Transition Restrictions - Join: AND, XOR - Split: AND, XOR

Figure 26: Mapping of manual processes

The Id will be generated out of the manual process' identifier and the attributes name and description can be directly mapped from the process definition to the XPDL-document. The organisational unit of a manual process can be mapped to the performer of the workflow activity. The transition restriction specifies whether all incoming transitions (Join) are synchronised (AND) or alternatives (XOR) as well as all outgoing transitions (Split) are parallel (AND) or alternative (XOR)[62]. This information is determined by the kind of in- and outgoing arcs of the business process models. A parallel split will be mapped to an AND-split and an alternative split to an XOR-split. Analogously, a parallel join is mapped to an AND-join and an alternative join to an XOR-join. Information which is not included in the business process model but necessary for a workflow activity's specification[63] is shown in Figure 27.

Attribute	Description
Deadline	Specification of a deadline and an action to be taken if it is reached.
Documentation	Identifier of an external documentation file (e.g. URL or a filename).
Start Mode	**Manual Mode**: The user has to start the activity manually (indicating the beginning of his work)
Finish Mode	**Manual Mode**: The activity has to finishes according to a user's interaction (indicating the end of his work).
Implementation	**No**: Implementation by manual procedures
Icon	Reference to an external file containing an image for the representation of an activity.
Limit	Expected maximum duration for the execution of a process (vendor-specific)
Priority	A value describing the initial priority of a process.
Simulation Information	Estimations for the simulation of an activity.

[62] See also section 3.2.2.
[63] Words given in a bold style correspond to concrete values.

Figure 27: Attributes of a manual activity[64]

As most of the workflow-relevant data is not available in a business process model, there has to be an extension to those models. We emphasise a so called workflow-diagram for every business process which has to be mapped to a workflow schema. This diagram contains all additional information for the implementation and simulation of business processes in a workflow-environment.

4.3.2 Semi-Automated Processes

Regarding the MEMO-OrgML, semi-automated processes are executed by human resources using IT-related resources (soft- and hardware). Hence, semi-automated processes rely on human and software-technical resources. Those kinds of processes can be mapped to generic activities. Route activities as well as block activities have no implementation and correspond to routing conditions or the execution of an embedded flow. A route activity is no real activity. It cannot be associated with a concrete task and exists only for reasons regarding control flow in workflow applications. A block activity has no inherent implementation as such an activity only calls an embedded activity set.
The mapping of general information on a semi-automated process is equivalent to the one of a manual process as given in section 4.3.1 (cf. Figure 26). Most workflow-specific information can be generated using a workflow-diagram. Nevertheless, Start- and Finish-Mode as well as the process' implementation are determined

Attribute	Description
Start Mode	**Manual Mode**: The user has to start the activity manually (indicating the beginning of his work)
Finish Mode	**Manual Mode**: The activity has to finishes according to a user's interaction (indicating the end of his work).
Implementation	**Tool:** Implementation is supported by an applications

Figure 28: Attributes of a semi-automated activity[65]

4.3.3 Automatic Process

An automatic process is executed without the intervention of a human participant. Nevertheless, an organisational unit can be assigned to an automatic process, too. That means that it is responsible for the execution of this process. Basically, the mapping of automated processes can be handled like a manual process (cf. Figure 26). In contrast to a manual or semi-automatic process the assignment of a performer has no effect on the execution of an automatic process.

Attribute	Description
Start Mode	**Automatic**: Triggered by the system.
Finish Mode	**Automatic**: Triggered by the system.
Implementation	**Tool:** Implementation is supported by an applications

Figure 29: Attributes of a automatic activity[66]

4.3.4 Aggregated Process

Aggregated processes in MEMO-OrgML have no inherent implementation but consist of other processes. A block-activity in XPDL corresponds to a set of sub-activities and has no

[64] Source: [Nori02, p.31]
[65] Source: [Nori02, p.31]
[66] Source: [Nori02, p.31]

own resources. We will not use subflows, because every subflow contains its own set of performers and tools. As there is no support of name-spaces (in OrgML) we will not support the concept of sub flows. Hence, every aggregated process will be mapped to a block-activity and all contained processes are collected into an activity set.

4.4 Control Flow and Events

The type of control flow is determined by the workflow activity's definition. All outgoing transitions of an activity are either parallel (AND) or alternative (XOR). Equally like, every join (incoming transitions) is either a parallel or alternative synchronisation. The specification of joins and splits is associated with a process' definition.
Events do not exist as specific concepts in XPDL. Hence, there is no direct correspondence between events in a business process model and a workflow-schema. Nevertheless events given in a business process model can be used for the definition of conditions on the firing of a transition. As given in Figure 30 an event's name and description can be mapped to an XPDL-description. The Id can be constructed from an event's Id and context. The condition has to be complemented to the activities description. Preceding and succeeding activities can be derived from the business process model.

OrgML:Event	XPDL:Transition Information
Id	Id
Name	Name
Description	Description
	Condition
	From (set of activities)
	To (set of activities)

Figure 30: Transitions

4.5 Data

Workflow-relevant data has to be specified in XPDL. There are two general usages of information in a workflow's context: the definition of a variable and its usage. The definition of an information-related variable consists of its name and type. The usage refers to its name and assignment of new values.

OrgML:Information	XPDL:Data
	Id
name	Name
description	Description
dataType	Data type
	Initial Value

Figure 31: Mapping of data

5 Prototypical Tool-Support

The mapping of business process models to XPDL-documents is the basis for the development of a prototype for software-generation. Goal of this development has been a tool for the generation of a software-system basing on business process models. The underlying vision is the automatic generation of software (i.e. executable programs) out of models. To achieve this goal, we used the following approach:

1.) Creating business process models using MEMO-OrgML
2.) Extending the business process models by workflow-relevant information
3.) Mapping the BPM to an XPDL-document
4.) Executing the processes on the basis of the XPDL-document using a Workflow-Engine
    a. Importing the XPDL-description into the WfMS
    b. Customisation of the WfMS

The following tools were used to achieve this goal:

- MetaEdit+ 4
- Shark Workflow Engine by Enhydra

5.1 Implementation of MEMO-OrgML using MetaEdit+

We implemented a MEMO-OrgML modelling tool using the meta-modelling-tool *MetaEdit+*. MetaEdit+ is a tool for the development of modelling tools and currently available in version 4. Basic concepts of MetaEdit+ are objects, relationships, roles and diagrams. Objects represent the nodes within a diagram and relationships the edges between objects. Roles specify additional information on the appearance of an object in a given relationship[67]. Proper combinations of objects via relationships using roles are defined in diagrams. The two most important diagram types (MEMO-OrgML) have been implemented:

a) Decomposition of processes
b) Process models

5.1.1 Process Decomposition Diagrams

A process-decomposition-diagram expresses decomposition-relationships between processes. Every composed process consists of several elementary and/or composed processes. Every composed process has the role of a *composite* in the context of decomposition. Every subordinated process (with respect to a decomposition-relationship) plays the role of a *part*. A part can either be another composed process or an elementary process. A composed process used in a process-decomposition-diagram can be further specified by a process-model-diagram (cf. section 5.1.2). Such a diagram can be connected to a given model-element by a so called *explosion*. An explosion is a concept given in MetaEdit+ and allows for the connection of an object in a diagram to another diagram. This concept can be used to connect a composed process with a process-model-diagram.
An example for a decomposition diagram is given in Figure 32. The process Invoice Processing (Id: Pay) is a composite consisting of the two processes Check Invoice (Id: Pay_1) and Pay Liability (Id: Pay_2). Pay_2 is a composite and Pay_2.1 and Pay_2.2 are its parts. The control flow of process Pay_2 is shown in Figure 33 and explained in the following section 5.1.2.

