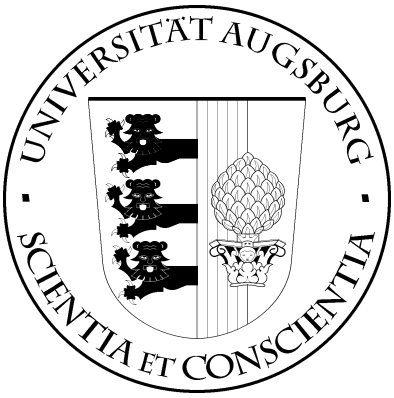
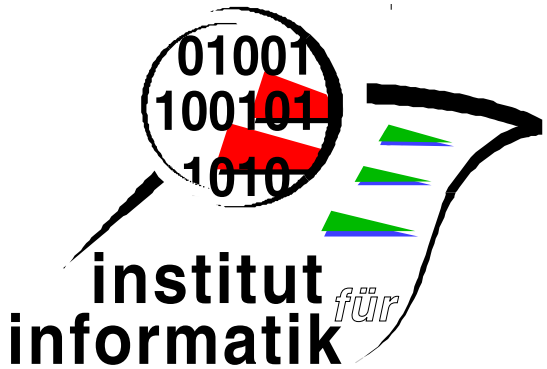
Universität Augsburg



Patterns for Semantic Business Process Modeling

Christian Seitz

Report 2008-07 Mai 2008



Institut für Informatik

D-86135 Augsburg

Business Process Management has been one of the main topics in commercial information technology for many years and is becoming even more important now. The graphical modeling of business processes and their processing into software products requires so much human labor, that the production cycles can not comply with the fast changing demands of today's global markets. To improve the degree of automatic processing in Business Process Management, techniques from the Semantic Web like ontologies and reasoners have been transfered to the business process world. A new research area – Semantic Business Process Management (SBPM) – with semantically enriched business process models as one of its main elements has been established.

Because Semantic Business Process Management is such a new research area, there are plenty of different ideas on how semantic technologies could improve Business Process Management. But they all have in common, that semantically enriched business process models are the key to all the future benefits. Nonetheless there is no common understanding on what additional information a business process model has to hold for all the different SBPM applications.

After an introduction to important concepts and technologies, this report provides patterns for Semantic Business Process Modeling. Based on the different SBPM applications, the new requirements for process models are determined and lead to a final pattern set, that describes what additional semantically enhanced information is necessary. Suggestions for the implementation of the patterns in modeling notations are included, too. Additionally, two actual modeling notations, the Business Process Modeling Notation (BPMN) and AgilPro are reviewed for their compliance with the patterns for Semantic Business Process Modeling.

Contents

1 Introduction

1.1 Motivation

Business Process Management (BPM) – the description, refinement, execution and analysis of business processes – has been one of the main topics in commercial information technology for many years and is becoming even more important now. Todays' globalized markets force organizations to apply highly flexible processes in order to adapt to rapidly changing situations. Business Process Management has to cope with this ever increasing demands, but the actual BPM technology has reached some limits. The graphical modeling of business processes and their processing into software products requires so much human labor, that the production cycles can not comply with the temporal demands. That is mainly due to the reporting character of actual business process models. Obviously there is a need to reduce the amount of human labor, which requires machine-readable business processes with enough information for automatic processing.

Here another research area of information science, that gains in importance more and more, comes into play. Technologies of the Semantic Web, that provide machine-interpretable semantic information about (web) content, seem to be a possibility to pave the way for automatic information reasoning and processing. The hopes in this technologies are very high, as can be seen e.g. by the THESEUS program[1]. This research program is initiated by the German Federal Ministry of Economy and Technology and has the goal to utilize the available knowledge on the internet much better as it is done today. The focus of this program is on semantic technologies.

The application of Semantic Web technologies in the BPM context has led to a rather new research area called Semantic Business Process Management (SBPM). The idea is to use new methods to break out of the dead end Business Process Management is in nowadays and offer new possibilities. The research in SBPM is still mainly on a very basic and theoretical level, but the great promises this approach seems to provide, should be worth all the effort.

[1] `http://theseus-programm.de/front`, as at 2008-03-20

1.2 Problem Definition and Approach

Because Semantic Business Process Management is such a new research area, there are plenty of different ideas on how semantic technologies could improve Business Process Management. But they all have in common, that semantically enriched business process models are the key to all the future benefits. Nevertheless, most researchers concentrate on specials aspects of SBPM. Either the basic technologies like ontologies, reasoning and annotation techniques are of interest, or a special application area like B2B scenarios is examined. But there is no overall definition what additional information a business process model has to hold for all the different SBPM applications.

So the task of this thesis is to link the modeling requirements for all the application ideas, that were developed so far, to a set of Semantic Business Process Modeling patterns. The patterns emphasize the additions of a SBPM business process model in contrast to a "normal" one. Based on the different SBPM applications, the new requirements for the process model are determined and lead to the final pattern set. The result is an overview, that provides some sort of a checklist for future SBPM research and particularly first applications. For example, the developers of a prototypical SBPM toolsuite could be geared to the patterns and already consider certain aspects, that are not their first priority, but essential for the support of SBPM as a whole. The layout of the patterns is intentionally similar to existing workflow patterns, as they are meant to complement the existing business process modeling knowledge.

But the patterns are of no use, if there is no modeling notation that supports the new requirements. Therefore two actual modeling notations, the Business Process Modeling Notation (BPMN) and AgilPro (see sections 2.5 and 2.6) have been reviewed in this thesis for their compliance with the patterns for Semantic Business Process Modeling. For every notation and every pattern the degree of support has been determined and a final summary and comparison shows promising results.

1.3 Outline

The next chapter begins with an introduction to all the necessary technologies and concepts, that will become important throughout this thesis. First the two basics of SBPM, Business Process Management and Semantic Web Services, are introduced in sections 2.1 and 2.2. Web Services and Semantic Web as the components of SWSs are presented as well. After that, section 2.3 describes the vision and concept of Semantic Business Process Management together with all the new possible applications. As SBPM is a rather new research area, there are many active research projects and groups. The most important ones are introduced in

2.4. As BPMN and AgilPro are used for the pattern-based analysis, a short overview on both modeling notations is given in 2.5 and 2.6 respectively.

Chapter 3 contains the patterns for Semantic Business Process Modeling. They are listed in four different groups, depending on their role: Base Patterns (3.1), Functional Patterns (3.2), Organizational Patterns (3.3) and Conditional Patterns (3.4). The description of each pattern contains an example, the motivation for the pattern, suggestions for an implementation approach and an evaluation criteria for the analysis.

The pattern-based analysis of BPMN and AgilPro is done in chapter 4. Every pattern for Semantic Business Process Modeling is examined on its own. Different possible implementation approaches of a pattern are considered and the analysis results presented in table form. Section 4.11 summarizes the results and gives a final overview of the results.

Finally, chapter 5 recapitulates the results of this thesis and gives prospects on the future challenges and options of Semantic Business Process Management and Modeling.

2 Basic Technologies and Concepts

In the course of this paper it will be necessary to have a basic understanding of certain technologies and concepts. Therefore the vision of Semantic Business Process Management (SBPM) is introduced after an overview of Business Process Management (BPM) and Semantic Web Services as the basic elements of SBPM. This is all necessary to get a better understanding of possible applications of SBPM examined in section 2.3.2 and the underlying technologies. In order to give an impression of the current SBPM research situation, the next section deals with several major research projects, their main focus, intentions and present results. Finally the two business process modeling languages BPMN and AgilPro are presented, which are subject of the pattern-based analysis in chapter 4.

2.1 Business Process Management

For the last few years Business Process Management has become more and more important for organizations in order to stay competitive on a global market, which requires to react as fast and cheap as possible to constantly changing conditions, business partners and customer needs.

> **Definition 2.1** Business Process Management
> *"Business Process Management is concerned with the modeling, automation, administration, monitoring, measuring, evaluation and optimization of business processes. It combines several existing technologies, such as workflow solutions, EAI (Enterprise Application Integration) products and Business Intelligence software"*
> [QW04, p. 28]

All the tasks mentioned in this definition should be accessible from a business perspective without the knowledge of technical details. Business experts should be able to do their work independently from assistance of the IT department. BPM can be understood as a further development of simple Workflow Management, which deals with the specification and software aided execution of business processes.

> **Definition 2.2** Business Process
> *"A business process is the complete and dynamically coordinated set of collaborative*

and transactional activities that deliver value to customers." [SF03, p. 47]

[vdAtHW03, p. 2 et seq.] identified three trends in information systems, which lead to the increasing relevance of Business Process Management. The first trend is, that the challenge for today's software developers is no longer to just produce the code for a single task, but to assemble complex systems from many different modules. This corresponds to the idea of business processes as a sequence of activities. Each activity can be represented by a single software module and the whole business process can be assembled from these parts. A second trend is, that there is a change from data-driven to process-driven approaches in system engineering, mainly due to recent developments in management theory where it is all about processes. The third and last trend the authors mention, is the continuous change of software systems and the reuse of parts of existing applications in new ones. This shift to a more dynamic software development corresponds to the very dynamic situations organizations are confronted with and the resulting agile business processes.

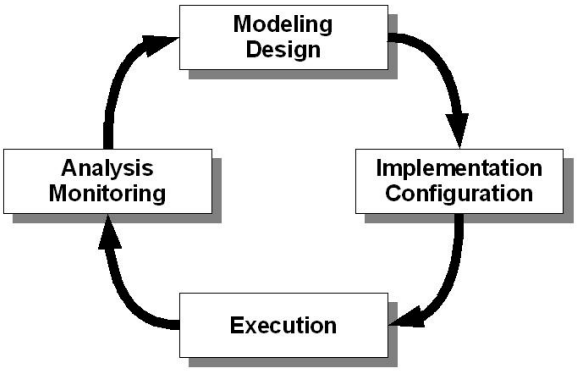


Figure 2.1: Business Process Management Lifecycle

There are several similar concepts of a BPM lifecycle with only slight differences (cf. [QW04, p. 28 et. seq] or [vdAtHW03, p. 5]). In a first phase, the design and modeling of the business process takes place. This includes both the modeling of the business process with a graphical tool/language by business experts and a formal specification of the process in a process language for the execution of the workflow. Today many workflow solutions are available, but the step from a graphical representation to a formal specification of a workflow, which needs technical details for the later execution, is still critical. The next phase can either be seen as simply the configuration of a workflow management system with a following execution phase [vdAtHW03], or as the implementation and execution of a business process condensed in one phase [QW04]. Each alternative is arguable. The latter perception seems to suit the approach of this thesis better, because by considering both implementation and execution in a single phase, the fact of permanently changing processes is expressed much

better than by the more static approach of first configuring a system and using it afterward. Instead of gluing together complete autonomous applications the promising trend now is to use Web Services to execute a process. Nevertheless, the SBPM life cycles presented so far mostly correspond to a version like the one shown in figure 2.1. The last phase is always an analysis of the executed processes to identify problems, check the performance or the compliance with specific goals and compare the real data with the planning in order to improve and redesign the process if necessary. The analysis can be divided into two parts: Business Process Analysis, which includes "for example simulation and diagnosis, verification and performance analysis" [vdAtHW03, p. 8] and Business Activity Monitoring (cf. [vdAtHW03, p. 5]), which is very similar to the existing technology in Business Intelligence (cf. [QW04, p. 34 et seq.], where data, gathered during the execution of processes, is stored and analyzed e.g. with Data Warehouse applications.

In nearly every field exists a bunch of different standards, what makes it hard to get a complete survey of the current BPM market situation. A single standardization for all BPM aspects still has to be done. Because of the current hype about the term Business Process Management, a surge of vendors claiming their products to be BPM-products floods the markets. For an overview of available products and a first guess on what they provide, the "BPM Vendor Directory Listing"[1] of the Object Management Group (OMG) might be a good starting point.

2.2 Semantic Web Service Technologies

Semantic Web Services use technologies of the Semantic Web to raise the potential of Web Services. This section describes each technology with the most common standards / languages short on its own previous to the examination of Semantic Web Service (SWS) technologies and concepts.

Definition 2.3 Semantic Web Service
"The Semantic Web Service vision is to describe Web services' capabilities and content in an unambiguous, computer-interpretable language and improve the quality and robustness of existing tasks, such as Web service discovery and invocation. Semantic Web services will also enable a broad range of new automation tasks (...) including automated composition, interoperation, execution monitoring, and recovery." [MM03, p. 90 et seq.]

[1] `http://bpm-directory.omg.org/vendor/list.htm`, as at 2007-10-25

2.2.1 Web Service

The idea of Web Services is to enable access to applications and their functions over the web independently from the underlying technology and implementation language, contrary to other invocation mechanisms like e.g. Remote Procedure Calls. To achieve this independence, Web Services use XML as base technology and the World Wide Web Consortium (W3C) standardized several XML-based Web Service technologies, namely SOAP, the Web Services Description Language (WSDL) and XML schemas to define data types. Another standard is UDDI from OASIS[1].

SOAP[2] provides a format/framework to exchange XML-messages in distributed environments independent from operating systems, programming languages or runtime components [WF07, p. 31]. There is no restriction on the used transport mechanism, nevertheless HTTP is the most commonly used. SOAP serves as message format for all necessary communication during discovery and invocation of a Web Service.

WSDL[3] is a XML-based Interface Definition Language (IDL) for Web Services that *"provide(s) an exact and machine readable definition of service interfaces"* [WF07, p. 36]. The service descriptions are necessary for a client to call the Web Service and its functions in the correct way. WSDL covers only the syntactical part of the interfaces, there is no representation of semantical information in a machine-readable form.

UDDI[4] *"is a platform independent electronic technology for general purpose business registries"* [WF07, p. 40]. Users can search for specific Web Services in a UDDI Web Service registry by using White Pages (basic information), Yellow Pages (industrial categorization) and Green Pages (technical information about the services). If a service suiting the customer's needs has been found, the WSDL description with all information on where and how to invoke the service are delivered. Service providers can publish their services in such a repository and make it available to a greater public. Again there is no formalism for a semantic representation of information that would allow automatic processing.

The line-up presented in figure 2.2 is very common throughout many Web Service application scenarios. Providers of Web Services register their services at a Service Registry, which includes provision of the WSDL files. Service requesters are now able to search services at the registry, receive the description (WSDL file) of the desired Web Service and start usage of the service. Of course the search via a service registry agency can be omitted, if the Web Service and its provider are already familiar to the requester.

[1] Organization for the Advancement of Structured Information Standards, `http://www.oasis-open.org`, as at 2007-10-29

[2] current version: SOAP 1.2 (`http://www.w3.org/TR/soap12/`, as at 2007-10-29)

[3] current version: WSDL 2.0 (`http://www.w3.org/TR/wsdl20/`, as at 2007-10-29)

[4] current version: UDDI 3.0.2 (`http://www.oasis-open.org/committees/uddi-spec/doc/spec/v3/uddi-v3.0.2-20041019.htm`, as at 2007-10-29)

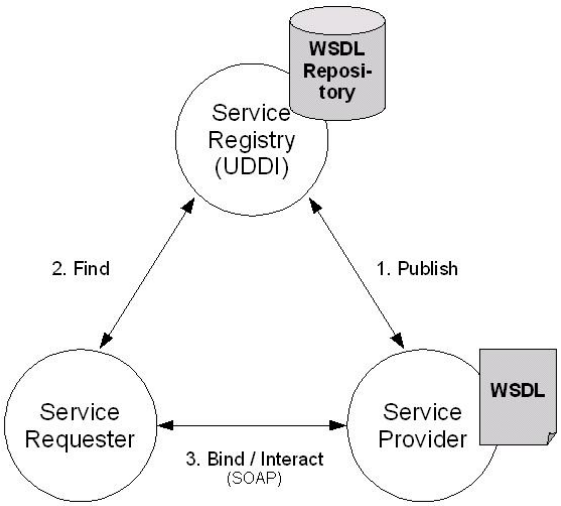


Figure 2.2: Web Service Role Model

Speaking of the combination and cooperation of several different Web Services, the terms 'choreography' and 'orchestration' have great importance. While choreography defines a more collaborative perspective where each participant describes its own role in the interaction with other web services, orchestration captures the interaction of many services that together execute something like a business process. In orchestration one party has the control over the whole process, each single Web Service is not necessarily aware of the overall picture, meaning e.g. which other services can and will invoke it (cf. [Pre07, p. 164 et seq.]).

2.2.2 Semantic Web

The vision of the Semantic Web by Tim Berners-Lee to bring machine-accessible semantics to web content is described pretty well in an article for the Scientific American: *"The Semantic Web will bring structure to the meaningful content of Web pages, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users"* [BLHL01]. The idea is to get away from today's HTML pages, where only a human reader with background knowledge can make expedient conclusions and use the information for other sophisticated tasks, toward a web where e.g. software agents can use the structured content of websites automatically.

To achieve this goal techniques from the research field of knowledge representation, particularly ontologies, are used.

Definition 2.4 Ontology
"An ontology is an explicit specification of a conceptualization.(...) In such an ontology, definitions associate the names of entities in the universe of discourse (e.g.

> *classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these terms."* [Gru93]

Ontologies usually consist of concepts, relations and instances. Concepts represent categories of the domain, relations their semantical connections and instances the concrete objects [GHA07, p. 70 et seq.].

Due to the work of the W3C, two main languages are of significant importance: the Resource Description Framework (RDF)[1] and the Web Ontology Language (OWL)[2]. Also the Web Service Modeling Language (WSML)[3] as an ontology language for the description of Web Services becomes more and more important. All those languages can be serialized with XML.

Statements are declared via subject-predicate-object constructions in RDF, Unique Resource Identifiers (URIs) are used to name the entities. The extension RDF Schema (RDFS) introduces the concepts of class (for subject/object position) and property (predicate), which allows the modeling of hierarchical relations. This enables the distinction between concepts of the domain and specific instances of such concepts, too (cf. [GHA07, p. 82 et sqq.]).

Contrary to RDF, OWL is based on Description Logics and therefore allows reasoning on information by using e.g. deduction mechanisms. The typical ontology elements – concepts, relations and instances as mentioned above – are represented by classes, properties and individuals in OWL. The approach of using class-hierarchies is very similar to object oriented languages. Complex classes are built from simpler ones using constructors, sufficient and/or necessary conditions can be specified (cf. [GHA07, p. 87 et sqq.] and [QW04, p. 6]). There is a whole family of OWL languages that differ in expressiveness and complexity and for each application the most suitable should be chosen.

Finally, WSML is the language for the Web Service Modeling Ontology (WSMO), which will be discussed later on. As a language for the description of Web Services, WSML has not only the typical ontological constructs, but also specific ones like 'goal', 'web service' or 'choreography'. But of course it has all the capabilities to describe e.g. domain ontologies just like the other languages discussed so far. WSML consists of a conceptual part where the typical ontological modeling with concepts, relations and instances is done, and a part where complex information is stated via axioms built with logical formulas. The usage of data types is supported as well (cf. [GHA07, p. 98 et sqq.]). Such as OWL, WSML has several language variants with different expressiveness.

[1] `http://www.w3.org/RDF`, as at 2007-02-11
[2] `http://www.w3.org/TR/owl-features/`, as at 2007-11-02
[3] `http://www.wsmo.org/wsml/wsml-syntax/`, as at 2007-11-02

Despite the large variety of ontology languages, the problem of ontology modeling is mostly not to find the right language and master it, but to actually find the right concepts and representation of a domain, that is well accepted by every user in such a domain.

2.2.3 Semantic Web Service

The absence of machine-interpretable semantic descriptions throughout all technologies of Web Services is the reason why the full potential of Web Services cannot be exploited. Semantical aspects are only described via unstandardized free text, so the automatic discovery, composition, orchestration and choreography of Web Services is very difficult, if not impossible at all. By the use of Semantic Web technologies this drawback is erased in Semantic Web Services.

The foundation of SWSs is the semantic annotation of Web Service descriptions. There are several different approaches, the three most promising ones at the moment are described shortly in the following.