[67] The concepts of MetaEdit+ are not further discussed within this introduction. We will rather refer to specific concepts while presenting the implementation of MEMO-OrgML.

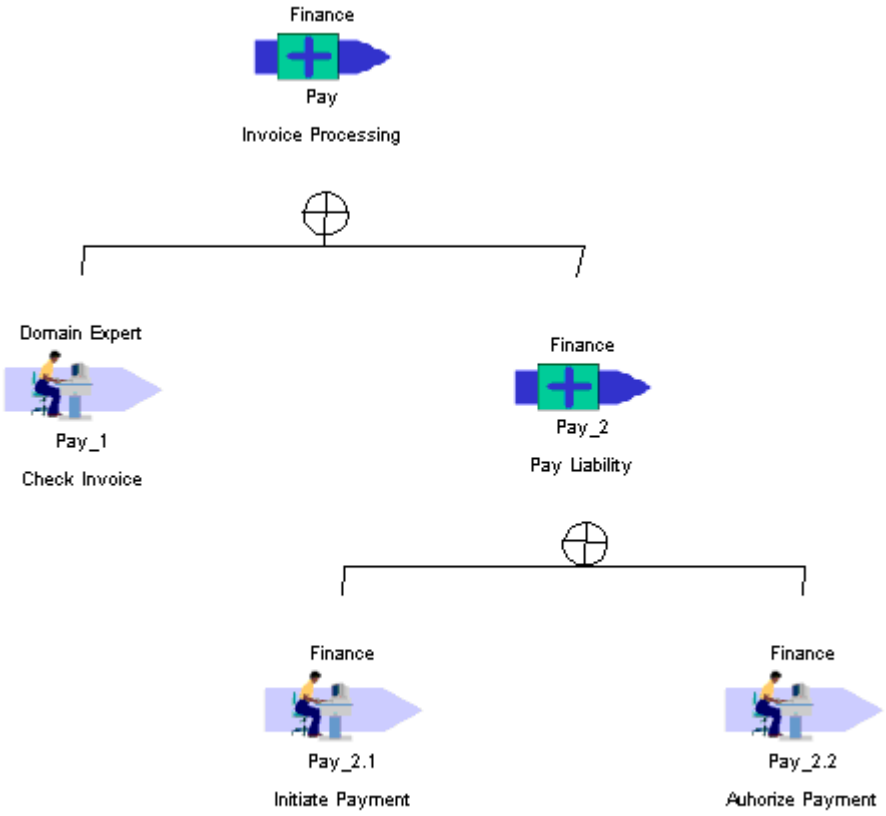

Figure 32: Example Process Decomposition Diagram as realised in MetaEdit+

The following concepts are used in a decomposition-diagram:

- Objects:
  - Composed Process
  - Elementary Process
    - Manual process
    - Automated process
    - Semi-automated process
- Relationships
  - Decomposition
- Roles
  - Composite
  - Part
- Explosions
  - Composed Process → Process-Model-Diagram

Every composed process participating in a decomposition relationship can play the role of a composite. Every composed and every elementary process can play the role of a part with respect to a decomposition in relation to a composite. A composed process can be associated with a process-model-diagram.

5.1.2 Process Model Diagrams

A process-model-diagram consists of events and processes and specifies logical/temporal relationships between business processes[68]. Possible control-flow-types are the sequence, alternative and concurrency. An example for a process-model-diagram is given in Figure 33. This diagram represents the control flow of the parts of process Pay_2 presented in the previous section 5.1.1. The process is started by event ePay_7 which results in the execution of process Pay_2.1. After its completion either the event ePay_12 (with a probability of 10%) is fired or event ePay_14. Event ePay_12 triggers the execution of process Pay_2.2 which in turn fires -- after its termination -- event ePay_14.

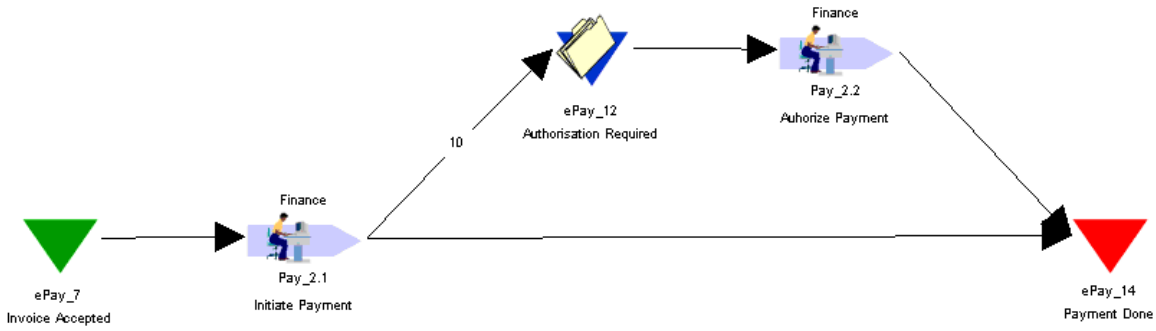

Figure 33: Example Process Model Diagram

Objects used in a Process Model Diagram comprise all kinds of processes and all event-types (refer to sections 2.2.1 and 2.2.2). They are connected by all types of control-flow given in MEMO-OrgML (refer to section 2.2.3). The most important objects are listed below:

- Objects:
  - Composed Process
  - Elementary Process
    - Manual process
    - Automated process
    - Semi-automated process
  - Event
    - Start
    - Stop
    - Incoming Message
    - Information Change
- Relationships
  - Sequence
  - Alternative Split and Join
  - Parallel Split and Join
- Roles
  - Predecessor
  - Successor

Objects are connected by the Sequence-, Alternative- or Parallel-relationships. The starting object of such a relationship plays the role of a predecessor and the ending object the one of a successor. There are no explosions defined between objects of a process-model-diagram and other diagram types. Example: The connecting arc between event ePay_7 and process

[68] These relationships have been presented as 'control flow' in section 2.2.3.

Pay_2.1 in Figure 33 defines a sequence between the objects and has the event as predecessor and the process as successor. Nevertheless, not all connections between arbitrary objects are useful (e.g. a sequence-relationship between a start- and a stop-event). Hence, there are some restrictions on the design of a process-model-diagram:

Start
- A start-event has no predecessor.

Stop
- A stop-event has no successor.

Incoming message
- An incoming-message-event has no predecessor and one or more processes as successor.

Sequence
- Every event is followed by a process and every process is followed by an event

AND-Split
- One event is followed by several (at least two) processes

AND-Join
- One process is preceded by two or more events

XOR-Split
- One process is followed by several (at least two) events

XOR-Join
- One event is preceded by two or more processes

5.2 Extending Business Process Models with Workflow-specific information

As shown in sections 2.2 and 3, there are many differences between MEMO-OrgML and XPDL. In order to map a business-process-model to an XPDL-document, missing information has to be added to the business-process-model. Two diagram-types have been added to achieve this goal: workflow-specification-diagram and workflow-activity-specification-diagram.

5.2.1 Workflow Specification Diagram

A workflow-specification-diagram supplements a business-process-model with workflow-related abstractions. This diagram-type contains objects of the following types:

Workflow Process
A workflow-process contains a reference to a corresponding business-process. Usually, the corresponding business process is a composed process which in turn consists of other business processes and their control-flow. Business processes are mapped to XPDL-activities according to the rules given in section 5.3. Additionally, a link to the documentation of a business process and the annotation of extended attributes is possible.

Workflow Participant
A workflow-participant is an XPDL-specific concept and determines the actor of a workflow. Such a participant is associated with an organisational unit in MEMO-OrgML and is supplemented with extended attributes and a participant-classifier. Possible classifiers are the ones given in section 3.3.1.

Workflow Application
An application is the specification of a workflow-related tool. Such a tool can either be an internal procedure or an external application (refer to section 3.3.2). The application-object in a workflow-specification specifies a unique identifier, name and formal parameters of a workflow-application.

Workflow Information
A workflow-information-object specifies information used in a workflow and can be seen as a variable. This specification consists of a unique identifier, a name and a data-type as well as a default-value and other XPDL-related fields (refer to section 4.2.3).

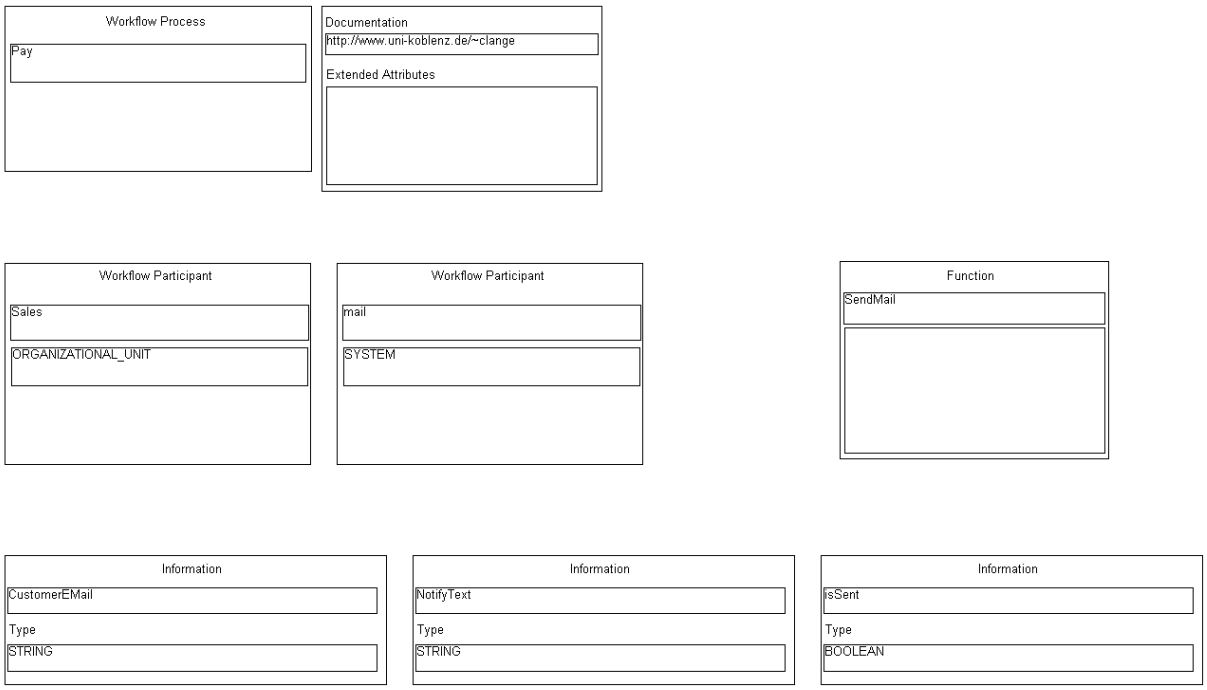

Figure 34: Example Workflow-Specification-Diagram

An example of a workflow-specification-diagram is given in Figure 34. This diagram is the workflow-specification of process Pay presented in section 5.1.1. This association is established by the top-left box called Workflow Process representing the business process Pay. To the right of this box is a link to the process' documentation and a list of extended attributes (empty in Figure 34). There are two workflow-participants and an application (Function) defined for the workflow. The Sales-participant corresponds to the organisational unit Sales and has as type Organisational_Unit. The participant called mail corresponds to a mail-sender and is a software-system for the sending of mails. Consequently, there is also an application for the delivery of e-mails. Every information (variable) used in a workflow has to be specified in a workflow-specification-diagram. According to the example given in Figure 34 there are three variables containing the e-mail-address of a customer (type String), a default-text for the notification of a customer (String) and a Boolean variable indicating whether a message has been send or not.

5.2.2 Workflow Activity Specification Diagram

A workflow-activity-specification-diagram is associated with an elementary process (using an explosion). The workflow-activity-specification describes the workflow-application as well as actual parameters used for the execution of such a process. Generally speaking, a workflow-activity-specification-diagram associates a process with an application and the assignment of actual parameters. Hence, an activity-specification associated with a process binds an application to workflow information. Additionally, it is specified whether an application is an internal procedure or an external tool.

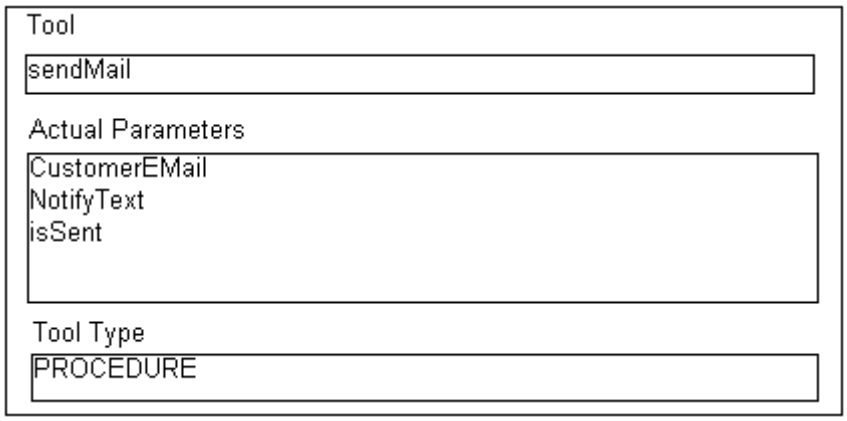

Figure 35: Example Workflow-Activity-Specification

The example given in Figure 35 assigns actual parameters to a procedure for the sending of e-mails (tool called: sendMail). Corresponding data are the e-mail-address of customer, a notification text and an internal Boolean status variable. The Boolean variable is set to true if the mail has been sent and false otherwise.