OWL-S[1] provides an upper ontology for services with the four main elements *Service*, *Service Profile*, *Service Model* and *Service Grounding* [LLP+07, p. 197 et sqq.]. Every Web Service is declared by an instance of the concept *Service*, which links the other concepts. The *Service Profile* describes information like the name of the service or its description, non-functional properties and the functionality by capturing inputs, outputs, preconditions and effects/results. The *Service Model* provides a description of how the service works and how to interact with it by modeling the service as a process. Finally, the *Service Grounding* concept contains information on how to access the service. OWL is used as language for OWL-S. But some other languages are allowed to describe preconditions and effects, because the expressiveness of OWL does not cover all needs. Examples for those languages are the Semantic Web Rule Language (SWRL), the Knowledge Interchange Format (KIF) and Declarative RDF System (DRS).

The Web Service Modeling Ontology (WSMO)[2] conceptual model was developed with the goal to enable the automatic execution of all tasks relevant for the dealing with Semantic Web Services (cf. [LLP+07, p. 182 et sqq.]). It is a metamodel for Semantic Web Services. The four core elements of WSMO are *Goals* (describe what the user wants a Web Service to fulfill), *Services* (represent the Web Services with their capabilities, interfaces, non-functional properties and information for choreography and orchestration), *Ontologies* (the terminology used by all other elements) and *Mediators* (description of elements that deal with interoperability problems concerning e.g. data, protocol or ontologies). All WSML dialects can be used to describe the elements.

[1] `http://www.w3.org/Submission/OWL-S/`, as at 2007-11-12
[2] `http://www.w3.org/Submission/WSMO/`, as at 2007-11-12

Another approach for the description of Semantic Web Services is SAWSDL[1], which is based on the previous work on WSDL-S[2]. Instead of providing a full framework like OWL-S or WSMO it just adds annotation tags with semantics to the normal WSDL descriptions of Web Services. There are annotation tags to reference a WSDL component to a concept in an ontology and to define the mapping between XML schema types and ontologies (important for mediation aspects). Obviously SAWSDL is a more lightweight approach than the others, but has to cope with the absence of a commitment to a specific ontology language. So far none of these approaches has prevailed, but it seems to be essential for the further development of Semantic Web Services to define a single accepted standard.

With the semantic description of all aspects of Web Services, the identification of relevant services during discovery becomes possible. The idea is, that software agents search for Web Services, that fulfill the users demands, and discover potential service providers. Therefore semantic capability descriptions from both requester and provider have to be compared. Domain ontologies play an important role in the description of the capabilities. The way how the matching is performed often depends on the way how capability descriptions were modeled. Concepts from the fields of knowledge representation and automated reasoning are used. A model independent approach is to use Description Logic (DL) inferencing. There are specific efforts for the matchmaking of service descriptions and the integration into existing discovery technologies like UDDI for WSDL-S, OWL-S and WSMO . For a detailed overview on discovery of Semantic Web Services see e.g. [Gri07].

Another benefit of SWS is the automatic composition (combination and coordination) of services, which means to compute a whole orchestration. The most convenient way would be to compose the services in a goal oriented manner. The user designs his request, a composition goal, out of several atomic goals. This step provides a set of choreographies which are considered in the next step, when atomic goals are replaced by SWSs [HK07, p. 246 et. seq.]. To describe the choreographies and orchestrations some sort of workflow language is needed, which is appropriate for reasoning of workflows. Automated composition is still a field of intense research and many techniques are considered to manage this problem, e.g. first-order logic approaches, type matching, problem-solving methods, AI planning and many more (cf. [HK07, p. 258 et sqq.]).

One obvious problem of Semantic Web Services is the heterogeneity at different levels, namely the data, protocol, ontology and process level. To allow a trouble-free usage of Web Services mediators are applied, which are *"defined as entities for establishing interoperability of resources that are not compatible a priori by resolving mismatches between them at runtime"* [CLB07, p. 288]. Data mediation deals with the different syntactic format of input and output data during message exchanges. Today the transformation is often implemented specifically

[1] `http://www.w3.org/TR/sawsdl/`, as at 2007-11-26
[2] `http://www.w3.org/Submission/WSDL-S/`, as at 2007-12-11

for each requester/provider combination, because semantic information about data and necessary ontology mediation is not considered. Ontology mediation transforms the different used ontologies, as it is possible, that requester and provider use different semantic models of a domain. There are mainly two general strategies in ontology mediation. Ontology alignment creates correspondences between different ontologies, while ontology merging creates a new ontology from the old ones. Several algorithms and tools for each approach have been developed, for a short summarization see e.g. [CLB07, p. 300]. Protocol mediation is concerned with the interoperability between different interaction protocols [CLB07, p. 304] and process mediation with the different business process concepts the Web Services represent.

Many concepts and problems of Semantic Web Services bear a resemblance to the ones of Business Process Management and so the ideas of Semantic Business Process Management come not that surprisingly.

2.3 Semantic Business Process Management

The general vision and concept of Semantic Business Process Management is explained first in order to give a common understanding of this term, before all the new possible applications of this technology are summarized and explained in greater detail.

2.3.1 Vision and Concept

The vision of SBPM can be traced back to [HLD+05]. The authors give reasons for the advancement of current BPM approaches and introduce their idea of combining technologies from Semantic Web/Semantic Web Services with BPM.

> **Definition 2.5** Semantic Business Process Management
> *"Semantic Business Process Management is a novel approach of increasing the level of automation of BPM by representing the various spheres of an enterprise using ontology languages and Semantic Web Services frameworks."* [HR07, p. 424]

Permanently evolving business connections and dynamic markets are amongst others key challenges for today's organizations to compete successfully. Thus it becomes crucial for a company to execute their processes efficiently, set up new processes at low costs and reduce the delay based on the switch from one process to another. The fundamental problem despite the widespread usage of BPM environments that prevents organizations from mastering these challenges, is the gap between the business expert's perspective, which determines the processes, and the real implementation in the IT landscape of the organization. Due to the lack of machine-accessible semantics, that describe both the business and technical aspect,

the degree of mechanization is very limited (no machine reasoning possible) and a complete unified view on the process space of an organization is missing (cf. [HLD+05, p. 536 et seq.]. This leads to a dissatisfying situation where unnecessary human labor is required, because all the information about an organization and its process space, that is already stored in the computer systems, can not be used efficiently (cf. [HR07, p. 426 et seq.]).

The solution [HLD+05] propose is to use SWS technology to access the process space in the two fundamental forms the authors of the paper mention: querying and manipulating the process space. Querying the process space means, that decision makers need access to all relevant information necessary to come to a decision. In current BPM environments analysts have to gather the information in cumbersome handwork, because there is no machine-readable representation of all relevant aspects in the process space of an organization as well as of the queries themselves on a semantical level. Manipulating the process space includes the fields of creation, modification, substitution and execution of processes. Of course these first thoughts have been just a starting point for many others to develop new ideas and proposals how technologies from Semantic Web/Semantic Web Services can be used to raise BPM to the next level. A detailed overview of possible applications of Semantic Business Process Management, which are also the foundation for the developed patterns in this thesis, will be given in section 2.3.2.

Semantic Web technologies useful for all those possible tasks are mainly ontologies, but also repositories, reasoners, mediation components and query languages. The goal of SBPM is to use the combination of BPM and Semantic Web Services to boost the degree of automation in modeling, composition and orchestration of processes, automatically generate implicit knowledge to get a better overview of the complete process space and to enable more intelligent queries, and finally execute the processes via Semantic Web Services. The SBPM environment should also be able to make its decisions at runtime, based on specified rules and goals developed by business experts and limitations due to the existent IT infrastructure. An introduction to Semantic Web Services has been given in the preceding section 2.2.

To get an idea of the usage of SBPM, a process life cycle in a Semantic Business Process Management environment is introduced in [HR07, p. 429 et sqq.]. Process models are the input to such an environment. The sources of such models are the same as for common BPM environments: modeling tools, process libraries, reverse business engineering or process mining tools. The first step in the lifecycle is to ontologically lift the input. In most cases this will require human labor, because the correct semantical annotation of the input is the key to everything else. A SBPM process formalism is used next to store the original workflows and add any kind of relation to elements in the organization (e.g. resources). So a process repository emerges that can be used by modeling tools and analytic tools. Once the process

formalism exists, either code for existing execution engines (e.g. using BPEL[1] or EPCs[2]) can be exported or a SBPM execution engine can execute the process models directly. If a process is specified only declaratively, the SBPM execution engine uses a reasoner to create a valid control flow or another tool can help the user to manually create a valid control flow. The SUPER research project produced an adapted version of that Semantic Business Process Management life cycle [FSM+07] - for an introduction to the SUPER project see section 2.4.2. Also a more abstract life cycle, very similar to the BPM life cycle, has been introduced with modeling, configuration, execution and analysis phases, but with an additional ontological foundation layer responsible for the ontological lifting and a strategic Semantic Business Process Management layer that deals with goals and organizational conditions of the entire enterprise.

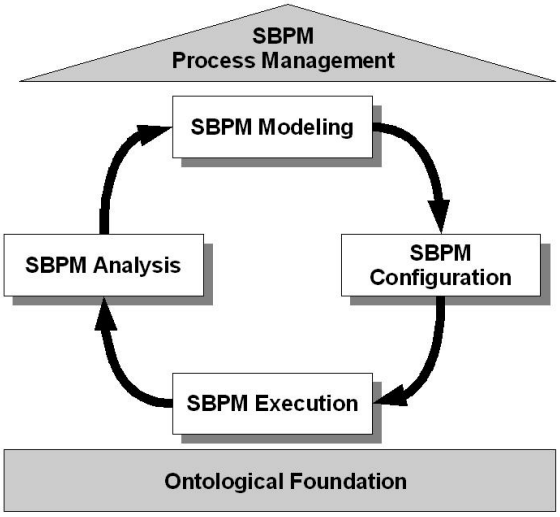


Figure 2.3: SBPM Life Cycle of the SUPER project

2.3.2 Applications

Since the presentation of the SBPM vision many other people have given some thought to possible applications and their benefits. This section gives an overview of those ideas, arranged according to the corresponding lifecycle phase. While some ideas are rather dreams of the future, others are the topic of intense research and applications are not that far away.

In order to make it more descriptive, the applications of SBPM are presented in the context of a use case scenario in a second passage for each application. An industrial machinery

[1] Business Process Execution Language: a xml-based business process modeling language used for the orchestration of the web services that execute single tasks

[2] Event-driven Process Chain: modeling method to describe business process workflows developed by Prof. Wilhelm-August Scheer

manufacturer IMM serves as sample company in this use case. All the details necessary for the description of a specific SBPM application will be presented in due course.

Modeling

An area where the greatest enhancements are provided by SBPM is the modeling of business processes itself. The semantic annotation of process model elements and the ontological capturing of a company's organization and IT infrastructure is not only useful for the later phases, but together with a business process repository the manual work of modelers can be reduced significantly.

- **Reuse of process fragments:** All possibly reusable fragments of a business process are stored in a process repository together with their semantic descriptions (e.g. domain, funcionality, etc.). If a business analyst creates a new business process later on, he can search the repository for existing process fragments that suit his needs and integrate them into his process (cf. [WMF+07]). By doing so, unnecessary double work as well as the error-proneness is reduced to a minimum.

  A conceivable scenario for our example company would be, that the business analyst has to model a process for the production and sale of a new machinery. The issuing of an invoice is the same for all products of the company, so the analyst can specify a search for a process fragment that covers exactly this part. The matching process fragment is integrated automatically into his process.

- **Replacement of process fragments:** It often happens, that small parts of a business process have to be replaced with an updated version that is more efficient, uses new technologies and so on. By using semantically enriched business processes, it can be automatically verified if the replacement of an old process fragment with a new one is possible (cf. [HLD+05]). Critical problems (e.g. the existing hardware cannot be used for the software introduced with the new process, a certain output is not produced anymore or the goal of the process is not fulfilled completely) can now be detected in the modeling phase and great costs for the company are avoided by the early detection.

  The old business process covering the design of new machinery of IMM has to be replaced by a new one to make it more efficient. By doing so, a new expensive CAD tool is introduced as well and the business analysts changes the process correspondingly. The SBPM modeling tool of the company detects that the existing PCs in the design department cannot run the new CAD tool (the complete infrastructure of the company is modeled and semantically enriched). This would have caused enormous costs for the company, either by buying new equipment or another tool that runs on the old

hardware. With the warning from the SBPM tool the business analyst is able to change to an applicable CAD tool.

- **Auto-Completion:** Modeling business processes is often a time-consuming task and assistance from the modeling tool is therefore desirable. A great help would be an auto-completion mechanism for unfinished process models (cf. e.g. [BKKO06]). By comparing the partial model with fragments from the process repository, the engine can make suggestions on how to complete it and do it after selection by the modeler. The similarity of process fragments has to be measured by considering the syntactical, semantical and structural aspects. Also the user behavior is of great importance (e.g.: if always the same completion option is chosen, put it at the top of the list). Before the completion of the process, the resulting process has to be checked if it fulfills certain properties, e.g. deadlock freeness. Such a auto-completion mechanism is highly applicable for process parts that appear very often throughout many different processes.

  In nearly all manufacturing processes of our IMM company a component has to be fetched from one of the storage rooms. Because there are no fixed storing positions, the exact location has to be looked up. If the business analyst now models a new manufacturing process and starts a sequence where a component from the storage is required, the modeling tool detects this context and the auto-completion mechanism suggests to insert the lookup part. The modeler approves it with a single click.

- **Automatic process generation:** The next step that goes much further than an auto-completion mechanism would be the (semi)automatic planning and generation of business processes (see e.g. the SEMPRO project presented in section 2.4.6). Given a goal that has to be reached by executing the process, a planning algorithm tries to build a correct business process from elements available in a repository. Of course semantic annotation is necessary for the algorithm to find a solution.

  The business analyst of IMM has to design a process that covers the holiday planning of a new department. He specifies the goal and requirements of this new process and the modeling environment creates a suggestion for the process by using parts from stored holiday planning processes from other departments. The designer can now make minor adjustments to the process specific for this department and has the entire business process ready in no time.

- **Verification of guidelines and regulations:** Semantically enriched process models open new opportunities for automatic model checking. It is conceivable, that business process models are checked for their compliance with quality standards, modeling guidelines and internal or legal regulations (cf. [SBO07] and [NS07]).

  Our example company has to comply with accounting regulations from the government and stock exchanges. Therefore the business processes dealing with the accounting

have to contain several specific tasks. After the creation of such a business process, the modeling environment can check the compliance with the regulations automatically and informs the modeler which tasks are missing in his process. It also prohibits the execution of the process as long as the regulations are not hold.

- **Modeling of B2B scenarios:** In B2B scenarios public processes from partners have to be integrated into the business processes of an organization by building a collaborative business process. Depending on the process of the partner, the collaborative process has to behave in a way, that the two processes can interact smoothly, e.g. messages have to be sent in a specific order. This time-consuming modeling task has to be done for every new partner or process change of a partner because of different terminology/process structure etc. A quick and flexible exchange of business partners is not possible in current BPM solutions. With semantically enriched process descriptions, an automatic generation of message mappings as well as the automatic integration of partner process steps under consideration of the organizations requirements, such as the order of the own tasks in a process (via formal verification), is possible (cf. [DLN06] and [NS06]) and the easy exchange of business partners is facilitated.

  The example company IMM has a single partner for the manufacturing of parts of their products. So far there was a long-term contract with the business partner, but IMM now wants to choose from different partners flexibly, dependent on costs and other conditions. The business process modeling environment supports this by taking the public process of the partner companies as input and creating the collaborative process automatically. Messages are sent in the right order and time to each new partner and only those partners are allowed that do not conflict with the internal process order of IMM.

Configuration

Because Semantic Business Process Management uses SWS technology it is just logical that Semantic Web Services are supposed to be used whenever possible to carry out the business processes. The search for matching services can be accomplished automatically. The goals, inputs, outputs, pre- and postconditions of tasks defined in the process model are compared with the semantic descriptions of services to find the right one (or a composition of services) to do the job at hand.

The composition and orchestration of all the services of a business process requires no additional human labor if only Semantic Web Services are used or at least reduces the manual work significantly. The configuration of new and often changing business processes becomes much faster and makes it more profitable for a company to adapt its processes to fast changing conditions.

The additional non-functional information about a SWS allows the SBPM tool to select those services, that comply best with the Quality of Service (QoS) standards, organizational policy or cost regulations determined for a whole process or certain tasks in the modeling phase. This makes an organization much more flexible (e.g. easy exchange of business partners) and profit-making than an organization with rigid business processes and long time-to-market cycles for setting up or changing processes (cf. e.g. [RBV+06]).

The IMM company wants to change their shipping partner as flexible as possible, depending on the machinery to be shipped (the big size and weight of some items can be problematic for some shippers), the destination (some shippers only deliver in Europe, others worldwide), some QoS terms and conditions (like guaranteed delivery time) or simply the costs. The interaction with all possible partners happens via Semantic Web Services, so the automatic search for and selection of services (which actually means business partners), composition and orchestration of the process can be done by the SBPM tool. All the business analyst has to do, is to express the conditions for every task/process in the process models, negotiate the terms with the shippers and add the results to the service description respectively.

Execution

The automatic selection, composition and orchestration of Semantic Web Services allows to shift the dynamic decision making on which services should be used into the execution phase and to let the business process execution engine do all the work. Configuration and execution phase of the business process lifecycle merge in some way, the specifications (in BPEL4WS or similar languages) from the configuration phase are more generic and the work with concrete services is almost only done in the execution phase. The whole process lifecycle is tightened and yet more efficient and agile. The examples presented for the configuration phase of course hold here as well. But of course the execution phase still has its original function, the execution of the business process. This involves invocation of the services, data mediation, role management and so on.

Analysis

Another part of the business process lifecycle that highly profits from the application of SBPM technology is the analysis phase. In this section the possible influence of those technologies on process monitoring (observation and analysis during process execution), process mining (analysis based on event logs) and queries about the whole organizations process space (including the process models and execution specification) is presented.

The great advantage of SBPM solutions is the semantical annotation of all elements that are relevant during the process lifecycle. This enables business analysts to formulate more intelligent queries. It is easier to decide whether a process is executed in the intended way (order of process tasks, the right roles perform the right tasks etc.) or not, or to find possible improvements of the processes including the discovery of performance bottlenecks (cf. [WMF+07]).

Queries on nearly every topic become possible through the semantic annotations and even the generation of implicit knowledge, that would not be accessible otherwise. It becomes feasible to search the organization's process space for system interdependencies, connections with other companies (which ones, intensity, influence etc.), multiple different methods for actually the same task or the compliance with cost restrictions or other specifications to name just a few (cf. [HLD+05]). This constitutes a powerful support for decision making in an organization.

Only a few new possibilities of the example company IMM shall be described here. Via process mining, the analysts detected, that several not very challenging tasks are executed by a department full of well paid academics. As this is a waste of money and expertise, the process is restructured so that another department executes the simple tasks and the academics can concentrate on more complex work. By formulating several queries on the organization's process space, the business analyst also detected a great dependency on a single component supplier, which is rumored to declare bankruptcy. So the analyst decides that the pool of component suppliers has to be diversified much more. Another application example is a new directive from the executive board, that a process for the manufacturing of a machine is not allowed to cost more than a certain price. The analyst finds all possible solutions to execute the process at the desired cost, but detects that the quality of the resulting product (the unique selling point of IMM) would suffer and convinces the management to revise their plans.

2.4 Research Projects in the Field of Semantic Business Process Management

Semantic Business Process Management is a rather new research area and so it is not very astonishing, that there are a couple of research projects with participants from both industry and research institutes. This chapter is meant to give a brief overview of the major running projects, their main focus, intentions and results published so far (end of 2007), if there are any available.

2.4.1 FUSION

The FUSION[1] research project is funded by the European Commission with a duration of 30 months (ending July 2008) and a 2.78 million euro project funding. The project consortium consists of 14 partners, both from industry and research institutes, under the lead of SAP.[2]

Objectives of the project are the *"development of an innovative approach, methodology and integration mechanism for the semantic integration of a heterogenous set of business applications (...), platforms and languages within SMEs"*[FUS] (Small and Medium-sized Enterprises) or several collaborating companies, the integration of related European research activities (BPM, Web Services, Semantic Web) and the validation of the results by *"developing proof-of-concept pilots in collaborative commerce across semantically-enriched supply chains and value networks across the Enlarged Europe"*[FUS]. Expected results are an approach and methodology for semantic service-oriented business application integration, a FUSION ontology, an integration mechanism to interconnected heterogeneous systems and use cases that prove the developed concepts and tools (cf. [FUS]).

Apart from some scientific papers related to different areas of SBPM, the project has published some technical deliverables so far. A state of the art document describes the current situation of Enterprise Application Integration, semantic technologies in general and semantics in Enterprise Application Integration (EAI) [FKG+06]. The FUSION approach [ABB+07] introduces a reference framework, a global architecture as well as a functional and technical architecture in order to develop *"a generic high-level architecture for the integration of semantically-enriched service-oriented business applications"*[ABB+07, p.14] that builds upon current technologies like Web Services, SWS or SOA. The architecture consists of a design-time and an execution environment. Also specifications for the systems components for semantic service profiling [MGB07], manual and semi-automatic process design [FLA+07] and the integration mechanism (execution, runtime environment) [SPKP07] are already available. Important technologies used in this project are BPEL4WS, semantically-enriched UDDI, SAWSDL for the annotation of Web Service interfaces and OWL for the FUSION ontology.

2.4.2 SUPER - Semantics Utilised for Process Management within and between Enterprises

The research project SUPER[3] is funded by the European Commission, too. The duration is 36 months (ending March 2009) and it has a 11 million euro funding. The 18 participants are

[1] `http://www.fusionweb.org/Fusion/`, as at 2007-11-13

[2] project fact sheet available at: `http://cordis.europa.eu/fetch?CALLER=PROJ_IST&ACTION=D&DOC=28&CAT=PROJ&QUERY=1194967789105&RCN=79359`, as at 2007-11-13

[3] `http://www.ip-super.org`, as at 2007-10-25

both from research institutes and industry with some major players like SAP, IBM or IDS Scheer. [1]

SUPER *"aims at providing a semantic-based and context-aware frame-work, based on Semantic Web Services technology that acquires, organises, shares and uses the knowledge embedded in business processes within existing IT systems and software, and within employees' heads, in order to make companies more adaptive"*[SUPa]. Objectives are the development of a technological SBPM framework, new generic languages for e.g. processes or goal descriptions, automated annotation techniques, process query tools, mediation procedures and the adjustment of existing reasoners. Both horizontal and vertical ontologies have to be built as wells as tools for every stage of SBPM [SUPb]. In a manner of speaking quite all aspects of the SBPM vision are captured.

First results of the project are several available deliverables besides many scientific papers on SBPM related topics. An overview of ontologies built by the project and first suggestions are made in [BCD+07]. The discussion is about organizational related ontologies, an upper process ontology (notions of process models at business level), semantic EPC and BPMN as ontology versions of the well-known process modeling languages together with an ontology defining the commonalities of both, semantic BPEL and ontologies capturing events during process execution and process mining/analysis aspects. Also the already existing WSMO is used for the description of Semantic Web Service aspects.

Furthermore a language extension to BPEL, named BPEL4SWS, has been defined, which is used for the execution of business processes in the SUPER architecture [FHK+07]. A description of the architecture and design of the architectural layer, which is *"responsible for orchestrating the execution of the semantic activities according to the control and data flow"*[IIN+07, p.7] is also already available. Additionally a detailed Semantic Business Process life cycle [FSM+07], a design for a business process library [KKM+07] and a conceptual framework for business process mediation [DCC+07] have been introduced.

2.4.3 SemBiz

SemBiz[2] is a smaller research project than the two mentioned so far with participants from the Universities of Innsbruck and Vienna and two Austrian companies. The goal of this project is to use semantic descriptions of business processes in order to reach a business level perspective of Business Process Management instead of the current prevailing IT perspective.

The approach is to define a semantic description framework for business processes with WSMO as initial technology. With a framework like this, inference-based techniques can be

[1] project fact sheet available at: `http://cordis.europa.eu/fetch?CALLER=PROJ_IST&ACTION=D&DOC=18&CAT=PROJ&QUERY=1195046500310&RCN=79373`, as at 2007-11-14

[2] `http://www.sembiz.org`, as at 2007-11-15

developed for tasks like *"querying on business process spaces, discovering appropriate business processes for specific objectives, and composing business processes out of existing process fragments"*[Sem]. As result of the project a prototypical tool suit should make use of the new potentials.

Among the present results of the project are a first version of their Business Process Modeling Ontology (BPMO) [CYH07], the semantic description framework of SemBiz, and the introduction of a framework for semantic query, discovery and composition of business processes [DHH+07].

2.4.4 SEBIS - Semantics in Business Information Systems

SEBIS is a research group led by Prof. Dr. Martin Hepp from the University of Innsbruck, Austria.[1] They are working on the usage of Semantic Web/Semantic Web Service technologies like ontologies or machine-reasoning in several Business Information Systems related aspects, e.g. ontology-supported content integration, Business Process Management or data and knowledge engineering. The group is active in several research projects and provides a variety of scientific publications, mostly more generic considerations and methodologies.

2.4.5 FIT - Fostering Self-Adaptive e-Government Service Improvement using Semantic Technologies

Another interesting research project concerning SBPM with a concrete practical application is FIT[2]. The European Commission funds this project with 2.37 million euro, the 30 month duration expires in June 2008.[3]

Its major objective is the development of a self-adaptive e-government framework based on semantic technologies. Changing preferences and expectations of the citizens should be matched by the continually fitted quality of public services, in a proactive manner best. The goal is customized service-delivery to e-users.

One of the public deliverables so far is a conceptual framework for self-adaptive e-government with the FIT ontology, which defines all aspects relevant for the systems adaptivity, and the identification of the services of the FIT system (a service-oriented architecture) and their interfaces [AFH+06]. An interesting point is that business rules, quality of service and other 'semantic' aspects play an important role (even consideration on runtime) contrary to ordinary BPM solutions. Standards recommended in this architecture are BPEL, OWL/OWL-S

[1] `http://sebis.deri.org`, as at 2007-12-03
[2] `http://www.fit-project.org`, as at 2007-11-15
[3] project fact sheet available at: `http://cordis.europa.eu/fetch?CALLER=PROJ_IST&ACTION=D&DOC=23&CAT=PROJ&QUERY=1195137600375&RCN=78385`, as at 2007-11-15

and SWRL. Also a first version of the quality model and ontology (quality of e-government services) [MPC+07] as well as the concept and modeling approach of adaptable processes and rules execution [FHT07] have been published.

2.4.6 SEMPRO

The DFG (German Research Foundation) project SEMPRO at the University of Augsburg, Germany, aims at (semi)automatic creation and adjustment of process models via (semi)automatic planning of processes using semantic process activity descriptions.[1] With Semantic-based Planning of Activities (SEMPA) a planning algorithm is developed, that proceeds in three steps. First, input and output parameters of processes are semantically matched and the information stored in an action dependency graph, which is, once calculated, used in the following steps. Second, a forward-search (beginning from the initial state) collects all applicable processes from the graph that can help to achieve a certain goal. This basis is used in the last step to create the process model by using control structures and syntactical elements of a specific modeling notation, e.g. UML activity diagrams. For more information on the SEMPA algorithm see e.g. [BHHL07]

2.5 Presentation of the Business Process Modeling Notation (BPMN)

The Business Process Modeling Notation (BPMN) is an official OMG specification for a graphical notation of business processes. Version 1.0 [BPM04], developed by the Business Process Management Initiative (BPMI), was published in May 2004. In June 2005 the Business Process Management activities of BPMI and the Object Management Group merged and so BPMN is now part of the OMG specification process. Due to the great importance of the OMG, BPMN is likely to become the broadly accepted business process modeling standard as UML is for many other IT related models. This is the reason why BPMN has been picked as one of the notations to be evaluated under consideration of the patterns for Semantic Business Process Modeling from chapter 3. The work in this thesis relies on the draft of BPMN version 1.1 from June, 3rd 2007 [OMG07b], as it is the newest available version at the moment.

The intention of the Business Process Modeling Notation is to *"provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and*

[1] for more information on the project see e.g. `http://www.informatik.uni-augsburg.de/lehrstuehle/swt/vs/projekte/semantik/sempro/`, as at 2007-12-06

monitor those processes." [OMG07b, p.1]. One effort to bridge the gap between business perspective and later technical implementation is the mapping of BPMN to process execution languages. A mapping to BPEL4WS is provided in the specification, but BPMN was designed in a way that allows the creation of a mapping to any other process execution language.

BPMN defines a flowchart-based Business Process Diagram (BPD) which enables both the development of rather simple, easily understandable diagrams and the specification of arbitrarily complex business processes. The graphical representation is influenced by other common modeling notations in order to facilitate the readability. It is possible to draw three types of models: Private (internal) business processes, representing processes inside a single organization, abstract (public) processes for the interaction between a private process and a process of another business entity without knowing details about the internals of that other process and only showing activities concerned with the message exchange, and collaboration (global) processes between two or more partners, which can be shown as an interaction of several abstract processes (cf. [OMG07b, p. 12 et sqq.]). Besides the graphical representation of a business process, many elements have additional attributes, which become exceptionally necessary if the process is meant to be executed and mapped to an execution language.

There are four basic categories of elements in a BPMN diagram:

Flow Objects are either of type *Activity*, *Event* or *Gateway*.

Activities represent *"work that is performed within a business process"* [OMG07b, p.53] and their graphical appearance is a rounded-corner rectangle. An atomic *Activity* is called *Task*. It is possible to specify a *Task* type, e.g. *Service Task* (providing a service), *Receive Task* (waiting for a message) or *User Task* (human performer). A *Sub-Process* is a compound activity that contains another process. This construct can be used to produce diagrams of different granularity (a subprocess can be opened to show the process "inside") or to reuse process fragments. For *Tasks* and *Sub-Processes* additional markers state if an *Activity* is performed in a loop (either checking a boolean expression after each cycle or determining the number of iterations at the beginning) or used for compensation. Additionally, an ad-hoc marker is provided for *Sub-Processes* and there is the possibility to give a *Sub-Process* a transactional behavior [OMG07b, p.53 et sqq.]. A *Process* has no graphical representation, but can be seen as a part of the control flow, a set of *Activities* at any level. A *Process* is an *Activity* as well.

"An Event is something that "happens" during the course of a business process.(...) BPMN has restricted the use of events to include only those types of events that will affect the sequence or timing of activities of a process" [OMG07b, p. 36]. The graphical representation of *Events* is a circle with different boundaries for each type, an image of the event trigger can be placed inside. *Start* and *End Events* are responsible for the

start and end of processes while *Intermediate Events* can occur within the process to deal with delays, exception handling or show expected messages. Possible event triggers are e.g. *Timer*, *Message*, *Conditional*, *Error* or *Termination*. Event triggers can be "catched" and/or "thrown" by *Events*. [OMG07b, p.36 et sqq.]

"Gateways are modeling elements that are used to control how Sequence Flow[sic] interact as they converge and diverge within a Process" [OMG07b, p.71]. A *Gateway* is represented by a diamond shape, potentially with a symbol inside determining the type of the *Gateway*. The different types are *Exclusive Gateways* (two or more alternative paths, data-based or event-based), *Inclusive Gateways* (conditional expressions are evaluated, but the positive evaluation of one condition does not exclude the evaluation of the remaining), *Complex Gateways* (a complex expression provided by the modeler determines the behavior) and *Parallel Gateways* (create and synchronize parallel flow). [OMG07b, p.71 et sqq.]

Swimlanes are a concept to structure activities both inside a company (departments, roles, etc.) and in B2B scenarios. Therefore a *Pool* represents a participant in a process, either a specific business entity like an organization or a general business role like supplier. Each *Pool*, a square-cornered rectangle, serves as container for the processes and their control flow of one participant or is used as a "black box" activities from other *Pools* can interact with via message flows. *Pools* can be subdivided through *Lanes*. The meaning of such a, probably nested, partitioning is up to the modeler, but often used to represent departments and/or roles in a company. [OMG07b, p.87 et sqq.]

Artifacts provide the capability to add additional information to a process without direct relation to the process flow. BPMN contains three standard *Artifacts*, *Data Objects*, *Text Annotations* and *Groups*, but everyone is free and encouraged to define his own *Artifacts* and include them into the model. *Data Objects* (the often used document symbol) can represent every object that is used during a process. They can be associated with flow objects (e.g. for input/output) as well as with a *Sequence* or *Message Flow* symbolizing the transfer of the object. *Text Annotations* enable the modeler to enrich the diagram with additional information and comments for a better understanding and the *Group* artifact is an option to group elements of a diagram without further impact. [OMG07b, p. 94 et sqq.]

Connecting Objects are used to connect all the flow objects in a diagram. A *Sequence Flow*, a solid line with solid arrowhead, shows the order in which *Activities* are performed in a process of one participant (one *Pool*). The connection of *Activities* in different *Pools* via *Sequence Flows* is prohibited. Such *Activities* have to be connected with a *Message Flow* (dashed line with open arrowhead). Finally an *Association* (dotted line with line arrowhead) is used to connect *Artifacts* with flow objects, symbolizing input and output

of *Activities*.

The development of BPMN is not finished and the requirements for BPMN 2.0 have already been determined [OMG07a]. So far, BPMN 1.0/1.1 has no metamodel. But it is generally accepted, that a metamodel for the precise definition of semantics and an exchange format is necessary. With the Business Process Definition Metamodel (BPDM) a metamodel and a mapping from BPMN to BPDM concepts already exists, but BPDM is more robust than BPMN. BPMN 2.0 should provide a notation for all additional concepts. Important points for the development are the representation of choreography, assistance for different perspectives on a process model and the preservation of the layout after exchange of process models between different modeling tools.

A BPMN example process is presented in figure 2.4 at the end of this section. In order to make the example not too complex an internal process is shown, so there are no message flows to another pool. The diagram shows a very simplified process for the initiation of an internal project at a company by a division director. After creation of a first project specification, the management potentially has to authorize the project depending on its expected costs and gets the specification for its decision making. There is only a link to the complete evaluation and approval process of the management because of reutilization and simplification of this process. If the project has clearance, people for the project have to be found on the organization's internal job market and resources have to be provided concurrently. At the end of the process the project is ready to be launched.

For other short introductions to BPMN see e.g. [Whi04] or [OR03].

2.6 Presentation of AgilPro

Besides BPMN, with the modeling environment of the AgilPro toolbox another process modeling language is evaluated in chapter 4. The AgilPro project[1], funded by the federal state of Bavaria, aims at providing a slight process integration framework for Small and Medium-sized Enterprises (SMEs) containing modeling and simulation tools as well as adapter to existing ERP applications (cf. [BLPR07]). The AgilPro LiMo, the process modeling tool developed by the University of Augsburg, Germany, is also a contribution to the Eclipse Java Workflow Tooling (JWT)[2] project as Workflow Editor (WE) by now. The current release version is 1.4.

The modeling of the process flow in AgilPro is similar to other approaches. *Actions* (elliptic figures), connected via *Activity Edges* (solid line with solid arrowhead), show the process flow. The divergence and convergence of process flows is supported by parallel (thick

[1] `http://www.agilpro.de`, as at 2007-11-29
[2] `http://www.eclipse.org/jwt/`, as at 2007-11-29

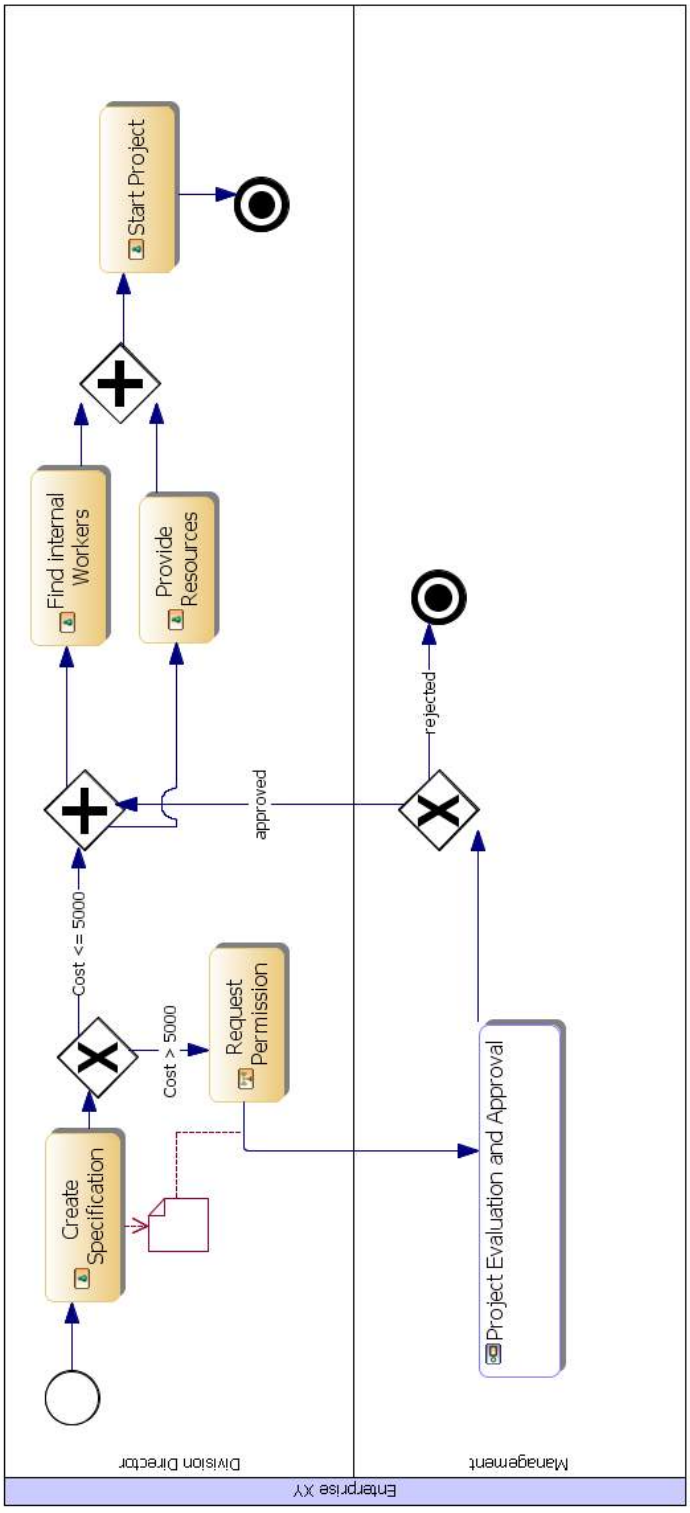


Figure 2.4: BPMN Example Process

solid line with a corresponding number of in-/outgoing activity edges) and exclusive (diamond shape) splits/merges. Besides the atomic actions, *Events* and *Subprocesses* can be integrated into the process flow, either by linking (can be opened in a different editor view) to another process or embedding it. The decision how the process flow proceeds after an exclusive split depends on the result of annotated boolean expressions. Start and end of a process are fixed by an *Initial Node* (blank circle) and an *End Node* (black filled circle inside a white circle) respectively.