5.2.3 Summary of Business- and Workflow-Diagrams

The actual implementation of the mapping of business process models to a running application consists of four different diagram types. A process-decomposition-diagram specifies the relationships between composed and elementary processes. Every composed process can be further specified by a process model and every elementary process can be further specified by workflow-relevant data. The relationships between the different kinds of diagrams used for the mapping of business process models to workflows are shown in Figure 36.

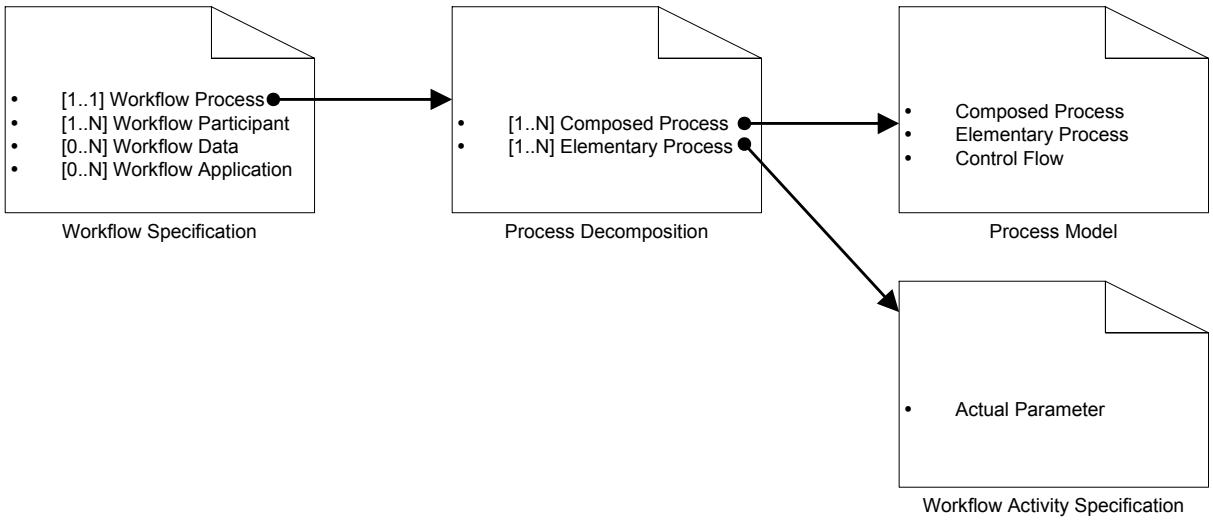

Figure 36: Structure of Diagram and Object Types

Root-diagram for the generation of an XPDL-conformant workflow-specification is a workflow-specification diagram. It contains all participants (one or more), workflow-data (zero or more) and applications required (zero or more) for the workflow-process. The workflow-specification-diagram also contains a reference (realised with an explosion) to a process-decomposition containing all (sub-)processes used within the given workflow process. The root process of the decomposition-hierarchy (mandatory) represents the workflow process itself and additional decomposed processes are possible. At least one elementary process has to be assigned to each composed process in the decomposition-

diagram using a decomposition-relationship. Additionally control flow of every decomposed process has to be specified using a process-model-diagram. This diagram has to include all sub-processes (connected via a decomposition-relationship) of the given composed process. Every elementary process in the process-decomposition-diagram has to be further described using a workflow-activity-specification-diagram. This diagram contains the binding of workflow-relevant data (actual parameters) to the formal parameters of a workflow-application.

Example:
The workflow-specification-diagram for process 0, Notify Customer in is shown in Figure 36. The diagram contains the workflow-process' participants, data and applications. The associated process-decomposition-diagram is given in Figure 37, containing the process 0 itself as well its sub processes 1 and 2.

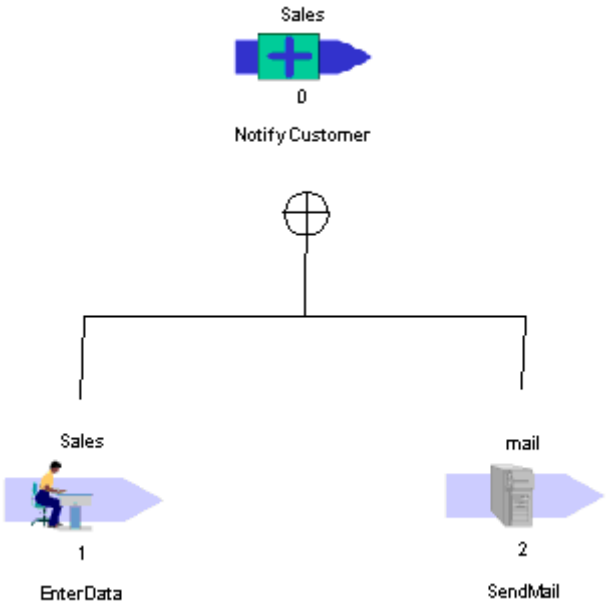

Figure 37: Decomposition of Workflow-Process 0

Figure 38 shows the process-model-diagram according to process 0. The tool used for the execution of process 2 SendMail is specified in Figure 35.

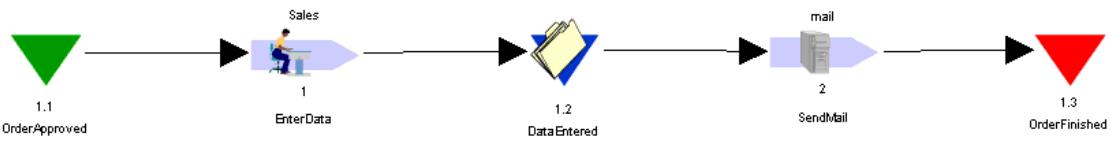

Figure 38: Process-Model of Workflow-Process 0

5.3 Mapping of OrgML-Models to XPDL-Documents

The mapping of MEMO-OrgML-models to XPDL-conformant workflow-definitions is realised using the code-generation mechanisms of MetaEdit+. MetaEdit+ includes a language for the specification of mappings between internal models and external textual specifications.

5.3.1 Workflow Process Specification Headers and Packages

Every workflow-process-specification is mapped to one XPDL-file. The header of the XML process-definition-language-based file starts with the XML-header (determining the XML-version and character-encoding):

```
<?xml version="1.0" encoding="UTF-8"?>
```

The WfMC recommends the generation of at least one package for each XPDL-file[69]. The generic default-package-header containing meta-information ob the XPDL-definition is given as follows:

```
<Package xmlns="http://www.wfmc.org/2002/XPDL1.0"
  xmlns:xpdl="http://www.wfmc.org/2002/XPDL1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:iwvi="http://iwvi.uni-koblenz.de/workflow"
  xsi:schemaLocation="http://www.wfmc.org/2002/XPDL1.0
  http://wfmc.org/standards/docs/TC-1025_schema_10_xpdl.xsd"
  Id="0" Name="ECOMOD Process">
```

This generic header specifies XML-related information like the XML-document-type-definition (http://www.wfmc.org/2002/XPDL1.0) as well as context-specific information like an ID (0) and a package's name (ECOMOD Process).

Following the generic header, a specific package header is written to the output-file. This header includes the version of the corresponding XPDL-specification, a vendor-id and the creation-date of the file.

```
<PackageHeader>
  <XPDLVersion> 1.0</XPDLVersion>
  <Vendor> IWVI, UNi Koblenz </Vendor>
  <Created>'; 'April 2004'; '</Created>
</PackageHeader>
<ConformanceClass GraphConformance = "NON_BLOCKED"/>
<Script Type="text/javascript" />
<TypeDeclarations/>
<Participants/>
<Applications/>
<DataFields/>
```

The header-specification is followed by the annotation of a conformance class, a script type, type declarations, participants, applications and data fields. The conformance class might be one of the following[70]:

Conformance Class	Description
NON_BLOCKED	There is no restriction on the network structure. This is the default-value.
LOOP_BLOCKED	The network structure is restricted to proper nesting of loops.

[69] Cf. [Nori02, p. 19]
[70] Cf. [Nori02, p. 22]

FULL_BLOCKED	The network structure is restricted to proper nesting of SPLIT/JOIN and loops.

The conformance-class NON_BLOCKED is chosen for the mapping of process models to XPDL because it is the least restricting. The script-field has been filled with the entry 'javascript' as an example. This entry will never been used within the XPDL-mapping. All other fields regarding packages will be left open. Those fields will never be used during the mapping and afterwards. Hence, no participants, applications or data will be declared on package-level.

5.3.2 Workflow-Specification

Every generated XPDL-file consists of exactly one package-definition and one workflow-definition included in this package. This workflow-definition consists of the following sub-definitions:

- data-fields
- participants
- applications
- activity-sets
- activities (cf. section 5.3.3)
- transitions (cf. section 5.3.4)

The specification of data-fields, participants and applications is derived from the workflow-specification diagrams. Afterwards, the activity-sets are generated basing on the workflow-specification-diagram. There will be an activity-set for every composed process in the decomposition-diagram (except for the root-process). The root-process is mapped to the package specification and the workflow-process-specification. Every composed process which is part of a decomposition of the root process will be mapped to an activity set. Each activity-set contains the herein included processes as well as their transitions. According to Figure 32 the process Pay will be mapped to a package- and workflow-process-definition. The herein contained process Pay_2 will be mapped to an activity-set.

5.3.3 Activities

After the generation of the headers, the workflow-process' data-fields, participants and applications as well as activity-sets, MEMO-OrgML processes are mapped to workflow-activities. Every process (composed or elementary) will be mapped to exactly one activity using the direct mapping given in Figure 26 and following the rules given in Figure 39. A manual process is performed by a human being and requires no IT-related resource. Hence, the corresponding activity-specification requires a workflow-participant and does not allow the annotation of a workflow-application. The workflow-activity has to be started manually (indicating the beginning of the execution of the manual process) and finished manually (end of process) as well. A semi-automated process also requires a participant but a workflow-application has to be specified, too. It has to be started and finished by the human participant. An automated process will be started and terminated automatically (by the WfMS). It does not necessarily need a participant (it can be annotated) but requires a workflow-application. A block-activity contains a set of activities (an activity-set) and does not require a participant or an application. Hence, a composed process is mapped to a block activity (as well as to an activity set containing its subsequent processes).

Process Type (MEMO-OrgML)	Specification in XPDL
Manual process	Participant: required Application: not applicable Start-mode: manual Finish-mode: manual
Semi-automated process	Participant: required Application: required Start-mode: manual Finish-mode: manual
Automated process	Participant: not required Application: required Start-mode: automatic Finish-mode: automatic
Composed process	Participant: not required Application: not required Start-mode: not applicable Finish-mode: not applicable

Figure 39: Mapping of processes to activities

Control flow between activities is specified in XPDL using transitions (discussed in the following section 5.3.4) and transition-restrictions assigned to a workflow-activity. If a process is preceded by an alternative join the join-transition-restriction of the corresponding workflow-activity is set to XOR. If it is preceded by a parallel join its join-transition-restriction is set to AND. Analogously, the activity's split-restriction will be set to XOR/AND if the process is followed by an alternative/parallel split. Note that the join- and split-types regarding a business process cannot always be determined by its directly preceding and following relationships (see the following section).

5.3.4 Transitions

A transition between two workflow activities corresponds to a followed-by relationship. A transition starting in activity A and ending in activity B means, that activity B can be started after the termination of activity A. Information about the kind of control-flow is not part of a transition-specification. Nevertheless it will be part of the activities' specification. Another conceptual difference between process-models in MEMO-OrgML and XPDL-activities is the absence of events in a workflow-specification. The mapping of MEMO-OrgML's control-flow to XPDL-transitions will be explained using an abstract example in Figure 40[71].This example consists of four processes and three events. The process called A is followed by event No. 10 which in turn results in the parallel (or concurrent) execution of processes B and C. The termination of process B is connected to the occurrence of event No. 11 and the one of C to the occurrence of 12. If both events occurred (parallel join), process D can be started.

[71] Please note that we used a different graphical notation for parallel splits and joins compared to the definition of OrgML presented in section 2.2.3.3. MetaEdit+ does not allow the differentiation between relationship types by different routings of lines.

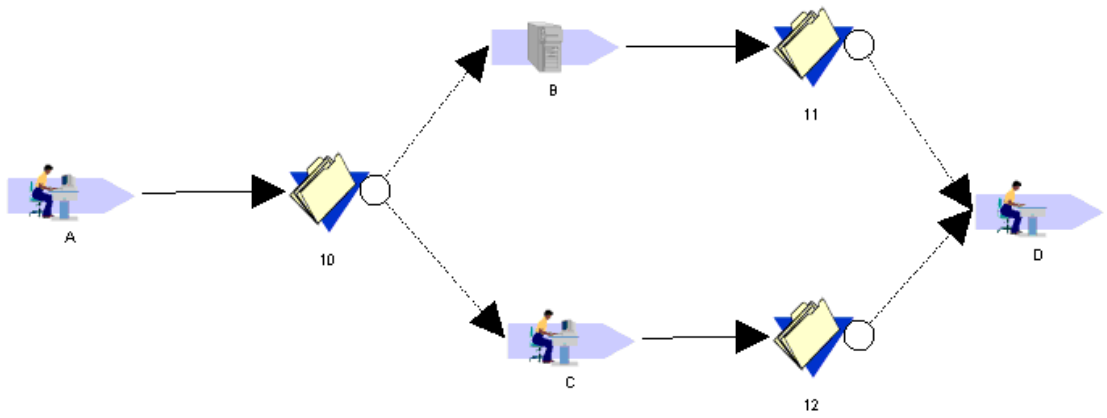

Figure 40: Example Containing a Parallel Join and Split

The generation of transition bases on a simple algorithm: Every MEMO-event will be mapped to at least one transition. Their will be one transition for every combination of starting (from) and ending (to) points. Event No. 10 given in Figure 40 will for example be mapped to two transitions: One transition from A to B and one from A to C. The unique identifier of each transition is generated by concatenating the identifier of the preceding process, the event's id and the identifier of the succeeding process. The information about the type of control-flow is associated with the preceding process. The split-transition-restriction of process A is set to AND. The events 11 and 12 will be mapped to one transition each and the join-transition-restriction of process D is set to AND.