An important fact is the possibility to create *Roles*, *Applications* and *Data* (each with its own, customizable icon) outside a process and link them to *Actions* inside processes via *References* (dashed line, sometimes with an arrowhead). So it is possible e.g. to create the entire organizational structure or application landscape of an organization before starting to model its business processes. *Roles* specify, who is responsible and/or allowed to accomplish a certain task, while *Applications* and *Data* (and additionally modeled *Data Types*) define which (software) application is used for a certain task and what are the input and output data. All this information can be used to simulate the process in the AgilPro simulation tool. The change of roles is as well supported as the evaluation of boolean expressions for the exclusive choices and even the start of the applications (depending on the accuracy of the model and the existence of adapter for the applications). Contrary to many other business process modeling tools/notations, AgilPro allows to show one and the same process in different *Views*, e.g. a business and a technical view, where only those elements, that are of interest for certain users, are presented. It is possible to add *Comments* to every object of a process, but they are only visible in a preference window.

To get an idea how a business process modeled with AgilPro looks like in comparison to BPMN, the same example as in the foregoing section has been used for the illustration (Figure 2.5). In AgilPro it is not possible to link *Roles* or *Data* to *Subprocesses* (like the evaluation process in this example), the linking has to be done in the model of the *Subprocess* itself. But this is no problem, as because of the *Reference* concept, the same entity can be used in several models. To show the usage of *Applications*, the action for finding new workers uses an intranet job exchange.

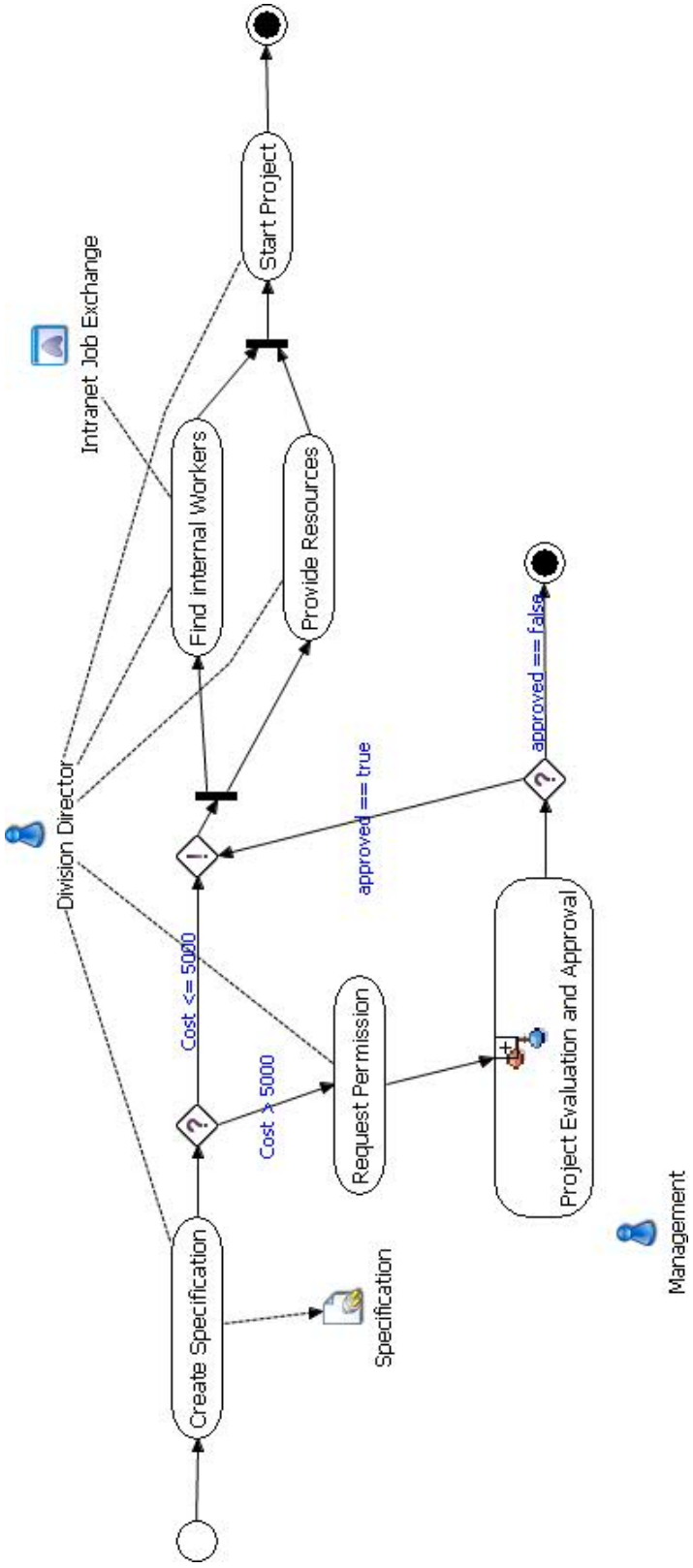


Figure 2.5: AgilPro LiMo Example Process

3 Patterns for Semantic Business Process Modeling

Semantic Business Process Modeling is the key element for Semantic Business Process Management as semantically annotated process models are the source for all further processing. Despite the general agreement on the necessity of semantic annotation there is nothing like a guideline how such process models should look like, what they should contain and which elements should be enhanced semantically in order to benefit from SBPM. The patterns introduced in this chapter show which steps have to be taken to get a 'SBPM-ready' business process model.

Patterns play a very important role in computer science. They often describe common problems, solutions and prevalent practices within a certain domain of interest. The presumably most famous ones are the design patterns for object-oriented software development by the Gang-of-Four (GoF) [GHJV95]. Also in the field of business process modeling several patterns have already been developed that describe common constructs and characteristics of certain workflow aspects, e.g. control-flow [RtHvdAM06] or data related concepts [RtHEvdA04], with regard to existing solutions and requirements. Following those exemplars, patterns for the SBPM perspective of business process modeling are presented in this thesis under consideration of existing business process modeling solutions such as BPMN. The patterns define what has to be done additionally to the 'normal' process modeling and make a general proposal how this could be realized with a process modeling notation and should be supported by it.

The presentation style is intentionally similar to the existing workflow patterns as the patterns in this paper are supposed to build on them. They have the following profile:

- **Description:** A short description of the pattern.
- **Example:** One or more practical examples to illustrate the usage and intention of the pattern.
- **Motivation:** Idea behind and reasons for this pattern as well as SBPM-tasks that necessarily need this pattern.
- **Implementation:** Recommendations on how the pattern could be realized with regard to preferably easy integration with existing process modeling solutions.

- **Evaluation Criteria:** Definition of criteria for the evaluation of process modeling notations. They specify if a notation is fully capable of supporting the pattern or not.

The patterns for Semantic Business Process Modeling are differentiated by four groups depending on their role. The Base Patterns are the foundation for all the other patterns, as they deliver the necessary concepts for advanced procedures. Functional Patterns cover all functional workflow aspects that are important before, during and after execution of a process step or larger parts of a process. They deal with the same topics as the description of Semantic Web Services and therefore establish a connection between modeled business processes and services that execute them. Besides the functional aspects, technological infrastructure, organizational structure and their connection to a business process play an important role in Semantic Business Process Modeling. This is outlined in the Organizational Patterns. Finally, the Conditional Patterns examine process relevant conditions like internal regulations, legal restriction or contract conditions.

During the description of each pattern, single tasks are referred to as 'activities'. Activities do not denote entire process models as in the description/metamodel of some modeling notations like e.g. AgilPro.

3.1 Base Patterns

The patterns presented in this section are premise for nearly all other Semantic Business Process Modeling patterns. While Semantic Annotation is the key of SBPM at all, the Grouping pattern provides additional benefits for many other patterns.

3.1.1 Pattern 1 (Semantic Annotation)

Description: Annotation of model elements with semantic information (particularly by using ontologies). This covers both simply the identifiers of model elements and elements/-concepts like e.g. non-functional properties themselves.

Example: In a business process model of a commercial bank each identifier of model elements such as activities, events or artifacts is connected with concepts of a commonly used and accepted domain ontology from the financial sector.

Motivation: This pattern is the precondition for all following patterns and the application of Semantic Business Process Management in general. It provides a machine-interpretable version of the meaning of an entire business process as well as its single parts and elements. This allows automatic reasoning, which leads e.g. to advanced search methods

(consider dependencies and references or compute implicit information), an easier collaboration of models from different authors (a merge or reuse is much easier because of the same used domain ontology), automatic search of services and execution or validation of certain guidelines to name just a few. All the thinkable applications of SBPM depend on the usage of ontologies and this pattern provides the link between process model and ontology.

Implementation: There are several approaches how the annotation of process models can be realized. One idea, proposed by [TF07], is to use a metadata-level between model and ontology. Only one ontology is used for model constructs (classes in the ontology) and domain information (instances). The process model and its constructs are represented on a metadata-level through instantiation of the classes in the ontology. This level is an exact mapping of the process. The linkage between this metadata and the domain information takes place via properties. Figure 3.1 illustrates this approach. *semType* is the property used in this very simplified example to link metadata and the domain information part of the ontology. The disadavantage of this approach is that both domain and notation information have to be represented in one ontology, which makes it hard to use existing domain ontologies.

To avoid this problem, one could think about using different ontologies for the metamodel of the modeling notation and the domain of interest. A transformation between the different technological spaces of ontology and modeling notation (cf. [GDDD04]) allows to connect a model and the ontology describing the metamodel of the modeling notation. After that other ontologies covering the domain of interest can be applied to semantically lift the model. Figure 3.2 tries to illustrate this approach.

Another rather simple approach would be to extend the metamodel of the modeling notation with an additional property for every significant element that captures the necessary information and provides the link to an ontology. The disadvantage of this approach is naturally, that changes to the metamodel of every modeling notation would be necessary. Also the reasoning and other computation in the course of SBPM applications would have to filter the necessary information out of the entire process model, while with the other approaches ontological perspective and the classic model can be considered separately.

Regardless of which implementation approach is chosen, it is always necessary to know which specific ontology has been used. This is important if e.g. different versions of an ontology are used over time or if there are many different ontologies available in a large company because of different business domains. It is also thinkable that within one process model different ontologies are used to annotate model elements, e.g. to describe aspects from different areas that normally do not occur together (and

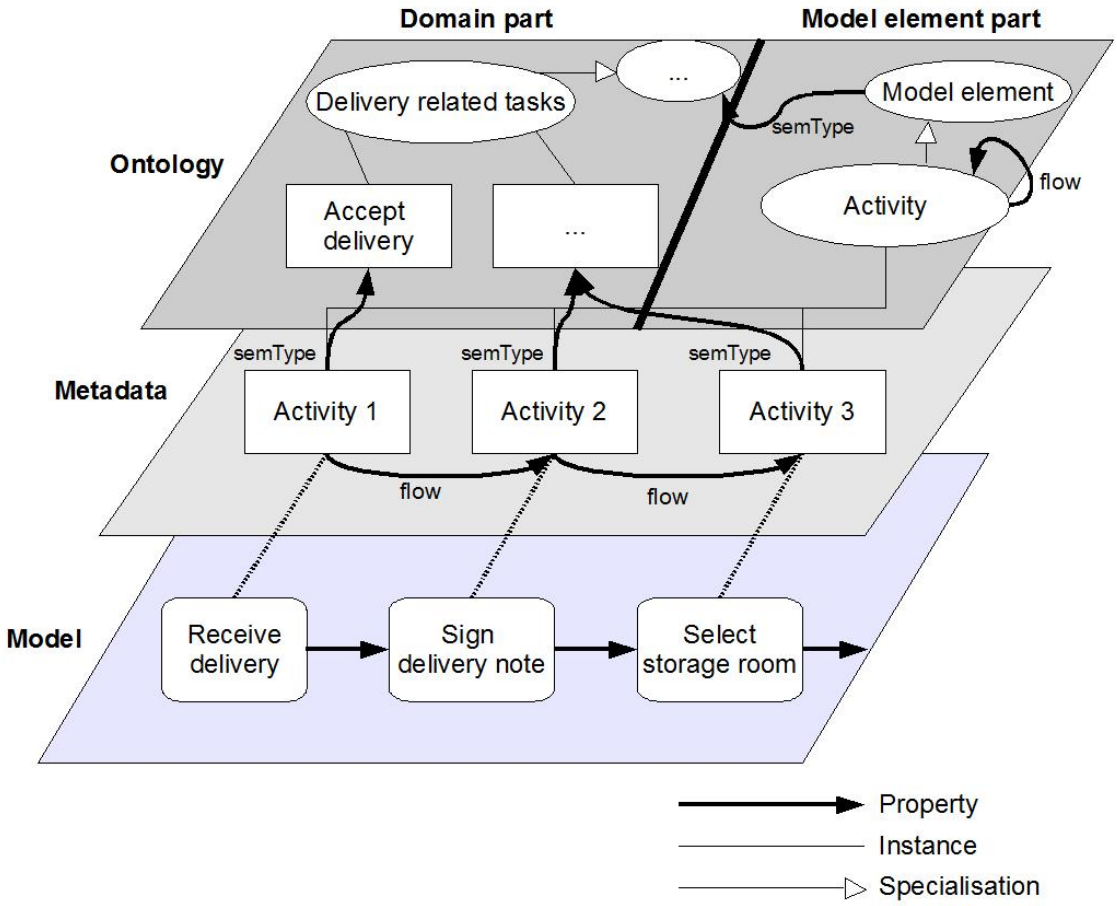


Figure 3.1: Semantic Annotation of process models using metadata

therefore no ontology exists that combines both). The knowledge on which ontologies have been used during the modeling phase is crucial for successful reasoning and other computation afterwards. Ontology languages like OWL use Unique Resource Identifiers for the reference to entire ontologies or their elements and this assures, that the used ontology is always clear.

Evaluation Criteria: Of course the requirements a business process modeling notation has to fulfill in order to support this pattern depend on the chosen implementation approach. Therefore a notation supports this pattern, if at least one implementation approach is possible, which will be in the majority of cases. If there is a common understanding in the research community which implementation approach is the right one, this evaluation criteria might have to be changed.

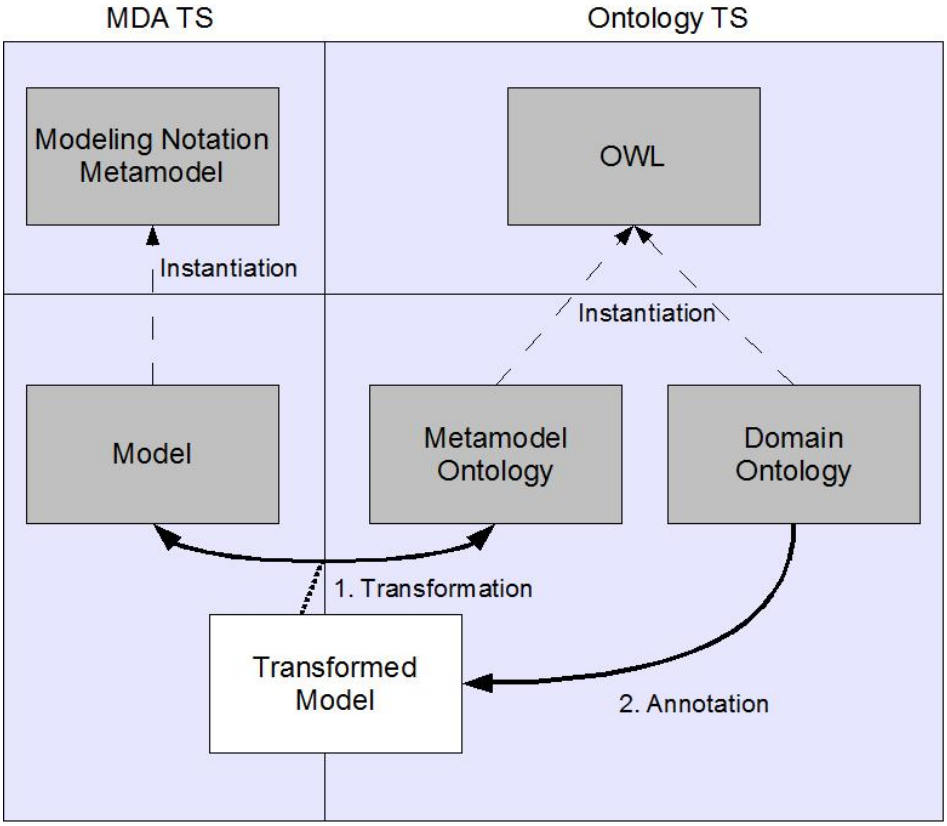


Figure 3.2: Semantic Annotation of process models using transformation between technological spaces

3.1.2 Pattern 2 (Grouping)

Description: Divide a business process model into several sections by grouping parts of the control flow in order to allow semantic annotation of cohesive process fragments.

Example: A business process describing the manufacturing of goods contains the shipment to the customer. Several process steps like packaging, loading etc. have to be performed, but all those steps are done by the same organization unit and have the overall goal that the incoming product is shipped to the customer. Instead of annotating every step of this part of the business process with the same semantical information, the process steps are grouped together and this group is semantically enriched as a whole.

Motivation: This pattern is useful for two main reasons. On the one hand it reduces unnecessary additional work. Many times some consecutive activities in a business process are performed by the same person and the same IT equipment, have the same artifact as input and output or are liable to the same regulations. All those things can be semantically annotated and this work would be the same for every of those process steps. So the grouping of process steps and semantic annotation of only that group produces

a great cutback of repetitive work.

On the other hand, goals can affect different ranges within a model: only one process step or a group of them (from very small to rather big groups). The search for existing process fragments or suitable Semantic Web Services for the execution can be made easier and more successful by the definition of goals for different levels.

Implementation: There are several approaches conceivable on how to group parts of a process model. One is to use a model construct that allows the modeler to draw a scope around certain parts of a process model. Everything inside this scope is part of the group. The appearance of the process model is not changed, contrary to the other approach where constructs like subprocesses or referencable processes (cf. the description of the BPMN or AgilPro notation in section 2.5 and 2.6 respectively) are used. Those concepts affect the appearance of the process model, as they group parts of the process and show them as single activity in the model, while the details of the subprocess are specified elsewhere. However, both approaches have to provide the same annotation abilities and connection with elements (like input/output or IT-system) as a single process step (activity). Therefore it is conclusive to model the grouping element as child of the activity in the metamodel of the modeling notation.

Evaluation Criteria: A business process modeling notation provides full support of this pattern if an element to group parts of the process model exists as well as this element has the same interaction possibilities (semantic annotation and connections to other model elements) as a single activity.

3.2 Functional Patterns

The functional patterns of this section contain aspects that are also used to describe Semantic Web Services. Input and Precondition give information about the situation before the execution of a process (step) and Output, Postcondition and Effect afterward. The Nonfunctional Properties specify parameters that are relevant for the execution and the Context pattern provides a container that combines the other aspects for easier reuse and further processing.

The chosen patterns follow in some way the functional description of Semantic Web Services in OWL-S and comprehend the same concepts. One could debate on the integration of additional aspects from other SWS ontologies like WSMO. For example, WSMO uses not only Effects (such as OWL-S) to describe the functionality of a web service, but also Assumptions (the state of the world before execution of the service). As this thesis concentrates more on OWL-S, Assumptions were left out. Furthermore it is not so clear what could be of such high importance before the execution of a process step, that is not a precondition. Effects

(state of the world after execution) instead are closer to reality as they describe the impact of a process execution.

3.2.1 Pattern 3 (Input)

Description: Define the input an activity/group of activities requires for its successful execution semantically. Input covers electronic data and information as well as physical objects.

Example: The *Accept Delivery* activity of a business process has the following inputs: The physical item delivered to the stock and an electronic shipping note with all information about the item, supplier, price, conditions and so on.

Motivation: Semantic annotation of input is of great value for several reasons. Innovative queries on the process space become possible. The usage of ontologies enables business analysts e.g. to search the processes of a company for inputs that consist of a specific material, require special treatment or other important information that are not visible by just looking at the process model. Also the specification of input together with other parameters enables automatic search for existing process fragments stored in a business process repository for further reuse. Electronic input data helps to find suitable Semantic Web Services for the task at hand and to support automatic composition and execution of such processes. Even business analysis after execution is supported if the input of process steps is semantically enriched. It becomes much easier to generate implicit information out of the process logs.

Implementation: Examining the implementation of this pattern, one has to consider how an input should be connected to an activity. An easy approach would be to use an "Input" model element, connect it with the desired activity and annotate the semantic information by just concentrating on the input element, which means e.g. to connect the identifier with an ontology.

Another approach is to use ontologies, that have originally been developed to describe Semantic Web Services. In the following the concentration will be on OWL-S, but WSMO or other similar concepts should do the same job. OWL-S uses a *Service Profile* to describe what the service does. Looking at the components of such a Service Profile (Figure 3.3, slightly adapted from the OWL-S submission[1], illustrates the parts of the OWL-S service profile relevant for the functional Semantic Business Process Modeling patterns), it becomes obvious that there is a striking resemblance of the description of Semantic Web Services to the description of activities we need in Semantic Business Process Management. The great advantage of this approach is, that the search

[1] `http://www.w3.org/Submission/OWL-S/`, as at 2008-02-15

for matching Semantic Web Services becomes much easier, because the necessary information is already in the same format that is used for the description of the services. Otherwise, it would be necessary to extract and express the OWL-S description (or WSMO or whatever is used) out of the process model first before the search can begin.

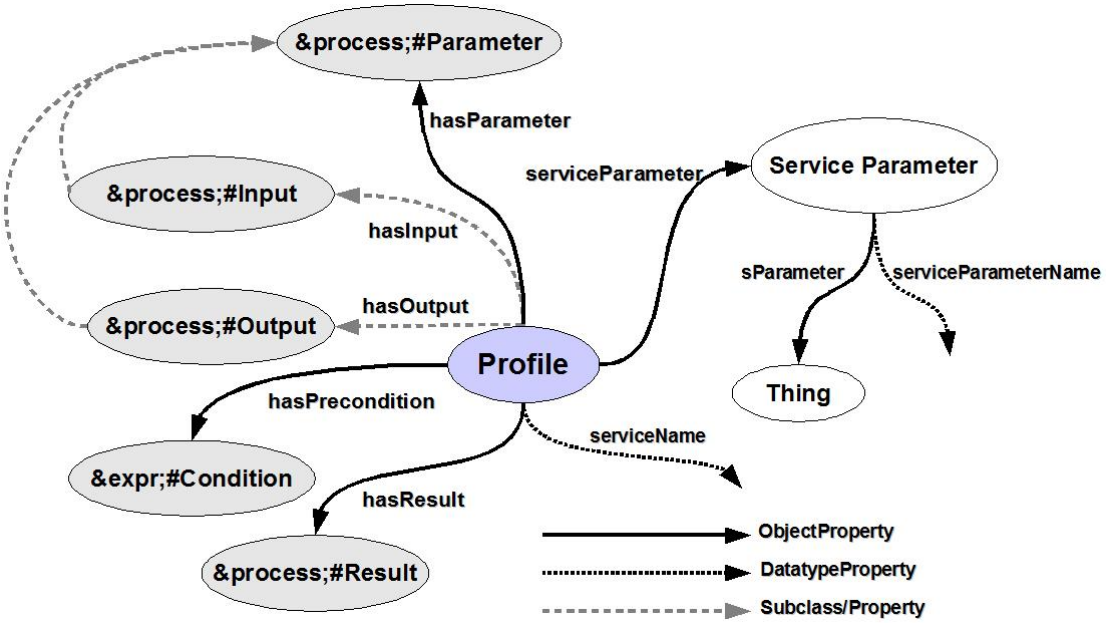


Figure 3.3: Selected classes and properties of the OWL-S Service Profile

Now the apparent idea is to describe activities in a business process model using OWL-S. The activity becomes the *Profile* in OWL-S and the *hasInput* property of the *Profile* is used to describe the input of the activity. Of course this approach also allows to display the input as an own modeling element. This is preferable because of the greater reuse possibilities, but not absolutely necessary. An additional property of the activity should work as well.

In OWL-S Inputs are subclasses of Parameter and therefore have a type as can be seen in the part of the OWL-S definition[1] shown below.

```
<owl:DatatypeProperty rdf:ID="parameterType">
  <rdfs:domain rdf:resource="#Parameter"/>
  <rdfs:range rdf:resource="&xsd;anyURI"/>
</owl:DatatypeProperty>

<owl:Class rdf:ID="Parameter">
 <rdfs:subClassOf>
   <owl:Restriction>
     <owl:onProperty rdf:resource="#parameterType" />
```

[1]W3C member submission of OWL-S at `http://www.w3.org/Submission/OWL-S/`, as at 2008-02-19

```
      <owl:minCardinality rdf:datatype="&xsd;#nonNegativeInteger">
                1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="Input">
  <rdfs:subClassOf rdf:resource="#Parameter"/>
</owl:Class>
```

Listing 3.1: OWL-S Specification: Parameter and Input

An example input described in OWL-S is shown in the following listing. The referenced concept of a shipping note is the input of the AcceptDelivery task.

```
<process:Input rdf:ID="ShippingNote">
  <process:parameterType rdf:datatype="&xsd;#anyURI">#ShipNote
  </process:parameterType>
</process:Input>

<owl:Class rdf:ID="Accept_Delivery">
  <rdfs:subClassOf rdf:resource="#Profile"/>
  ...
  <profile:hasInput rdf:resource="#ShippingNote"/>
  ...
</owl:Class>
```

Listing 3.2: OWL-S Example: Input

Another issue the implementation should consider is the difference between electronic and physical input. A flag stating that an input is physical and not some sort of information data helps to prevent searches for SWSs where no computer is used in the process step. Also services for activities with both physical and electronic input can be found easier if it is clear, which input has to be considered.

Evaluation Criteria: A business process modeling notation supports this pattern if it is possible to model the input of an activity in some way and enrich this input semantically. The flag for physical/electronic input is not necessarily required.

3.2.2 Pattern 4 (Output)

Description: Define the output an activity/group of activities requires for its successful execution semantically. Output covers electronic data and information as well as physical objects.

Example: The *Assemble Doors* activity as part of a car manufacturing process has the following outputs: the car framework, now with doors attached, and the electronic assembly plan and documentation with new entries, that is passed from working station to working station.

Motivation: Actually the motivation for the output pattern is the same as for the input pattern. For example, semantic annotation of output makes improved search methods, automatic search and reuse of process fragments or the automatic search and execution of Semantic Web Services for this process step possible. Output is the counterpart of input. Often the input of a process is also the output (maybe with some alterations or additions), but it is also very common that a process step produces a complete new output, emerging the first time in the course of the process.

Implementation: The possible implementation approaches for this pattern are the same as for the Input pattern. Again either the connection of an additional output element with an ontology or the application of a SWS ontology like OWL-S or WSMO (cf. the explanation given in the implementation section of the input pattern) is a possible solution.

Considering OWL-S, where the Service Profile will be used to describe an activity, the *hasOutput* property of the *Profile* is obviously appropriate to semantically describe the output of an activity. The next listing is the definition of Output in OWL-S, leaving out the fragments already presented in the previous pattern:

```
<owl:Class rdf:ID="Output">
  <rdfs:subClassOf rdf:resource="#Parameter"/>
</owl:Class>
```

Listing 3.3: OWL-S Specification: Output

An additional flag for physical/non-physical output is as beneficial as for the input (cf. explanation at the previous pattern). The following listing shows how outputs can be defined using OWL-S:

```
<process:Output rdf:ID="AssemblyDocumentation">
  <process:parameterType rdf:datatype="&xsd;#anyURI">#AssemblyDoc
  </process:parameterType>
</process:Output>

<owl:Class rdf:ID="AssembleDoors">
        <rdfs:subClassOf rdf:resource="#Profile"/>
  ...
  <profile:hasOutput rdf:resource="#AssemblyDocumentation"/>
  ...
```

```
</owl:Class>
```

Listing 3.4: OWL-S Example: Output

Evaluation Criteria: A business process modeling notation supports this pattern if it is possible to model the output of an activity in some way and enrich this output with semantic information. The flag for physical/electronic output is not necessarily required.

3.2.3 Pattern 5 (Precondition)

Description: Specify the preconditions of an activity that will only be executed properly, if all preconditions are true.

Example: Before the salary payment can be performed by the human resources department, it has to be the last workday of the month and the time sheet of an employee has to be filled out correctly.

Motivation: Preconditions play an important role in modeling and execution of business processes. They help to find suitable existing process fragments or Semantic Web Service and to analyze failures during process execution.

The impact of preconditions can be seen in two ways. Either they determine under which conditions it is allowed to start and perform an activity or they guarantee, that the activity will be performed successfully if the preconditions are true. The first perception is rather strict and sometimes the process modeler may want to allow the execution of an activity even if not all preconditions are fulfilled. But the second perception has the disadvantage that there is a need to specify what happens if the activity is performed ignoring the preconditions and not completed successfully. A complex error handling or a fix specification becomes necessary. Therefore one has to trade off between the different perceptions and choose the one that seems to fit better.

Implementation: The idea how to implement preconditions is to use logical expressions. Again there is the possibility to annotate such expressions simply as an additional property of an activity with no connection to anything else or to use an SWS ontology like OWL-S or WSMO to embed this property. As the description of the functional patterns focuses on OWL-S, the implementation of preconditions using this approach is described in more detail.

The *Profile* in OWL-S has the property *hasPrecondition* to describe preconditions using logical formulas. There is no regulation on which logical language has to be used. It is possible to use either string literals (languages like KIF [KIF98] and PDDL [Gha98]) or XML-literals, enabling e.g. the use of SWRL. *Conditions* in OWL-S are subclasses

of *Expression*, the expression language is annotated and the expression stated in the *expressionBody* property.

```
<owl:Class rdf:ID="Condition">
   <owl:subClassOf rdf:resource="&expr;#Expression"/>
</owl:Class>

<owl:Class rdf:ID="Expression">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#expressionLanguage"/>
      <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">
                  1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#expressionBody"/>
      <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">
                  1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>

<owl:ObjectProperty rdf:ID="&expr;#expressionLanguage">
  <rdfs:domain rdf:resource="&expr;#Expression"/>
  <rdfs:range rdf:resource="&expr;#LogicLanguage"/>
</owl:ObjectProperty>

<owl:DatatypeProperty rdf:ID="expressionBody">
  <rdfs:domain rdf:resource="#Expression"/>
</owl:DatatypeProperty>
```

Listing 3.5: OWL-S Specification: Condition

The following listing exemplifies the definition of preconditions using OWL-S. A precondition is defined so that a correct time sheet has to exist (details on what makes the time sheet correct are left out for this example). This precondition occurs in the profile description of the salary payment activity.

```
<process:hasPrecondition>
  <expr:SWRL-Condition rdf:ID="TimeSheetExists">
    <rdfs:label>timeSheetExists(Timesheet)</rdfs:label>
    <expr:expressionLanguage rdf:resource="&expr;#SWRL"/>
    <expr:expressionBody rdf:parseType="Literal">
      <swrl:AtomList>
        <rdf:first>
          <swrl:IndividualPropertyAtom>
```

```
            <swrl:propertyPredicate rdf:resource="#timeSheetExists"/>
            <swrl:argument1 rdf:resource="#TimeSheet"/>
          </swrl:IndividualPropertyAtom>
        </rdf:first>
        <rdf:rest rdf:resource="&rdf;#nil"/>
      </swrl:AtomList>
    </expr:expressionBody>
  </expr:SWRL-Condition>
</process:hasPrecondition>

<owl:Class rdf:ID="SalaryPayment">
        <rdfs:subClassOf rdf:resource="#Profile"/>
  ...
  <profile:hasPrecondition rdf:resource="#TimeSheetExists"/>
  ...
</owl:Class>
```

Listing 3.6: OWL-S Example: Precondition

OWL-S makes no statement about how to deal with processes/activities that are performed even if the preconditions are not true. This lies in the scope of duties of the business process modeling notation. It should either be possible to model error handling in case of inaccurate process executions or a clear general definition of consequences is provided by the modeling notation.

Evaluation Criteria: This pattern is supported by a business process modeling notation if there is a possibility to define preconditions for activities. Handling of process steps, that are executed even though the preconditions are not satisfied, is an additional plus.

3.2.4 Pattern 6 (Postcondition)

Description: Specify the postconditions of an activity. They determine the conditions for a successful execution of a process (step). While postconditions state *when* an execution is successful, effects (see next pattern) describe *what* is the impact of the execution.

Example: After completion of the *Purchase Goods* activity the correct type and amount of items have to be stocked in the storage area. If there are either missing or wrong goods, this process step was unsuccessful.

Motivation: In many cases it is not obvious at first glance what makes the execution of a process task successful. And even if the human reader is able to infer the conditions, there is no way a computer could do that automatically. Therefore the postconditions, that define if an activity is executed successfully or not, have to be annotated explicitly in a machine-readable form. Just like the preconditions, this helps to detect reusable

process fragments and Semantic Web Services or to improve the analysis of executed process steps. Again one has to think about solutions on how to deal with activities, that were performed deficiently. The postconditions only help to identify such activities, but make no statement about the implications.

Implementation: The implementation approach is very similar to the one identified for preconditions. Logical expressions are used to describe the conditions in a machine-interpretable manner. Those expressions can either be annotated by only adding another independent property to an activity or use the SWS ontologies like OWL-S or WSMO.

Contrary to WSMO, where the postcondition is a property on its own in the functionality description (*Capability*), OWL-S applies a little different concept. The *Service* that normally describes the functionality of a Semantic Web Service, but the activity in this adapted case, has an *hasResult* property that covers several aspects. A *Result* consists of several parts (properties of the Result). The *inCondition* property is responsible for the specification of the condition under which this result occurs, meaning the outputs and effects of this result ensue (defined via *withOutput* and *hasEffect* properties). Now this *inCondition* property can be used to describe the postconditions in the same way as the preconditions (cf. the Precondition pattern for applicable languages), which can be seen in the part of the OWL specification below. The *hasResultVar* property is used to declare variables that are bound in *inCondition*

```
<owl:Class rdf:ID="Result">
  <rdfs:label>Result</rdfs:label>
</owl:Class>

<owl:ObjectProperty rdf:ID="inCondition">
  <rdfs:label>inCondition</rdfs:label>
  <rdfs:domain rdf:resource="#Result"/>
  <rdfs:range rdf:resource="&expr;#Condition"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasResultVar">
  <rdfs:label>hasResultVar</rdfs:label>
  <rdfs:domain rdf:resource="#Result"/>
  <rdfs:range rdf:resource="#ResultVar"/>
</owl:ObjectProperty>
```

Listing 3.7: OWL-S Specification: Result and inCondition property

The next listing illustrates the declaration of a *Result* and the *inCondition* property and how to add them to a profile:

```
<process:hasResult>
  <process:Result rdf:ID="PurchaseGoodsResult">
    ...
```

```
    <process:hasResultVar>
      <process:ResultVar rdf:ID="ItemType">
      <process:parameterType rdf:datatype="&xsd;#anyURI">&concepts;
      #ItemType</process:parameterType>
    </process:ResultVar>
    </process:hasResultVar>
    <process:inCondition>
    <expr:SWRL-Condition>
        --- here is the SWRL expression, for more details see the example
        in the Precondition pattern ---
    </expr:SWRL-Condition>
    </process:inCondition>
    ...
  </process:Result>
</process:hasResult>

<owl:Class rdf:ID="PurchaseGoods">
        <rdfs:subClassOf rdf:resource="#Profile"/>
  ...
  <profile:hasResult rdf:resource="#PurchaseGoodsResult"/>
  ...
</owl:Class>
```

Listing 3.8: OWL-S Example: Result with inCondition property

Evaluation Criteria: This pattern is supported by a business process modeling notation, if there is any possibility to define postconditions for activities. The handling of deficiently executed process steps is an additional plus.

3.2.5 Pattern 7 (Effect)

Description: Specify the effect(s) of an activity. Effects describe the changes to the world after successful execution of a process step.

Example: After completion of the *Purchase Goods* activity, the purchaser obtains ownership of the items and because of the bought amount, the discount for further trades has increased.

Motivation: Effects help to describe the functionality of an activity. They give an idea on what the process step has achieved and/or changed. Such as with postconditions, a human reader might be able to infer from the identifiers what the effects could be, but this is not explicit and an automatic processing by the computer is again impossible. In search for reusable process fragments or Semantic Web Services effects play an important role to identify appropriate findings and if this search should be performed automatically, a machine-readable definition of effects is significant.

Implementation: Such as preconditions and postconditions, effects can be described using logical expressions. Therefore the implementation approach resembles the one for the other two patterns. The choice again is between an independent property for the effect or the embedding into an SWS ontology. While WSMO has an own property for the effect, in OWL-S it is part of the *Result* concept (cf. the description given in the Postcondition pattern). The appropriate property of the Result is *hasEffect* and the definition of the expressions works just the same as for pre- and postcondition:

```
<owl:ObjectProperty rdf:ID="hasEffect">
  <rdfs:label>hasEffect</rdfs:label>
  <rdfs:domain rdf:resource="#Result"/>
  <rdfs:range rdf:resource="&expr;#Expression"/>
</owl:ObjectProperty>
```

Listing 3.9: OWL-S Specification: hasEffect property

Again a little example is given to illustrate the specification of effects with OWL-S, the result element is the same as in the previous pattern:

```
<process:hasResult>
  <process:Result rdf:ID="PurchaseGoodsResult">
    ...
    <process:hasEffect>
    <expr:SWRL-Condition>
        --- here is the SWRL expression, for more details see the example
        in the Precondition pattern ---
    </expr:SWRL-Condition>
    </process:hasEffect>
    ...
  </process:Result>
</process:hasResult>
```

Listing 3.10: OWL-S Example: hasEffect property

Again the effects of deficiently executed process steps should be considered in some way as well, but this is not absolutely necessary for the application of SBPM.

Evaluation Criteria: A business process modeling notation supports this pattern, if it is possible to define effects that occur after successful execution of activities.

3.2.6 Pattern 8 (Nonfunctional Properties)

Description: Declare nonfunctional properties that are relevant for an activity. They can cover e.g. financial, performance and reliability aspects or any other properties that are important for a process step. The decision which properties are necessary and reasonable

is up to the process modeler, but they have to exist in an underlying ontology so that automatic reasoning is possible.

Example: During the business process of an online shop, the activity *Check Credit Card* is applied. The check is performed by the Web Service of another company and there are several guarantees made by the service provider. For example, the reliability is 99,97%, up to 500 simultaneous queries are supported and the result is sent back within 5 seconds maximum. Also every query is accounted for with 0.03 €.

Motivation: The specification of nonfunctional properties of activities provides several advantages. It enables improved analysis of existing processes and planning of new/adapted ones, as aspects like cost or duration can be considered. It also enhances the search for suitable services. All services, whose nonfunctional properties do not satisfy the nonfunctional properties defined in the process model are discarded. Now it is possible to allow only services that e.g. have a given accuracy, performance, reliability, scalability or use a specific language. Nearly all nonfunctional properties that were detected for Semantic Web Services are applicable for the description of process activities and the modeler is free to think of additional ones that suite his specific needs. The definition and survey of business rules, regulations or contract conditions requires nonfunctional properties, as they are often subject of such rules. Last but not least after process execution the business analyst is able to compare the nonfunctional properties as they were modeled and their real values.

Implementation: Again there are the two ideas how to implement nonfunctional properties: an independent additional property of the activity or the property is embedded into a SWS ontology like WSMO or OWL-S. No matter which approach is chosen, it is of high importance, that both nonfunctional properties and their values (except it is a numerical value) are based on an ontology.

To stay in line with the description of the OWL-S approach presented so far for the previous patterns, the implementation of nonfunctional properties using OWL-S is described in more detail. The *Profile* has a *Service Parameter* property, that can be used to model nonfunctional properties (*serviceParameterName* can be described by using a URI instead of a simple literal). The *sParameter* property of the *Service Parameter* points to the value of the parameter within some OWL ontology. The subsequent listing shows how the definition of nonfunctional properties can be done using OWL-S. The reliability of a service that validates a credit card can be specified with different levels (here Highest is chosen) that are modeled in another ontology (RelLevel).

```
<profile:serviceParameter>
   <addParam:reliability rdf:ID="ServiceReliability">
   <profile:serviceParameterName>CheckCrediCardService Reliability
   </profile:serviceParameterName>
```

```
    <profile:sParameter rdf:resource="&RelLevel;#Highest"/>
    </addParam:reliability>
</profile:serviceParameter>
```

Listing 3.11: OWL-S Specification: serviceParameter

Evaluation Criteria: A business process modeling notation provides support for this pattern, if there is a way to attach nonfunctional properties to an activity.