5.4 Configuration of the WfMS

An open-source workflow-engine[72] implemented in the programming language Java has been used for the execution of the XPDL-files. The Shark-engine fully supports the XPDL-standard of the WfMC and provides the association of workflow-participants with concrete users (section 5.4.1) as well as the association of workflow-applications (in XPDL) with procedures (section 5.4.3).

5.4.1 Mapping of Participants to users

Workflow-participants can be declared using the XPDL-concepts presented in sections 3.3.1 and 5.2.1. Different types of participants are

- Resource (also: resource-set; a set of resources)
- Role
- Organizational_Unit
- Human
- System

The Shark-engine provides a mechanism for the association of participants with specific users of the WfMS. This is applicable if the type of the participant is resource (resource-set), role or organisational unit. Depending on the context, the role of an escalation coordinator for example might be assigned to a specific user and reassigned to another user after the reorganisation of the company. If a participant cannot automatically be mapped to a Shark-user, the activity will – per default – be assigned to the administrator. If the type of participant

[72] The workflow-engine is called *Shark* and has been developed by *Enhydra*. It is available on the web-page of enhydra: http://shark.enhydra.org

is Human the participant will be mapped to a WfMS-user with the same name. For example: A workflow-participant of the type Human and with the name jjung will be associated with the local user (say: in the Enhydra Shark-engine) named jjung. If the participant's type is System, the actor is assumed to be a software-system and the activity is performed by a workflow-application. Hence, the association between workflow-participants and WfMS-users generally has to be defined by an administrator. Such an association has to be omitted for human participants and software systems. In the latter cases an automatic assignment is provided by the workflow-engine.

5.4.2 Updating Workflow-Data

The XPDL-standard provides no mechanism for the manipulation of workflow-relevant data except for the execution of workflow-applications. The Shark-engine allows for the manipulation of data by using extended attributes in the context of workflow-activities. The relevant extended attributes are listed in Figure 41. The first attribute specifies data which can be manipulated in a workflow-activity and the second one only allows for the display of a variable's value. The symbolic identifier <workflow-data> has to be replaced by a specific workflow-data's name.

attribute-name	attribute-value
VariableToProcess_UPDATE	<workflow-data>
VariableToProcess_VIEW	<workflow-data>

Figure 41: Extended Attributes of the Shark-Engine

5.4.3 Mapping of Workflow-Applications to Procedures/Applications

A workflow-application is specified in an XPDL-file by an application-declaration (cf. section 3.3.2). Such a specification only includes a symbolic reference to a concrete application. Generally, the WfMC distinguishes between a procedure (an application implemented within the WfMS) and an application (an external piece of software). The current version of the Shark-engine only supports internal procedures implemented in Java. The association of a workflow-application to a procedure is provided by the Shark-engine. Every application included in an XPDL-specification has to be associated with a Java class which is under the control of the Shark-engine. Such a Java-class has to be placed in the storedprocedure-directory of the Shark-installation and has to implement a public static method called execute without a return-parameter. Examples for prototypical workflow-applications are given in the following section 5.4.4.

5.4.4 Example-Implementation of Prototypical Workflow-Applications

This section presents some prototypical implementations for workflow-applications. In XPDL applications are only specified on an abstract level. Such a description mainly consists of a symbolic name and a list of formal parameters. There is no reference to a specific application. The mapping of such an application definition to a concrete application has to be done using features of the used WfMS. The Shark-engine supports the integration of Java-based applications. Prototypical implementations including the sending of e-mails, the composition of textual documents as well as the editing of existing documents have been developed for the generation of executable software on the basis of business process model.
A commonly used feature of corporate information systems is the notification of a customer sending an e-mail. A Java-based application using an external mail-tool has been developed for automatically sending an e-mail message to a customer after the termination of a process. The XPDL-specification of this application is presented in Figure 42, its implementation in Figure 43 and the mapping of workflow-data to formal parameters in Figure 44.

```
<Application Id="1" Name="SendMail">
 <FormalParameters>
  <FormalParameter Id="receiver" Index="0" Mode="IN">
   <DataType>
    <BasicType Type="STRING"/>
   </DataType>
  </FormalParameter>
  <FormalParameter Id="text" Index="1" Mode="IN">
   <DataType>
    <BasicType Type="STRING"/>
   </DataType>
  </FormalParameter>
  <FormalParameter Id="isSent" Index="2" Mode="OUT">
   <DataType>
    <BasicType Type="BOOLEAN"/>
   </DataType>
  </FormalParameter>
 </FormalParameters>
 <ExtendedAttributes>
 </ExtendedAttributes>
</Application>
```

Figure 42: XPDL-specification of the e-mail-sender

The XPDL-code in Figure 42 defines the abstract e-mail-application including its parameters. Id and name of the application are given in the first line. The set of formal parameters is listed between <FormalParameters> and </FormalParameters>. Every formal parameter is described by its identifier, an index, a mode and a data-type. The name is a character string defined by the user. The index is the position of the parameter in the parameter-list. The mode specifies whether the parameter is passed from the WfMS to the application (IN), returned from the application (OUT) or both (INOUT). The data-type might of any type defined by the WfMC. We currently only support the basic types (String, Float, Integer, DateTime, Boolean) given by the WfMC.