3.2.7 Pattern 9 (Context)

Description: Unite the input, output, preconditions, postconditions, effects and nonfunctional properties of an activity/group of activities to a Context for easier handling and further reuse. The Context describes the intention of an activity by subsuming all relevant aspects that were previously specified on their own. A Context pertains to a single activity, a group of them or the entire process.

Example: The part of a business process that covers the manufacturing of heavy machinery has a Context that contains the input of several raw material and a construction plan, the output is the assembled machine, there is no precondition but a postcondition stating that the machine has to complete several tests successfully and nonfunctional properties like a maximum production time of four workdays.

Motivation: At first glance it might seem a little bit dispensable to only unite several properties of an activity without adding any new information. But the introduction of a Context construct provides several benefits. The reuse of the functional description of an activity is much easier if one element contains all relevant information. Otherwise tedious gathering and copying of every single information would be inescapable. Also the search for process fragments or Semantic Web Services that match the Context description becomes faster if all aspects are available in a compact form, at best even in a form that is equal to the one that was used to describe the services. The Context also builds a scope where it is clear that the same elements are considered in precondition, postcondition etc. while in other Contexts the elements are different entities.

Implementation: The implementation of the Context pattern should offer a construct that works as a container for input, output, precondition, postcondition, effect and nonfunctional properties. Now it becomes much easier if an implementation approach for the previous patterns was selected, that relies on SWS ontologies like OWL-S or WSMO, because they provide such a container innately (*capability* in WSMO and *Service Profile* in OWL-S). Attach a *Service Profile* to an activity/group of activity and include the relevant properties as described in the patterns above and you already have your container that serves as implementation of a Context (see the abstract listing below).

```
...
<profile:serviceName>ActivityName</profile:serviceName>
...
<profile:hasInput rdf:resource="#Input1"/>
...
<profile:hasPrecondition rdf:resource="#Precondition1"/>
...
<profile:hasResult rdf:resource="#Result2"/>
...
<profile:hasOutput rdf:resource="#Output4"/>
...
<profile:serviceParameter>
   //some serviceParameter definition
</profile:serviceParameter>
```

Listing 3.12: OWL-S Example: Profile

Annotate the activity with the Context (*Service Profile*) instead of just lifting the identifier semantically (the *serviceName* of the profile can play the role of the identifier), and no further changes to the modeling notation become necessary. Of course, a business process modeling environment has to provide the modeling of all the properties anyway. If some of the properties like input/output are already part of the modeling notation, an integration into the *Service Profile* can be carried out automatically.

If the implementation approach with independent properties has been chosen, now is the time to think about their connection. An additional Context construct attachable to an activity might be necessary to achieve this, which implies a modified metamodel with Context and all the other properties becomes necessary. The changes to the modeling notation are rather serious, but no additional advantages are provided. Therefore this implementation approach is not recommended.

Evaluation Criteria: This pattern is supported by a business process modeling notation, if there is a way to attach something like a Context to activities/groups of activities. Depending on the chosen implementation approach, the support of this pattern might make the compliance with pattern 3 to 8 nonessential for the modeling notation, but of course not for the modeling environment. It also depends on the chosen approach how to annotate model elements using ontologies, as it might even be needless to provide something like a Context element explicitly.

3.3 Organizational Patterns

The Organizational Patterns of this section deal with the structure of technology and human resources within an organization and their connection to the business process.

3.3.1 Pattern 10 (IT-Landscape)

Description: Create a semantically enriched model (use ontologies) of the complete IT-landscape of the organization. Every existent hard- and software combination should be covered.

Example: The IT-landscape of an organization contains many different configurations. There are three different possible server configurations and ten different desktop configurations, that have to be in the model. One of this desktop configuration is: Intel Core2Duo e8500 3,16GHz, 2048 MB RAM, ATI Radeon HD 3870 graphics card, 500GB hard disk, 100MBit network interface card, Windows Vista, Office 2007, JDK 6 Update 4 and Eclipse 3.3.

Motivation: Even though the creation of a model, that covers the complete IT-landscape of an organization, requires extra work at the beginning, it pays off if one considers the benefits that become possible in the future. The configurations created in this model can be used later on to annotate activities with them. Because of the application of ontologies automatic reasoning is possible and opens up new possibilities(for more details take a look at the next pattern: System Annotation). Once modeled, the IT units are available in every business process. It will not be necessary to model parts of the IT-landscape every time they occur in a business process, the existing repository avoids needless double work.

Implementation: There are basically two different ways to create a model of the IT-landscape. Either the model is created completely outside of the business process modeling notation, which means the modeling is done completely within an ontology language, or the modeling notation supports the creation of IT entities that are accessible not only within one business process but in additional ones as well. A transformation into an ontology is nonetheless inevitable.

As there should be a common understanding on the elements a computer system consists of, an ontology seems appropriate that describes the parts like processor, main memory, operating system or office system together with their properties and dependencies on class level. Instances of those classes then constitute the concrete systems. Ideally a standardized ontology is used, that is commonly accepted and applied in industry.

Evaluation Criteria: This pattern is supported by every business process modeling notation, as there is no need to model the IT-landscape necessarily within the notation. However, for better comfort the provision of this capability might be an additional plus.

3.3.2 Pattern 11 (System Annotation)

Description: Annotate activities or groups of activities, that are planned to be performed with the help of a computer system, with exactly that system configuration(s) (part of the previously modeled IT-landscape) that is(are) available to do the job.

Example: Parts of the development process for new products of a company is done by a department where every employee has the same desktop computer to work with. All activities of this department are grouped together and annotated with the corresponding computer system configuration.

Motivation: The annotation of activities with system configurations enables completely new possibilities to verify new or adapted process models that result in the introduction of new software. An automatic check if the existing computer systems are capable of executing the new software may prevent an organization from making wrong decision that could cause enormous costs. Also the search and change of Semantic Web Services before or even during execution of a process can be improved by checking the capability of the systems. It might also help to identify bottlenecks within a business process, e.g. a certain task could last disproportional long only because of the lack of computational power.

Implementation: Again the decision on how to implement this pattern is between the two approaches often described in some of the previous patterns. The first one is to introduce an own model element for the system. This has the advantage that it appears only once in the model and can be reused by connecting it to every activity/group necessary. The second approach is to just use an additional property of the activity to attach a system configuration. Whatever approach is chosen (it also depends on the decision how to implement the IT-Landscape pattern), both have to ensure, that the system configuration is semantically enriched. Ideally the configurations are picked from a repository containing all configurations available at the organization described with an ontology language.

Evaluation Criteria: A business process modeling notation provides support for this pattern, if there is any possibility to annotate activities with semantically described IT systems (configurations).

3.3.3 Pattern 12 (Organizational Structure)

Description: Create a semantically enriched model of the entire organizational structure of the organization.

Example: A company has ten different departments with several sub-departments. The model of the organizational structure illustrates both the hierarchical structure of the business units and departments and the persons working in them with their hierarchical structure as well (e.g. a small department consists of project director Mr. Muller and the four people working for him).

Motivation: The creation of a model, that captures the organizational structure, is important for several reasons. The organization units resulting from this modeling act can be used to annotate activities in a business process model (for the benefits of that see the following pattern). Once modeled, the business entities can be used in many process models, changes of the hierarchical structured have to be made only in this model and the adaption of process models that contain affected business units becomes needless or can be done automatically. Also the modeling of hierarchical structure and job description helps business analysts to discover process steps, where e.g. overqualified employees do simple jobs.

Implementation: Similar to the creation of IT-landscape models, there are basically two different approaches for the implementation of this pattern. The first one is to keep such a model completely out of scope of the business process modeling notation. The complete hierarchical structure is modeled using an ontology language.

The second approach is to model the organizational structure with constructs of the process modeling notation. It is important that this model is not only available for a single business process, but for as many different processes as desired. Of course this model has to be transformed into an ontological version as well in order to allow automatic reasoning on the elements.

The question which approach should be chosen depends highly on the intention of the modeling notation. While some want to support modeling of organizational aspects (especially hierarchical dependencies), others concentrate only on the processes themselves. The implementation approach with the least implications on the notation should be chosen.

Evaluation Criteria: In principle every business process modeling notation supports this pattern, as there is no need to model the organizational structure necessarily within the notation. However, this could be accounted for an additional plus, as it keeps all relevant aspects within one notation.

3.3.4 Pattern 13 (Organizational Unit Annotation)

Description: Semantically annotate activities or groups of activities with the organizational units that are intended to execute them.

Example: The process steps, that have to be executed in order to pay out the salary at the end of the month, are performed by the human resource department. This department consists of four colleagues and every one of them can be involved into the tasks. Therefore, the relevant activities within a business process model are annotated with the entire department as performing organizational unit/role instead of a single person.

Motivation: Several steps of the SBPM lifecycle can be influenced favorably by attaching the executing organizational units to activities. Workflow Management Systems are often used to guide the employees through a business process. It is highly important for such a system to know which user/group of users is responsible and allowed to perform certain tasks. Then it is possible to assign the tasks automatically to the right staff member and thus expedite the process. Via process simulation wrong assignments can be detected before the process is rolled out.

Also the analysis phase benefits from the application of this pattern. It becomes possible to identify process steps where other people than the actually intended ones execute process steps, or high qualified, well-paid specialists do less challenging work. The results of such analysis enable the process planer to optimize the assignation of employees to their tasks. Finally there are additional search possibilities, e.g. the management can find out in how many different processes an employee or department is involved.

Implementation: As described for several other patterns so far, there are again two main approaches on how to implement this pattern. The first one requires an additional property of the activity. This property contains the reference to an instance of an organization unit in the ontology that represents the organizational structure of the company.

Another approach is to model organization units or roles with an explicit model element and connect it with activities. This model element as well represents an instance of organization units, be it single persons, departments or any other thinkable unit. Again the intention of the business process modeling notation is decisive. Either the modeling of organization units plays an important role in the concept of the notation or not. The approach that causes the less changes to the notation should be chosen.

Evaluation Criteria: A business process modeling notation provides support for this pattern, if there is any possibility to annotate activities with semantically described organization units/roles.

3.4 Conditional Patterns

There are two different conditional patterns in this section. The first one examines internal regulations, legal restrictions or something similar that affect the process, while the second

one covers conditions that appear with the cooperation with external business partners.

3.4.1 Pattern 14 (Regulations)

Description: Define and attach rules to activities, groups of activities or entire processes, that express regulations, guidelines or similar requirements.

Example: In a company the decision on investments greater than 5000 € has to be confirmed by at least two persons. This rule holds for every process and therefore every affected process has to be checked whether this rule is met. The processes are annotated with the respective rule.

Motivation: Business processes often have to comply with internal or external regulations based on best practices, company policy or national/international law. As there might be a large amount of such regulations, it is not always easy to detect violations, committed deliberately or not. This could cause financial or penologic consequences and an automatic verification, whether the given rules are followed in process design would be very helpful.

Another problem is that maybe the rules have been kept in mind during process modeling, but the actual execution of the process is different and violates the regulations. This can be detected in the analysis of business processes, when modeled rules and the real process execution are compared. All this shows the importance of the annotation of business processes with machine-readable regulations.

Implementation: As all elements that are relevant for a business process, are semantically described if the previous patterns were applied, it makes sense to choose an implementation approach that utilizes the ontologies developed so far to express the regulations. If OWL was applied as ontology language (as assumed throughout this chapter), SWRL might be a good choice, as it allows to formulate Horn-like rules in both OWL DL and OWL Lite[1]. But any other language that provides the ability to express rules based on ontology constructs is suitable.

Once defined, there has to be a possibility to annotate activities, groups of activities and business processes with those rules. This can be achieved by introducing additional properties for activities, groups and processes, that hold the SWRL expressions.

Evaluation Criteria: This pattern is supported by a business process modeling notation, if there is a possibility to specify regulations for activities, processes and groups (if something like groups exists).

[1] OWL DL and OWL Lite are sublanguages of OWL.

3.4.2 Pattern 15 (Business Partners + Contract Conditions)

Description: Model business partners and their public processes together with contract conditions to enhance the modeling of collaborative business processes and dynamic exchange.

Example: Raw materials needed for production are purchased from different business partners, depending on the actual price offering. Several business processes involve process steps both of the original company and the business partner. This interaction is modeled together with the contract conditions between the two companies (price, liability issues etc.).

Motivation: The importance of collaborative business processes grows more and more, as the interaction with business partners increases in course of a globalized market. As the flexible change of business partners may be the key to be better than competitors, an automatic adaption of business processes is highly desirable. But even the dynamic selection of business partners during runtime becomes possible if business partners are modeled together with the contract conditions. With automatic reasoning exactly that business partner can be selected that matches the requirements of the current instance best.

Additionally it might be useful to store public processes of other companies together with their contract conditions (this is the case if the entire process in a model is performed by the business partner) in order to exchange business partners and automatically adapt the collaborative processes.

Implementation: The business partner represents some kind of organizational unit that is responsible for the execution of several process steps and therefore the implementation is very similar to the one for normal organization units, except the additional contract conditions. Business partners should be attachable to activities, groups and processes.

One solution would be that business partners are an own model element with either a expandable list of properties for the contract conditions, or only one property containing the ontology instance of a contract condition that covers all possible properties. Either way, the contract conditions have to be represented by an ontology so that automatic reasoning becomes possible.

Evaluation Criteria: A business process modeling notation provides full support for this pattern, if there is a way to express contract conditions to organizational units that represent business partners.

4 Pattern-Based Analysis of BPMN and AgilPro

The major drawback of many new technologies is their incompatibility with existing tools and standards. Therefore it seems necessary to combine Semantic Business Process Management with existing business process technology, especially business process modeling notations. Many companies already have models of their business processes, albeit often only for documentation purposes. And the change from a well-known notation to a completely new one is not desirable, which makes it more difficult to convince the management to introduce SBPM technology if this would lead to a replacement of existing resources.

Thus the best idea to pave the way for a broad introduction of Semantic Business Process Management and Modeling seems to be the embedding of the new modeling concepts, that are necessary for the application of SBPM, into existing business process modeling notations. At best there are only minor changes to the notation necessary. By means of the upcoming standard BPMN (see chapter 2.5) and AgilPro (chapter 2.6), a solution developed by the University of Augsburg, Germany, this part of the thesis examines the capability of existing notations to model SBPM related aspects. Both notations are checked if they fulfill the evaluation criteria of the pattern from chapter 3. As the rating often depends on the implementation approach chosen for the according pattern, the evaluation also considers the different options. This may lead to varying results for one pattern and one notation for different implementation approaches.

To allow an easy estimation of the compliance of the modeling notations with the SBPM patterns at first glance, at the end of each section as well as in a summary at the end of this chapter the support of each pattern by a notation is classified into three categories.

These categories are:

+ Full support.

◦ Partial support and/or minor changes to the notation are necessary.

- No support and greater changes to the notation would be necessary.

4.1 Semantic Annotation

The different ideas on how to annotate business process models with semantic information differ in the requirements for a modeling notation. The first two approaches presented in section 3.1.1 do not need a special model element or something like that. Thus both BPMN and AgilPro are in principle capable of semantic annotation. Nonetheless these implementations require some additional work before the process models can be semantically lifted. By all means, the metamodel of the corresponding notation has to be available as ontology. For BPMN some work in this area already has been done, e.g. the SUPER project presented an ontology covering the structural elements of BPMN[1] in WSML. There is no ontology concerning AgilPro available so far, so this work has still to be done. The implementation approach that uses different technological spaces additionally requires some transformations that also still have to be developed. But this does not influence the conclusion, that both BPMN and AgilPro support this pattern if one of these implementation approaches is chosen.

As one could guess, there is no special property (or attribute) of model elements in BPMN or AgilPro, that is meant to establish the link to an ontology (e.g. an URI as value of the property) as the third suggested implementation approach requires it. The question that turns up is, if an existing property of model elements can be misused for semantic annotation or if a completely new property has to be introduced. In the case of BPMN there are two attributes common with all BPMN elements, that are worth looking at (see [OMG07b, p. 35]. One is the *Documentation* that normally contains a textual documentation about the object and is of type String, the other is *Categories*. A modeler may add one or more *Category*, an own concept in BPMN with a *Name* attribute, that has user-defined semantics. The *Documentation* attribute is appropriate if there is no intention by the modeler to make additional textual descriptions, as there is only one *Documentation* per element allowed. *Categories* already have the intention to add some semantic information to the model elements and therefore seem to suit as well, the *Name* attribute of a category could contain e.g. a URI that links to an ontology.

AgilPro offers a quite similar opportunity to add ontology references to model elements. It is possible to add *Comments* to every model element. *Comments* are an own concept with a *text* attribute that can be used to store something like an URI as link to an ontology. As all the existing attributes/properties have a different initial intention, it might also be a good choice to introduce a completely new property, but this would require a change of the notation, which is not always desirable.

Finally, both notations offer an opportunity to use also this implementation approach without changes to the notations metamodel. But if this is done, it should be visible in some

[1] see [BCD+07] Appendix G for the ontology definition, Appendix K for an example

way, which properties are reused for the new purpose.

	BPMN	**AgilPro**
Pattern 1: Semantic Annotation	+	+

Table 4.1: Evaluation Pattern 1: Semantic Annotation

4.2 Grouping

BPMN has a few objects that come into question if one searches for a element that fulfills the requirements of the Grouping pattern. The first candidate might be the *Group* object (cf. [OMG07b, p. 97 et sqq.], that allows the modeler to informally group elements of a diagram. But this solution has a great drawback. A *Group* is an *Artifact* and not a *Flow Object* in BPMN, which means that it has not the same modeling possibilities as an *Activity*. But this is highly necessary, because the intention of this pattern is to allow the same semantic annotation of activity groups with e.g. input/output, conditions and so on as for a single *Task*.

A better solution is to use *Sub-Processes* ([OMG07b, p. 65 et sqq.] to group activities. A *Sub-Process* is an *Activity*, that contains a flow of other *Activities* (*Tasks* and/or *Sub-Processes*). This enables the modeler to group a part of the process flow in a *Sub-Process*, whereas at the same time the *Sub-Process* behaves like an *Activity*. The semantic annotation possibilities are the same as for an *Activity*. BPMN distinguishes between three different types of *Sub-Processes*, an *Embedded*, *Reusable* and *Reference Sub-Process*. The most adequate one seems to be the *Embedded Sub-Process*, as the process flow inside the *Sub-Process* is modeled in the same diagram right where the *Sub-Process* is located. Therefore BPMN supports the Grouping pattern completely with the *Sub-Process* object.

AgilPro as well has an *Embedded Subprocess* object that seems to be appropriate. It contains entire control flows and can be connected with other *Actions* (please remind: Activities as we know them from BPMN are *Actions* in AgilPro, an *Activity* in the AgilPro metamodel is an entire process/model). Therefore it seems that AgilPro fulfills this pattern, but there is a little shortcoming with this concept. *Roles* and *Applications* can not be attached to a *Subprocess* and therefore a *Subprocess* does not exactly behave like a normal *Action*. The reason for this lies in the metamodel of AgilPro.

A *StructuredActivityNode*, the object in the metamodel that becomes an *Embedded Subprocess* later on, inherits only from *ExecutableNode* and not from *Action*, while it is only allowed to connect *Roles*, *Applications* and *Data* with *Actions* (see Figure 4.1 for the crucial part of the metamodel). A change to the metamodel should correct this drawback and after that the

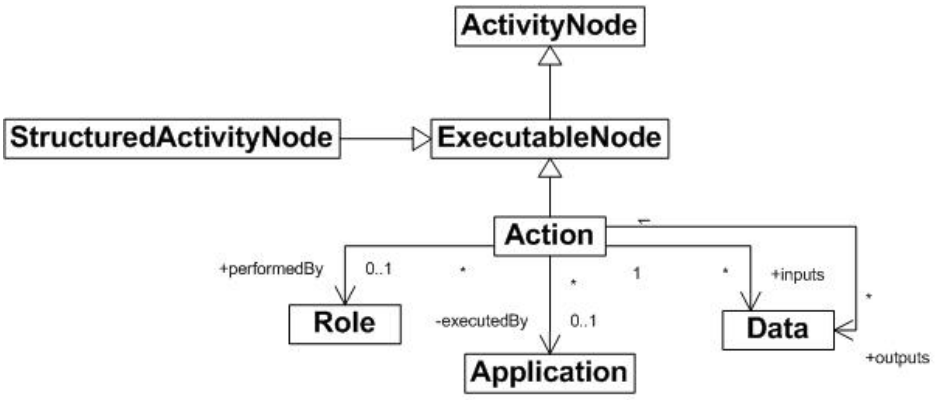


Figure 4.1: Part of the AgilPro metamodel

Grouping pattern is fully supported. But such a change raises new questions, e.g. how to deal with situations when a *ReferencableElemenst* is connected with a *StructuredActivityNode* and *Actions* within it. So the metamodelchange has to be thought out well.

	BPMN	**AgilPro**
Pattern 2: Grouping	+	○

Table 4.2: Evaluation Pattern 2: Grouping

4.3 Input/Output

As the Input and Output patterns have very similar requirements to modeling notations, the analysis of BPMN and AgilPro concerning these two patterns can be combined. The analysis has to distinguish between two general concepts. There is either an explicit representation of inputs and outputs in the model or they only appear in the semantic annotation, which is the case if e.g. an OWL-S profile with corresponding content is applied to annotate an activity. In the last case, simply every modeling notation that allows semantic annotations supports this and the other functional patterns. But sometimes it might come in handy to model something like inputs and outputs directly in the process model and not only in an rather abstract semantic annotation. Additional linkage with an ontology and even embedding it into something like an OWL-S profile should still be possible after all. Therefore both modeling notations are examined if there is a possibility to model inputs and outputs.

In BPMN the entire *Process*, a *Subprocess* and a single *Task*, which are all *Activities*, have the attributes *InputSets* ([OMG07b, p. 282]) and *OutputSets* ([OMG07b, p. 284] that capture possible inputs and outputs. They can be *ArtifactInputs*/*ArtifactOutputs*, which suits our needs in this case, and/or *PropertyInputs/-Outputs*. An *ArtifactInput/-Output* usually

describes some sort of data object and because it refers to an *Artifact*, it can be displayed in the process model and connected with *Activities* via an *Association*. Semantic annotation and combination with concepts like the OWL-S Profile is possible and that is exactly what is needed to support the input and output pattern.

AgilPro has a rather similar approach for inputs and outputs. It is possible to model *Data* elements and connect them with an *Action*. This is done by an arrow, whose direction states if the *Data* is input, output or both. Although the *Data* element in AgilPro has primarily been developed to describe computational data (e.g. *Data* elements can have a *DataType* and *Parameters*), there is no reason why they cannot be used for non-computational input and output. So finally, AgilPro supports the input and output pattern, too.

	BPMN	**AgilPro**
Pattern 3: Input	+	+
Pattern 4: Output	+	+

Table 4.3: Evaluation Pattern 3/4: Input/Output

4.4 Precondition / Postcondition / Effect

The patterns for preconditions, postconditions and effects all have in common, that they use some sort of rule language for their expressions. If an activity is annotated e.g. with an OWL-S Profile, the declaration of pre-/postconditions and effects can be left out of the modeling notation and is the responsibility of the ontology part. Then of course both BPMN and AgilPro support the three patterns, but it is perhaps sometimes also desirable to use concepts of the notations to express them.

	BPMN	**AgilPro**
Pattern 5: Precondition	+	+
Pattern 6: Postcondition	+	+
Pattern 7: Effect	+	+

Table 4.4: Evaluation Pattern 5-7: Specification outside the modeling notation

As mentioned before, in BPMN there are the attributes *InputSets* and *OutputSets* of an Activity that are helpful. Input- and OutputSets can contain *PropertyInputs* and *PropertyOutputs* respectively, that use the concept *Property* [OMG07b, p. 285] for their definition. Properties have two attributes convenient for our purposes: *Type* and *Value*. The *Type* is a String attribute that can provide information about whether the *Property* is a precondition, postcondition or effect. The *Value* attribute is of type *Expression* [OMG07b, p. 281] and

perfect to define the conditions. An *Expression* consists of an *ExpressionBody* attribute for the textual expression and an *ExpressionLanguage* attribute to specify the used language like SWRL or something else. Altogether (Figure 4.2 shows this little part of BPMN; attributes and concepts that are nonrelevant for our purposes have been left out), this allows the modeler to use elements of BPMN to express preconditions, postconditions and effects. Of course they can be combined to something like an OWL-S Profile again.

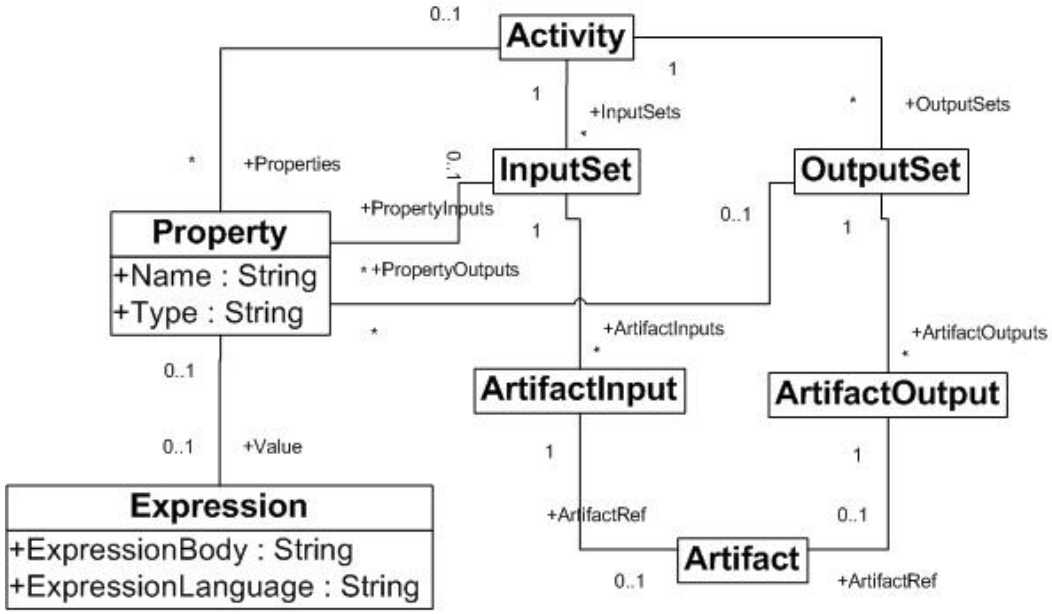


Figure 4.2: View on several BPMN concepts

Another attribute of BPMN Activities that seems to be appropriate at first glance is *IORules*, that also contains an Expression. But while IORules are originally meant to describe the relationship between input and output data, the pre-/postconditions and effects of the Semantic Business Process Modeling patterns do cover additional aspects. A reinterpretation of IORules might do as well, but as there is another possibility to describe the conditions and effects, this solution should not be the first choice.

There is no way in AgilPro to specify preconditions, postconditions and effects with elements of the notation. The idea to use *Guards*, that allow the annotation of *ActivityEdges* with boolean expressions, is not applicable. This would conflict with the original function of a *Guard* as regulator of the control flow. Therefore the only way (if greater changes to the metamodel, that would introduce such elements, are not wanted) is to declare the conditions and effects within an ontology. The Actions have to be annotated with an OWL-S Profile or anything similar.

	BPMN	AgilPro
Pattern 5: Precondition	+	-
Pattern 6: Postcondition	+	-
Pattern 7: Effect	+	-

Table 4.5: Evaluation Pattern 5-7: Specification within the modeling notation

4.5 Nonfunctional Properties

Again there are two different ways to define nonfunctional properties for an activity. If they are only specified within an ontology (annotate an activity with an OWL-S profile for example) of course both modeling notations support the Nonfunctional Properties pattern. But there is also the possibility to model nonfunctional properties with instruments of the notation.

With the *Properties* attribute of *Activities*, BPMN offers a way to define your own properties. There is no restriction which properties you want to model and so it is perfect for the declaration of every nonfunctional property the modeler finds useful. As already stated in the section before, a *Property* has a *Name*, a *Type* and a *Value*, which suffices the needs of this pattern. Of course a connection with an ontology is necessary after all in order to contribute to SBPM benefits.

The only nonfunctional property, that can be modeled directly within AgilPro is the execution time of an *Action*. There is no way to create user-generated properties like in BPMN and so AgilPro supports the Nonfunctional Properties pattern only if an implementation approach is chosen, where an *Action* is annotated with a container of several aspects like the OWL-S profile.

Pattern 8: Nonfunctional Properties	**BPMN**	**AgilPro**
Specification outside the notation (OWL-S Profile)	+	+
Specification within the notation	+	-

Table 4.6: Evaluation Pattern 8: Nonfunctional Properties

4.6 Context

The easiest and probably best way to support the Context pattern is the annotation of a process step with an OWL-S Profile (or its counterpart in other SWS ontologies). If this is done instead of only semantically lifting the identifier of an activity, both notations support the pattern.

But maybe for some reason it could be desirable to have an own property of *Activities* (*Actions* in AgilPro), either for the link to an OWL-S Profile or something similar, or for the explicit aggregation of input, output, pre-/postcondition, effect and nonfunctional properties within the modeling notation. While the *Properties* attribute of an BPMN *Activity* could hold the link to the OWL-S Profile, there is no such possibility in AgilPro. Both modeling notations understandably do not provide a possibility for the aggregation within the notation. So this would require greater changes to the metamodel of both notations.

Pattern 9: Context	**BPMN**	**AgilPro**
Specification outside the notation (OWL-S Profile)	+	+
Annotation within the notation	+	-
Aggregation within the notation	-	-

Table 4.7: Evaluation Pattern 9: Context

4.7 IT-Landscape + System Annotation

Albeit every notation supports the IT-Landscape pattern anyway (an ontology mapping of the IT-landscape is always necessary), it is worthwhile to examine if a model of the IT-landscape can be created within the notation. Computer systems, applications or the like are not part of BPMN (cf. [OMG07b, p. 12]). But the modeler is free to add new types of *Artifacts* to a diagram, and this could be the elements of the IT-landscape. But there are two major drawbacks of this idea. First, there is no possibility to model a hierarchy or dependencies between the *Artifacts*. This has to be done additionally in the ontology describing the IT-landscape. Second, the generated *Artifacts* are only available in the process model they were created in. A further reuse, which is the moving spirit for this pattern, is not possible. So BPMN is not really capable to model the IT-landscape of an organization.

AgilPro has some advantages over BPMN concerning the modeling of IT-landscapes. The bad news are, that it is only possible to model applications, but no technological infrastructure like servers or desktop computers with their components. There is no such way like in BPMN to add something like user-generated objects. But the reuse factor of this pattern is supported much better by AgilPro. The *Applications*, that have been created in one workflow file, are accessible in all process models in this file via references. In addition, the *Application* can be exported and used by other workflow files. So the idea to model available applications once and reuse them in as many process models as one likes is highly supported by AgilPro. With the introduction of additional classes for computer systems etc. AgilPro would fully support this pattern.

The degree of support for the System Annotation pattern of course depends on the possi-

Pattern 10: IT-Landscape	**BPMN**	**AgilPro**
Specification outside the notation	+	+
Specification within the notation	-	○

Table 4.8: Evaluation Pattern 10: IT-Landscape

bilities to model the IT-landscape. Either an own element for the system can be used or just an additional property of the activity provides the link to the ontology. As described before, in BPMN new *Artifacts* can represent IT systems and be connected with *Activities*. So this approach fulfills the requirements of the System Annotation pattern. But also the *Properties* attribute of an *Activity* makes up a possible solution. A common *Type* of such *Properties*, like e.g. 'System', distinguishes this *Property* from others and it now is able to establish a link to the IT-landscape ontology.

Besides *Applications*, AgilPro offers no way to annotate *Actions* directly with IT systems or with a new property, as there is no possibility for the modeler to define new properties. Now either the *Application* concept has to be extended so that it includes systems as well, or a new property has to be introduced in the metamodel. At the current status, AgilPro does not fully support the System Annotation pattern.

	BPMN	**AgilPro**
Pattern 11: System Annotation	+	○

Table 4.9: Evaluation Pattern 11: System Annotation

4.8 Organizational Structure + Org. Unit Annotation

Similar to the IT-landscape, a modeling of the organizational structure is not supported by BPMN. Again the only possibility would be to invent new types of *Artifacts* with the same drawbacks concerning reuse and hierarchy/dependencies as shown in the previous section. The better way seems to use only an ontology model of the organizational structure completely outside the modeling notation.

An important concept in AgilPro are *Roles*, that can describe organizational units from a single actor up to a whole department or similar. It is even possible to define hierarchy relations. Regrettably, it is not possible to represent this graphically at the moment. Similar to *Applications*, once defined *Roles* can be referenced in every process model and exported. The reuse factor of the Organizational Structure pattern is well served.

As the organizational structure normally should be represented by an ontology (and not by new types of *Artifacts*, even though such *Artifacts* can be connected with *Activities* as well)

Pattern 11: Organizational Structure	BPMN	AgilPro
Specification outside the notation	+	+
Specification within the notation	-	+

Table 4.10: Evaluation Pattern 12: Organizational Structure

there has to be a way to annotate a BPMN *Activity* with an organizational unit. An *Activity* has the *Performers* attribute that suits perfectly. It is meant to add one or more organizational units that are responsible for an *Activity*. As *Performers* is only a String attribute, there is no problem to use it as link to the ontology that covers the whole organizational structure. But also the concept of *Swimlanes* and *Pools* provides a graphical method to partition activities with regard to the participants. These concepts again may link to the organizational structure ontology and therefore are a way to support the Organizational Unit Annotation pattern.

In AgilPro organizational units have their own model element, namely *Roles*, and of course they can be attached to *Actions*. The link to the organizational structure ontology has to be made by the *Role* objects. The other approach that uses an additional property to connect an organizational unit with an *Action* is not possible, as no such property exists and the definition of new properties is not allowed to the user. But this is not necessary anyway, because there is no reason why the existing *Role* concept should not be used.

	BPMN	AgilPro
Pattern 13: Organizational Unit Annotation	+	+

Table 4.11: Evaluation Pattern 13: Organizational Unit Annotation

4.9 Regulations

In order to support the Regulations pattern, a modeling notation has to provide a possibility to specify regulations (typically expressions in a rule language) for activities, groups of activities and the whole process model.

In BPMN again the *Properties* attribute of *Activities* becomes useful, as it allows the modeler to add a new property of a certain type (here the type could be e.g. 'Regulation') and with a *Value* that contains an *Expression* in a freely selectable language. As already seen before, both *Tasks* and *Sub-Processes* have this attribute. Two of the three relevant objects are covered. The business process diagram itself does not have this attribute, but the *Process* concept, a "graph of flow objects, which are a set of other activities and the controls that sequence them" [OMG07b, p. 32], does, as it is an *Activity*, too. And because *Processes* may be defined at any level, they can cover the entire control flow of a model. Thus, BPMN supports the Regulations pattern.

In AgilPro neither *Actions* and *Subprocesses* nor an *Activity* (the entire model) have a property that could be used to add expressions for regulations. Such an additional property has to be added in the metamodel if the Regulation pattern is ought to be supported.

	BPMN	**AgilPro**
Pattern 14: Regulations	+	-

Table 4.12: Evaluation Pattern 14: Regulations

4.10 Business Partners + Contract Conditions

The Business Partners and Contract Conditions pattern makes some heavy demands if a modeling notation wants to support it. There has to be a way to define organizational units that represent business partners plus the contract conditions between the company and its business partners.

In BPMN *Pools* together with *Message Flows* enable the modeling of external business partners and the necessary interaction. But there is no way to integrate contract conditions. The *ParticipantRef* attribute of a *Pool* defines the responsible business unit, which is in this case the external partner. But the *Participant* concept offers no possibility to define something like contract conditions. An additional attribute, that contains an *Expression* would be required to support the Business Partners + Contract Condition pattern.

With *Roles*, organizational units have their own model element in AgilPro. Because business partners are only a specific type of organizational unit, they can be defined as well. But similar to BPMN there is no way to add an expression for contract conditions to a *Role*. So AgilPro does not support this pattern and an additional property for the conditions becomes inevitable.

	BPMN	**AgilPro**
Pattern 15: Business Partners + Contract Conditions	-	-

Table 4.13: Evaluation Pattern 15: Business Partners + Contract Conditions

4.11 Analysis Summary

Despite the fact, that Semantic Business Process Management is a pretty new research area and was not a priority in the development process of current business process modeling notations, the analysis in this chapter showed that they are already prepared for it. Of course sometimes it becomes necessary to reinterpret elements of the notation or leave the semantic description mostly outside the notation and use ontologies to define the details, but in general the two rather new notations BPMN and AgilPro provide support for almost all the patterns for Semantic Business Process Modeling. This might be different for other modeling notations, especially the ones like Petri Nets or UML Activity Diagrams, that were originally not intended for the representation of business processes. A further analysis of other notations concerning the patterns could be very revealing, but is out of the scope of this thesis.

A final overview of how BPMN and AgilPro support the patterns for semantic business process modeling shows table 4.14. As there were different results depending of the chosen implementation approach of the patterns, this table assumes, that the approach is chosen that allows the modeling notation to cover as much of the patterns as possible. So the table shows the best possible support of the patterns by each notation (even if this means the definition of certain aspects has to be done completely outside the notation and the notation actually is not involved at all) and as can be seen easily, they both perform pretty well. BPMN does slightly better than AgilPro, but they are both a very good starting point for semantically enriched business process models and all the future applications of Semantic Business Process Management. Nevertheless, there is always space for improvements, especially concerning the patterns where the best solution at the moment seems to be doing everything completely outside the notation and only within an ontology. Although BPMN does a little bit better than AgilPro when it comes to support the patterns at least at a very basic level, AgilPro also has some advantages over BPMN in some aspects as can be seen in the more detailed analysis of each pattern.

	BPMN	AgilPro
Pattern 1: Semantic Annotation	+	+
Pattern 2: Grouping	+	○
Pattern 3: Input	+	+
Pattern 4: Output	+	+
Pattern 5: Precondition	+	+
Pattern 6: Postcondition	+	+
Pattern 7: Effect	+	+
Pattern 8: Nonfunctional Properties	+	+
Pattern 9: Context	+	+
Pattern 10: IT-Landscape	+	+
Pattern 11: System Annotation	+	○
Pattern 12: Organizational Structure	+	+
Pattern 13: Organizational Unit Annotation	+	+
Pattern 14: Regulations	+	-
Pattern 15: Business Partners + Contract Conditions	-	-

Table 4.14: Evaluation of the SBPM perspective

5 Conclusion and Prospects

Semantic Business Process Management opens up new possibilities for automatic processing in business process modeling, implementation, execution and analysis. These possibilities, the underlying technologies and current SBPM research projects have been presented at the beginning of this thesis.

If one takes a closer look at SBPM, it becomes quite obvious, that semantically enriched business process models are the essential element for all the possible applications. But unfortunately there is no common understanding on the content of semantic business process models so far. Another unsolved question is, how the new information can be integrated into models, that were created with existing modeling notations. The patterns for Semantic Business Process Modeling of this thesis provide answers for both problems. On the one hand the patterns are a summary of all the additional semantic information, that is necessary for the different SBPM applications, while on the other hand possible solutions for the integration of each pattern into process modeling notations are pointed out.