The implementation of the e-mail-sender is listed in Figure 43. The Shark-workflow-engine assumes a Java-class with a class-method called execute as the implementation of a workflow-application. The parameter-list of the method must correspond to the formal parameter-list of the XPDL-specification of the workflow-application. The data-type of the parameters has to be Any meaning an arbitrary data type. The parameters are assigned to typed attributes using mapping methods of the Any-type. The prefix of those methods is extract and there are two examples (extract_string()) in Figure 43. After the definition of default-parameters and the extraction of concrete values from the parameters, a command-line-string for the execution of a mail-sender is constructed. This mail-tool is not implemented using Java but a Windows command-line tool. Hence, it cannot directly be started from a Java-class. The class Runtime is used to call the application and the Java-based workflow-application will wait until its termination. After the execution of the tool a Boolean value indicating the successful or unsuccessful sending of the mail is inserted in the OUT-parameter.

```
package wfmapps;

import java.io.IOException;
import org.omg.CORBA.Any;
import org.omg.CORBA.BAD_OPERATION;

public class WfSimpleMailSender {
	public static void execute(Any aReceiver, Any aText, Any isSent){
		try {
			/*Default parameters:
			 * host: mailserver for sending mails
			 * from: email-address of the sender
			 * name: name of the sender
			 * subj: subject*/
			String host = "mailhost.uni-koblenz.de";
			String from = "ecomod@uni-koblenz.de";
			String name = "ECOMOD WfMS";
			String subj = "Message from ECOMOD WfMS";

			/*Passed parameters:
			 * receiver: email-address of the receiver
			 * text: message body*/
			String receiver = aReceiver.extract_string();
			String text = aText.extract_string();

			//Construct command-line for external mail-tool
			StringBuffer command = new StringBuffer("c:\\xpdl\\apps\\mail\\netmailbot");
			command.append(" -to "); command.append(receiver);
			command.append(" -from "); command.append(from);
			command.append(" -fromfriendly "); command.append("\""+name+"\"");
			command.append(" -subject "); command.append("\""+subj+"\"");
			command.append(" -server "); command.append(host);
			command.append(" -body "); command.append("\""+text+"\"");

			//Call external mail-tool
			Runtime rt = Runtime.getRuntime();
			rt.exec(command.toString());
			/*The mail is assumed to be sent
			 * if no error occured*/
			isSent.insert_boolean(true);
		} catch (Exception e) {
			e.printStackTrace();
		}
	}
}
```

Figure 43: Implementation of an E-Mail-Notification

The assignment of workflow-data to formal parameters of an e-mail-application is shown in Figure 44. The XPDL-excerpt is part of the specification of the process regarding the e-mail-based notification of a customer specified by process No. 2 shown in Figure 37. The ordering of actual parameters determines the mapping to a formal parameter with an index corresponding to the actual parameter's order.

```
<Implementation>
  <Tool Id="1" Type="PROCEDURE">
    <ActualParameters>
      <ActualParameter>CustomerEMail</ActualParameter>
      <ActualParameter>NotifyText</ActualParameter>
      <ActualParameter>isSent</ActualParameter>
    </ActualParameters>
  </Tool>
</Implementation>
```

Figure 44: Mapping of actual parameters to an application

6 Summary and Future Work

This research report presents one possible implementation for the generation of an information system out of process models. The implementation bases on the mapping of business process models to a workflow model and the appropriate configuration of a WfMS. The overall vision including used software and corresponding documents as well as languages is shown in Figure 45. MetaEdit+ has been used for the modelling of business processes (using MEMO-OrgML) and their mapping to workflow-descriptions (using XPDL). The Shark workflow-engine uses such a workflow-description for the execution of workflows. Additionally, users have to be managed and associated with workflow-participants inside the Shark-workflow-engine. Java-applications have to be developed and linked to workflow-applications in the workflow-engine, too.

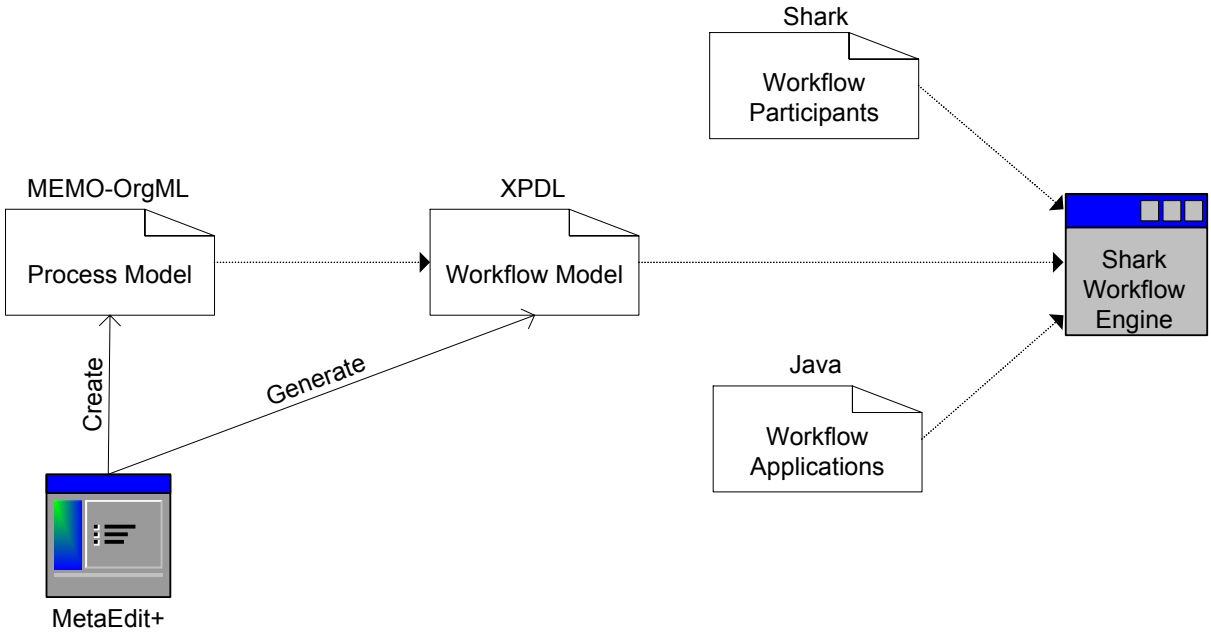

Figure 45: From process models to information systems

The approach presented in this research report is currently used and evaluated in the ECOMOD-project. ECOMOD[73] (E-COmmerce MODelling) aims at providing reference business process models for small and medium enterprises in the area of e-commerce and the generation of e-business-applications basing on these models. The reference process models will be modelled and managed using MetaEdit+. After selecting some of these generic processes for an e-commerce-application, these processes have to be configured for the special needs of a given company and mapped to XPDL using MetaEdit+. The Shark-WfE[74] will be pre-configured for the reference process models containing pre-defined actors (generic WfMS-users) and their mapping to the reference models' workflow-participants. There will be several Java-classes representing the concrete implementation of the workflow-applications of the generic business process models.

7 Acknowledgments

This research has been kindly funded by the German Research Community (DFG: Deutsche Forschungsgemeinschaft) as part of the ECOMOD project at the University of Koblenz.