One of the major possible drawbacks of SBPM is the missing acceptance of it in the economy. This threat can be reduced if existing technologies, that are already in use and well accepted, can be easily extended for the new technology. In the case of Semantic Business Process Modeling this means, that existing modeling notations should support the new requirements, that arise with SBPM and are represented by the patterns, with as less changes to the notation as possible. The pattern-based analysis of BPMN and AgilPro in this thesis examines how these two current process modeling notations cope with the SBPM requirements. The pleasant result of this analysis is, that both notations support most of the patterns in some way. The annotation of process models with all the necessary semantic information is possible without greater changes to the metamodel. Nonetheless, there is still room for improvement and further development to facilitate Semantic Business Process Modeling.

Because Semantic Business Process Management and Modeling is such a new research area, much work has to be done until SBPM can be introduced in organizations. The technologies of the Semantic Web like e.g. reasoners have still to become faster and more efficient. Some other important things like semantic business process repositories are not available at all. There is also no support for Semantic Business Process Modeling in current established

modeling environments, but the easy annotation of business processes with semantic information by using well-known tools is a basic prerequisite for the success of SBPM. However, with WSMO-Studio[1] a first new Semantic Business Process Modeling environment is available. If SBPM implies too much additional intricate work, it will be much more difficult to convince organizations of the benefits. Additionally, some sort of SBPM toolsuite, that covers all phases of the SBPM lifecycle, has to be developed. The best thing would be a further development of existing BPM environments, as there should be a higher acceptance then.

This all sounds as if there is a huge amount of work for the next years and it will likely take yet another couple of years until SBPM has reached a status, that allows organizations to use this new technology. But the great benefits of SBPM seem to be worth all the hard work and it might even be inescapable to use new technologies, that allow more automatic processing in business process affairs, in order to survive on fast developing global markets.

Semantic Business Process Modeling is one of the parts of SBPM, that is closest to realization and the sooner companies can create semantically enriched process models the better. Once process models with semantic information exist, more and more SBPM applications can be introduced one after another without adapting the models every time. Semantic business process models stand at the beginning of SBPM and possibly a new way of Business Process Management.

[1] `http://www.wsmostudio.org`, as at 2008-03-25

Bibliography

[ABB+07] Asma Alazeib, Markus Bauer, Athanasios Bouras, Maja Dyczkowska, Andreas Friesen, Panagiotis Gouvas, Grzegorz Jurkowski, Piotr Jurkowski, Kostas Kalaboukas, Dimitrios Kourtesis, Gergely Lukacsy, Peter Martinek, Grigoris Mentzas, Albina Pace, Stelios Pantelopoulos, Iraklis Paraskakis, and Bianca Szalai. Fusion approach. Technical report, FUSION project, 2007. `http://www.fusionweb.org/Fusion/download/StW/D12+FUSION+Approach+M18+Update+Final+Version+.pdf`, as at 2007-11-14.

[AFH+06] D. Apostolou, D. Feldkamp, C. Halaris, K. Hinkelmann, B. Magoutas, X. Papadomichelaki, C. Prackwieser, F. Probst, K.U. Schmidt, B. Stoiljkovic, V. Stoiljkovic, L. Stojanovic, N. Stojanovic, S.M. Thomas, B. Thönssen, W. Utz, and R. Woitsch. Framework for self-adaptive e-government. Technical report, FIT project, 2006. `http://www.fit-project.org/Documents/D2.pdf`, as at 2007-11-15.

[BCD+07] Roxana Belecheanu, Liliana Cabral, John Domingue, Walid Gaaloul, Martin Hepp, Agata Filipowska, Monika Kaczmarek, Tomasz Kaczmarke, Jörg Nitzsche, Barry Norton, Carlos Pedrinaci, Dumitru Roman, Michael Stollberg, and Sebastian Stein. Business process ontology framework. Technical report, Integrated Project SUPER, 2007. `http://www.ip-super.org/res/Deliverables/M12/D1.1.pdf`, as at 2007-11-14.

[BHHL07] Bernhard Bauer, Matthias Henneberger, Bernd Heinrich, and Florian Lautenbacher. Semantic-based planning of process models, 2007.

[BKKO06] Stefanie Betz, Stefan Klink, Agnes Koschmider, and Andreas Oberweis. Automatic user support for business process modeling. In *Proceedings of the Workshop on Semantics for Business Process Management, Workshop at the 3rd European Semantic Web Conference, 11 June 2006, Budva*, pages 1 – 12, 2006.

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American, http://www.sciam.com/print_version.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21, as at 2007-11-02*, May 2001.

[BLPR07] Bernhard Bauer, Florian Lautenbacher, Günther Palfinger, and Stephan Roser. "agilpro": Modellierung, simulation und ausführung agiler prozesse. *OBJEKTspektrum*, (1), 2007.

[BPM04] BPMI. *Business Process Modeling Notation (BPMN) Version 1.0 - May 3, 2004*. 2004. `http://www.bpmn.org/Documents/BPMN%20V1-0%20May%203%202004.pdf`, as at 2007-11-27.

[CLB07] Oscar Corcho, Silvestre Losada, and Richard Benjamins. Mediation - bridging between heterogeneous web services systems. In Rudi Studer, Stephan Grimm, and Andreas Abecker, editors, *Semantic Web Services - Concepts, Technologies and Applications*, pages 287 – 308. Springer Berlin Heidelberg, 2007.

[CYH07] Emilia Cimpian, Zhixian Yan, and Ta'id Holmes. Business process modeling ontology (bpmo) version 1. Technical report, SemBiz project, 2007. `http://www.sembiz.org/attach/D1.2.pdf`, as at 2007-11-15.

[DCC+07] Christian Drumm, Lilliana Cabral, Emilia Cimpian, John Domingue, Adrian Mocan, and Jussi Vanhatalo. Business process mediation conceptual framework. Technical report, Integrated Project SUPER, 2007. `http://www.ip-super.org/res/Deliverables/M12/D4.1.pdf`, as at 2007-11-14.

[DHH+07] Schahram Dustdar, Jörg Hoffmann, Ta'id Holmes, Adina Sirbu, Huy Tran, and Uwe Zdun. Semantic querying, discovery, and composition framework. Technical report, SemBiz project, 2007. `http://www.sembiz.org/attach/D2.2.pdf`, as at 2007-11-15.

[DLN06] Christian Drumm, Jens Lemcke, and Kioumars Namiri. Integrating semantic web services and business process management: A real use case. In *Proceedings of the Workshop on Semantics for Business Process Management, Workshop at the 3rd European Semantic Web Conference, 11 June 2006, Budva*, pages 13 – 27, 2006.

[FHK+07] Agata Filipowska, Armin Haller, Monika Kaczmarek, Tammo van Lessen, Jörg Nitzsche, and Barry Norton. Process ontology language and operational semantics for semantic business processes. Technical report, Integrated Project SUPER, 2007. `http://www.ip-super.org/res/Deliverables/M12/D1.3.pdf`, as at 2007-11-14.

[FHT07] Daniela Feldkamp, Knut Hinkelmann, and Barbara Thönssen. Semantic modelling of adaptable processes. Technical report, FIT project, 2007. `http://www.fit-project.org/Documents/D8.pdf`, as at 2007-11-15.

[FKG+06] Andreas Friesen, Péter Krauth, Panagiotis Gouvas, József Kerekes, and Athanasios Bouras. State of the art. Technical report, FUSION project, 2006. `http://www.fusionweb.org/Fusion/download/StW/D11+State+of+the+art+update+final.pdf`, as at 2007-11-14.

[FLA+07] Andreas Friesen, Jens Lemcke, Asma Alazeib, Andras Balogh, Albina Pace, and Markus Bauer. Fusion process designer specifications. Technical report, FUSION project, 2007. `http://www.fusionweb.org/Fusion/download/StW/D21c+FUSION+Process+Designers+Specification.pdf`, as at 2007-11-14.

[FSM+07] Paola Fantini, Alberto Savoldelli, Micaela Milanesi, Giulio Carizzoni, Jana Koehler, Sebastian Stein, Ralf Angeli, Martin Hepp, Dumitru Roman, Christian Brelage, and Matthias Born. Semantic business process life cycle. Technical report, Integrated Project SUPER, 2007. `http://www.ip-super.org/res/Deliverables/M12/D2.2.pdf`, as at 2007-11-14.

[FUS] FUSION. Fusion objectives. `http://www.fusionweb.org/Fusion/objectives/FUSION_objectives.asp`, as at 2007-11-13.

[GDDD04] Dragan Gasevic, Dragan Djuric, Vladan Devedzic, and Violeta Damjanovic. Approaching owl and mda through technological spaces. Workshop in Software Model Engineering (WiSME), Lisbon, Portugal, October 11th, 2004.

[Gha98] M. Ghallab. Pddl-the planning domain definition language v. 2. Technical Report, report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, 1998.

[GHA07] Stephan Grimm, Pascal Hitzler, and Andreas Abecker. Knowledge representation and ontologies. In Rudi Studer, Stephan Grimm, and Andreas Abecker, editors, *Semantic Web Services - Concepts, Technologies and Applications*, pages 51 – 105. Springer Berlin Heidelberg, 2007.

[GHJV95] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, Massachusetts, 1995.

[Gri07] Stephan Grimm. Discovery - identifying relevant services. In Rudi Studer, Stephan Grimm, and Andreas Abecker, editors, *Semantic Web Services - Concepts, Technologies and Applications*, pages 211 – 244. Springer Berlin Heidelberg, 2007.

[Gru93] Thomas Gruber. Toward principles for the design of ontologies used for knowledge sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands, 1993. Kluwer Academic Publishers.

[HK07] Laurent Henocque and Mathias Kleiner. Composition - combining web services functionality in composite orchestrations. In Rudi Studer, Stephan Grimm, and Andreas Abecker, editors, *Semantic Web Services - Concepts, Technologies and Applications*, pages 245 – 286. Springer Berlin Heidelberg, 2007.

[HLD+05] Martin Hepp, Frank Leymann, John Domingue, Alexander Wahler, and Dieter Fensel. Semantic business process management: A vision towards using semantic web services for business process management. In *ICEBE 2005, IEEE International Conference on e-Business Engineering, Beijing, China, 18-21 October 2005*, pages 535–540. IEEE Computer Society Press, 2005.

[HR07] Martin Hepp and Dumitru Roman. An ontology framework for semantic business process management. In Andreas Oberweis, Christof Weinhardt, Henner Gimpel, Agnes Koschmider, Victor Pankratius, and Björn Schnizler, editors, *eOrganisation : Service-, Prozess-, Market-Engineering : 8. Internationale Tagung Wirtschaftsinformatik - Band 1*, pages 423–440. Universitätsverlag Karlsruhe, 2007. proceedings of the 8th international conference Wirtschaftsinformatik 2007, February 28 - March 2, Karlsruhe.

[IIN+07] IBIS, IDS, NUIG, NIWA, ONTO, and USTUTT. Execution engine design and architecture. Technical report, Integrated Project SUPER, 2007. `http://www.ip-super.org/res/Deliverables/M12/D6.1.pdf`, as at 2007-11-14.

[KIF98] KIF. Knowledge interchange format: Draft proposed american national standard (dpans). Technical Report 2/98-004, ANS, 1998.

[KKM+07] Monika Kaczmarek, Mihail Konstantinov, Zhilei Ma, Karol Wieloch, and Pawel Zebrowski. Business process library design and first prototype. Technical report, Integrated Project SUPER, 2007. `http://www.ip-super.org/res/Deliverables/M12/D3.1.pdf`, as at 2007-11-14.

[LLP+07] Holger Lausen, Rubén Lara, Axel Polleres, Jos de Bruijn, and Dumitru Roman. Description - semantic annotation for webservices. In Rudi Studer, Stephan Grimm, and Andreas Abecker, editors, *Semantic Web Services - Concepts, Technologies and Applications*, pages 179 – 209. Springer Berlin Heidelberg, 2007.

[MGB07] Peter Martinek, Panagiotis Gouvas, and Thanasis Bouras. Fusion semantic profiler specifications. Technical report, FUSION project, 2007. `http://www.fusionweb.org/Fusion/download/StW/D21b+Specifications+of+the+Semantic+Profiler.pdf`, as at 2007-11-14.

[MM03] Sheila McIlraith and David Martin. Bringing semantics to web servies. *IEEE Intelligent Systems*, (18(1)):90 – 93, 2003.

[MPC+07] Babis Magoutas, Xenia Papadomichelaki, Christos Chalaris, Maria Legal, Efi Vladimirou, and Bratislav Stoiljkovic. Quality of e-government services (qegs) model and ontology (initial version). Technical report, FIT project, 2007. `http://www.fit-project.org/Documents/D7.pdf`, as at 2007-11-15.

[NS06] Kioumars Namiri and Nenad Stojanovic. Towards business level verification of cross-organizational business processes. In *Proceedings of the Workshop on Semantics for Business Process Management, Workshop at the 3rd European Semantic Web Conference, 11 June 2006, Budva*, pages 101 – 112, 2006.

[NS07] Kioumars Namiri and Nenad Stojanovic. A model-driven approach for internal controls compliance in business processes. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007), Innsbruck, Austria, June 7, 2007*, pages 40 – 43, 2007.

[OMG07a] OMG. *Business Process Model and Notation (BPMN) 2.0 Request For Proposal*. June 2007. `http://www.bpmn.org/Documents/BPMN%202-0%20RFP%2007-06-05.pdf`, as at 2007-11-28.

[OMG07b] OMG. *Business Process Modeling Notation (BPMN) Specification Version 1.1 – DRAFT*. June 2007. `http://www.omg.org/cgi-bin/apps/doc?dtc/07-06-03.pdf`, as at 2007-11-27.

[OR03] Martin Owen and Jog Raj. *BPMN and Business Process Management - Introduction to the New Business Process Modeling Standard*. Popkin Software, 2003. `http://www.bpmn.org/Documents/6AD5D16960.BPMN_and_BPM.pdf`, as at 2007-11-28.

[Pre07] Chris Preist. Goals and vision - combining web services with semantic web technology. In Rudi Studer, Stephan Grimm, and Andreas Abecker, editors,

Semantic Web Services - Concepts, Technologies and Applications, pages 159 – 178. Springer Berlin Heidelberg, 2007.

[QW04] Martin Quantz and Thorsten Wichmann. Report on current usage of web services and semantic web. Technical report, DIP - Data, Information and Process Integration with Semantic Web Services, June 2004. `http://dip.semanticweb.org/documents/D12.1ReportoncurrentusageofWebServicesandSemanticWeb.pdf`, as at: 2007-10-29.

[RBV+06] Ray Richardson, Aidan Boran, Tomas Vitvar, Paavo Kotinurmi, David Lewis, John Keeney, and Declan O'Sullivan. Adaptive technologies to address operational complexity in highly configurable value chains. In *Proceedings of the Workshop on Semantics for Business Process Management, Workshop at the 3rd European Semantic Web Conference, 11 June 2006, Budva*, pages 113 – 124, 2006.

[RtHEvdA04] Nick Russell, Arthur H.M. ter Hofstede, David Edmond, and Wil M.P. van der Aalst. Workflow data patterns. QUT Technical report, FIT-TR-2004-01, Queensland University of Technology, Brisbane, 2004.

[RtHvdAM06] Nick Russell, Arthur H.M. ter Hofstede, Wil M.P. van der Aalst, and Nataliya Mulyar. Workflow control-flow patterns: A revised view. BPM Center Report BPM-06-22 , BPMcenter.org, 2006.

[SBO07] Rainer Schmidt, Christian Bartsch, and Roy Oberhauser. Ontology-based representation of compliance requirements for service processes. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007), Innsbruck, Austria, June 7, 2007*, pages 28 – 39, 2007.

[Sem] SemBiz. Sembiz project. `http://www.sembiz.org/index.html`, as at 2007-11-15.

[SF03] Howard Smith and Peter Fingar. *Business Process Management - The Third Wave*. Meghan-Kiffer Press, Tampa, Florida, USA, 1st edition, 2003.

[SPKP07] Kostas Sidiropoulos, Stelios Pantelopoulos, Dimitrios Kourtesis, and Iraklis Paraskakis. Specification of the integration mechanism. Technical report, FUSION project, 2007. `http://www.fusionweb.org/Fusion/download/StW/D31+Specifications+of+the+Integration+Mechanism.pdf`, as at 2007-11-14.

[SUPa] SUPER. Integrated project super. `http://www.ip-super.org/content/view/27/44/`, as at 2007-11-14.

[SUPb] SUPER. Scope and content of the objectives. `http://www.ip-super.org/content/view/25/42/`, as at 2007-11-14.

[TF07] Oliver Thomas and Michael Fellmann. Semantic epc: Enhancing process modeling using ontology languages. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007), Innsbruck, Austria, June 7, 2007*, pages 64 – 75, 2007.

[vdAtHW03] Wil M.P. van der Aalst, Arthur H.M. ter Hofstede, and Mathias Weske. Business process management: A survey. In *Business Process Management: International Conference, BPM 2003, Eindhoven, The Netherlands, June 26-27, 2003. Proceedings*, pages 1 – 12. Springer Berlin Heidelberg, 2003.

[WF07] Christian Werner and Stefan Fischer. Architecture and standardisation of web services. In Rudi Studer, Stephan Grimm, and Andreas Abecker, editors, *Semantic Web Services - Concepts, Technologies and Applications*, pages 25 – 48. Springer Berlin Heidelberg, 2007.

[Whi04] Stephen A. White. *Introduction to BPMN*. IBM Corporation, 2004. `http://www.bpmn.org/Documents/Introduction%20to%20BPMN.pdf`, as at 2007-11-28.

[WMF+07] Branimir Wetzstein, Zhilei Ma, Agata Filipowska, Monika Kaczmarek, Sami Bhiri, Silvestre Losada, Jose-Manuel Lopez-Cobo, and Laurent Cicurel. Semantic business process management: A lifecycle based requirements analysis. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007), Innsbruck, Austria, June 7, 2007*, pages 1 – 11, 2007.

List of Figures

List of Tables

Listings

5.1 List of Abbreviations

B2B Business to Business

BPD Business Process Diagram

BPDM Business Process Definition Metamodel

BPEL Business Process Execution Language

BPEL4WS Business Process Execution Language for Web Services

BPM Business Process Management

BPMI Business Process Management Initiative

BPMN Business Process Modeling Notation

BPMO Business Process Modeling Ontology

CAD Computer Aided Design

DFG Deutsche Forschungsgesellschaft - German Research Foundation

DL Description Logic

DRS Declarative RDF System

EAI Enterprise Application Integration

EPC Event-driven Process Chain

ERP	Enterprise Resource Planning
FIT	Fostering self-adaptive e-government service improvement using semantic technologies
GoF	Gang of Four
HTTP	Hypertext Transfer Protocol
IDL	Interface Definition Language
IT	Information Technology
JWT	Java Workflow Tooling
KIF	Knowledge Interchange Format
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
OWL	Web Ontology Language
PDDL	Planning Domain Definition Language
QoS	Quality of Service
RDF	Resource Description Framework
RDFS	RDF Schema
SAWSDL	Semantic Annotations for WSDL
SBPM	Semantic Business Process Management
SEBIS	Semantics in Business Information Systems
SEMPA	Semantic-based Planning of Activities
SME	Small and Medium-sized Enterprise
SOA	Service-Oriented Architecture
SOAP	before version 1.2: Simple Object Access Protocol
SUPER	Semantics Utilised for Process Management within and between Enterprises
SWRL	Semantic Web Rule Language
SWS	Semantic Web Service
SWSF	Semantic Web Services Framework
SWSL	Semantic Web Service Language
UDDI	Universal Description, Discovery and Integration
UML	Unified Modeling Language
URI	Unique Resource Identifier
W3C	World Wide Web Consortium

WSDL Web Services Description Language

WSML Web Service Modeling Language

WSMO Web Service Modeling Ontology

XML Extensible Markup Language