[73] Be related to [FrLa04a] and [FrLa04b] for further information about the project.
[74] Workflow Engine

Abbreviations

BPM	Business Process Modelling
DTD	Document Type Definition
IT	Information technology
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
MEMO	Multi-Perspective Enterprise Modelling
Wf	Workflow
WfE	Workflow Engine
WfM	Workflow Management
WfMC	Workflow Management Coalition
WfMS	Workflow Management System
WfS	Workflow System
XML	eXtensible Markup Language
XPDL	XML Process Definition Language

References

[Baum96] Baumgarten, B.: ”Petri-Netze: Grundlagen und Anwendungen.” Heidelberg, Berlin, Oxford: Spektrum Akademischer Verlag, 2. Auflage, 1996

[BeJo01] 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

[Böhm00] Böhm, M.: “Entwicklung von Workflow-Typen: Ein Leitfaden der methodischen Anwendungsentwicklung am Beispiel ausgewählter Workflow-Aspekte.” Berlin et al.: Springer, 2000

[CKO92] Curtis, B.; Kellner, M.I.; Over, J.: “Process Modelling.” In: Communications of the ACM, September 1992/Vol. 35, No. 9, pp. 75-90

[EJLT99] 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

[Fra99] Frank, U.: ”MEMO: Visual Languages for Enterprise Modelling.” Research Report of the IS Research Institute, University of Koblenz, Nr. 18, 1999

[FrLa03] Frank, U.; Laak, Bodo van: „Anforderungen an Sprachen zur Modellierung von Geschäftsprozessen.“ Research Report of the IS Research Institute, University of Koblenz, Nr. 34, 2003

[Gad01] Gadatsch, A.: “Management von Geschäftsprozessen.” Wiesbaden: Vieweg, 2001

[GrRo99] 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

[Hans92] Hansen, H.: „Wirtschaftsinformatik I: Einführung in die elektronische Datenverarbeitung.“ 6. Auflage, Stuttgart, Jena: Gustav Fischer Verlag, 1992

[Hei88] Heinen, E.: „Produktions- und Kostentheorie**.“** In: Jacob, H. (Hrsg.): Allgemeine Betriebswirtschaftslehre, pp. 209-299, Gabler, 5. Edition, 1988.

[Herb97] Herbst, H.: ”Business Rule-Oriented Conceptual Modeling.” Physica-Verlag, 1997

[Holl95] Hollingsworth, D.: “The Workflow Reference Model.” Document Number TC00-1003, Winchester: Workflow Management Coalition, 1995

[Holl04] Hollingsworth, D.: „The Workflow Reference Model: 10 Years on.“ In: Fischer, L. (editor): „Workflow Handbook 2004.“ Lighthouse Point (FL/USA): Future Strategies, 2004

[Jung01] Junginger, S.: „Workflowbasierte Umsetzung von Geschäftsprozessen.“ Dissertation, Universität Wien, 2001

[Jung03] Jung, J.: “Some Reflections on the Basic Conceptualisation of a Resource Modelling Language for Business Process Modelling: Concepts, Requirements and Open research Questions.” Research Report of the IS Research Institute, University of Koblenz, Nr. 35, 2003

[JuKi04] Jung, J.; Kirchner, L.: “A Framework for Modelling E-Business-Resources.” Research Report of the IS Research Institute, University of Koblenz, Nr. 44, 2004

[KoPl00] Koubarakis, M.; Plexousakis, D.: ”A Formal Model for Business Process Modelling and Design.” In: Wangler, B.; Bergman, L. (Eds.): CAiSE 2000, LNCS 1789, Springer, 2000, pp. 142-156

[Mato03] Matousek, P. :”Verification of Business Process Models.” PhD Thesis, Technical University of Ostrava, 2003

[Nori02] Norin, R.: „Workflow Process Definition Interface: XML Process Definition Language.“ Document Number WfMC.TC-1025, Lighthouse Point (Fl): Workflow Management Coalition, 2002

[Nübe01] Nübel, H.: “The resource renting problem subject to temporal constraints**.”** In: OR Spektrum (2001) 23, pp. 359-381.

[Obe96] Oberweis, A.: ”Modellierung und Ausf¨uhrung vonWorkflows mit Petri-Netzen.” Stuttgart, Leipzig: Teubner, 1996

[Öst95] Österle, H.: ”Business Engineering: Prozess- und Systementwicklung.” Springer, 1995

[PSO99] 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

[Sche99] Scheer, A.-W.: ”ARIS - Business Process Modeling.” 2nd edition, Springer, 1999

[SS01] Schiemenz, B.; Schönert, O.: „Entscheidung und Produktion.“ Lehr und Handbücher der Betriebswirtschaftslehre, München, Wien, Oldenbourg Verlag, 2001.

[Sta97] Stark, H.: “Understanding Workflow.” In: Lawrence, P.: “Workflow Handbook 1997.” Chichester et al.: Wiley, 1997, pp. 5-25

[SuOs97] 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

[WaWe93] Wand, Y.; Weber, R.: ”On the Ontological expressiveness of Information Systems Analysis and Design Grammar.” In: Journal of Information Systems, Vol. 3, Nr. 2, 1993, pp. 217-237

[Web97] Weber, R.: ”Ontological Foundations of Information Systems.” Coopers and Lybrand Accounting Methodology, Monograph No. 4, 1997

[WfMC98] o.V.: “Workflow Management Application Interface (Interface 2&3) Specification.” Document Number WFMC-TC-1009, Workflow Management Coalition, 1998

[WfMC99] o.V.: “Terminology & Glossary.” Document Number WFMC-TC-1011, Workflow Management Coalition, 1999

Previous Reports

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 Wirtschaftsinformatik, **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.; Frank, U.: „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 2003

Jung, J.: “Some Reflections on the Basic Conceptualisation of a Resource Modelling Language for Business Process Modelling - Concepts, Requirements and Open Research Questions.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 35**, Koblenz 2002

Troitzsch, K. G.; Kaiser, S.; Mayer, A.; Meyer, U.: „E-Government. Forschungsfragen, State-of-the-Art und Perspektiven.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 37**, Koblenz 2003

Lange, C.: „Analyse und Entwicklung von Strategien für KMU im Electronic Commerce.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 38**, Koblenz 2003

Lange, C.: „Developing Strategies for Electronic Commerce in Small and Medium Sized Companies - Guidelines for Managers.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 39**, Koblenz 2003

Lange, C.; Frank, U.: „Ein Bezugsrahmen zur Verfeinerung und Umsetzung von Unternehmensstrategien im Electronic Commerce.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 40**, Koblenz 2004

Frank, U.; Lange, C.: „A Framework to Support the Analysis of Strategic Options for Electronic Commerce.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 41**, Koblenz 2004

Frank, U.; Lange, C.: „Corporate Strategies for Electronic Commerce – Stepwise Refinement and Mapping to Generic Business Models.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 42**, Koblenz 2004. (forthcoming)

Frank, U.; Jung, J; Kirchner, L.: “A Library of Generic Business Process Models for Electronic Commerce.” Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 43**, Koblenz 2004. (forthcoming)

Jung, J.; Kirchner, L.: „A Framework for Modelling E-Business Resources.” Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 44**, Koblenz 2004.

Frank, U.; Lange, C.: „Einführende Lehrbücher für ‚Information Systems’ aus dem Blickwinkel der Wirtschaftsinformatik – Vorbild oder Bedrohung?“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 46**, Koblenz 2004.

Jung, J.: „Mapping of Business Process Models to Workflow Schemata – An Example using MEMO-OrgML and XPDL.“ Arbeitsberichte des Instituts für Wirtschaftsinformatik, **Nr. 47**, Koblenz 2004.