Lightweight Process Modelling

Methodology, Language, and Tools

DISSERTATION
of the University of St. Gallen,
School of Management,
Economics, Law, Social Sciences
and International Affairs
to obtain the title of
Doctor Oeconomicae

submitted by

Stephan Florian Schnabel

from

Germany

Approved on the application of

Prof. Dr. Beat Schmid

and

Prof. Dr. Reinhard Jung

Dissertation no. 4044

The University of St. Gallen, School of Management, Economics, Law, Social Sciences and International Affairs hereby consents to the printing of the present dissertation, without hereby epressing any opinion on the views herein expressed.

St. Gallen, May 21, 2012

The President:

Prof. Dr. Thomas Bieger

Acknowledgements

Many people have strongly supported my efforts in writing this dissertation thesis. I highly appreciate their fruitful discussions, inspiration, and inputs.

Firstly, I would like to thank my PhD supervisors, Prof. Dr. Beat F. Schmid and Prof. Dr. Reinhard Jung. They provided the basis to conduct my research at the University of St. Gallen (HSG) in an applied manner. The inspirational discussions with them have guided my research and this thesis.

I'd like to thank the team of the Institute of Media and Communication Management (MCM) at the University of St. Gallen. The MCM provided an atmosphere where I could consequently follow my research ideas. I appreciate the feedback that influenced my dissertation thesis.

Further, I would like to thank the team of the SAP Research CEC St. Gallen. In particular, I would like to thank Dr. Jürgen Vogel and Dr. Oliver Christ who guided me in applying the research work to industrial scenarios and the research project SOA4All. I am as well thankful to the team of the research project SOA4All who created a very constructive and collaborative research environment for the core topic of this thesis.

I am highly grateful to my family: to my girlfriend Melina Wittmann for her grate energy and continuous support, to my parents Rainer and Ingrid Schnabel, and to my brother Marc Schnabel.

Florian Schnabel, 21.05.2012

Table of Contents

List of Figures

List of Tables

Acronyms

AI	Artificial Intelligence
API	Application Programming Interface
BPEL	Business Process Execution Language
BPD	Business Process Diagram
BPDM	Business Process Definition Metamodel
BPM	Business Process Modelling
BPMN	Business Process Modelling Notation
BPMS	Business Process Management Systems
EAI	Enterprise Application Integration
EPC	Event-Driven Process Chain
EUC	End-User Computing
EUD	End-User Development
GAMP	Generally Accepted Modelling Principles
IS	Information System
IT	Information Technology
LPM	Lightweight Process Modelling
LPML	Lightweight Process Modelling Language
LPMS	Lightweight Process Modelling Solution
PS	Public Sector
RDF	Resource Description Framework
REST	Representational State Transfer
RIA	Rich Internet Application
SEE	Semantic Execution Environment
SWS	Semantic Web Services
WfMS	Workflow Management System
WSDL	Web Service Definition Language
WSMO	Web Service Modelling Ontology
WS-BPEL	Web Service Business Process Execution Language
YAWL	Yet Another Workflow Language

Management Summary

Currently, modelling and executing processes requires a high level of expertise in business and IT rendering existing process modelling languages and tools unsuitable for the non-experienced business user. However, the business users build the majority of information workers and deciders. By non-experienced business users, users are referred to that are 'not casual, novice, or naive', but have got strong domain-specific business skills. Concerning IT, they have computational needs, but limited IT knowledge and no interest in becoming an IT professional.

This thesis addresses the need for a process modelling solution that the business user might use in a lightweight way. In this sense, the term lightweight concerns the user interaction and means easy to understand in the context of the modelling language and easy to deploy, implement, and execute in a tooling context. However, in order to realize a lightweight user access, sophisticated backend solutions are needed.

The objective of this thesis is to define the design of a framework for Lightweight Process Modelling (LPM) targeting the business user. This comprises three major components. Firstly, a design principles framework, structured by the LPM metamodel, for artefacts supporting the business user in modelling and executing processes is defined. The second component is a process modelling language defining syntax and semantics. The language has two representation layers. One for abstract business processes for documentation, communication, and collaboration purposes for business users. The second layer is a canonical representation format for process execution. The two abstraction layers are based on the LPM metamodel. The third component comprises a technical architecture and tools that support the business user in modelling, deploying, and executing the process models. This comprises prototype specifications for both front- and back-end tools, such as a process editor and an execution engine. Furthermore, a design process is built defining the interactions of the tools in order to enhance abstract processes by execution details.

By describing LPM and creating the artefact of a Lightweight Process Modelling Solution (LPMS), this thesis follows the design science research methodology.

Zusammenfassung

Die gleichzeitige Modellierung und Ausführung von Prozessen erfordert momentan ein hohes Mass an Fach- und IT-Wissen. Existierende Sprachen und Tools sind damit für unerfahrene Fachanwender wenig geeignet. Fachanwender bilden jedoch die Mehrheit an Informationsarbeitern und –entscheidern in Unternehmen. Unerfahrene Fachanwender sind in diesem Zusammenhang aber keine Gelegenheitsanwender, neue oder naive Anwender. Sie haben ein grosses, domänenspezifisches Fachwissen. Und spezifische Bedürfnisse an die IT, jedoch nur beschränktes Wissen und kein Interesse daran, ein IT-Experte zu werden.

Diese Dissertation adressiert das Bedürfnis für eine Prozessmodellierungslösung, die der Fachanwender leichtgewichtig nutzen kann. Der Begriff leichtgewichtig bezieht sich hierbei auf die Nutzerinteraktion und bedeutet leicht zu verstehen in bezug auf eine Modellierungssprache und einfach zu implementieren, einzusetzen und auszuführen in bezug auf Werkzeuge. Um jedoch einen leichtgewichtigen Nutzerzugang zu realisieren, sind komplexe Lösungen im Hintergrund erforderlich.

Das Ziel der Dissertation ist, das Design eines Rahmenwerks für leichtgewichtige Prozessmodellierung (LPM) für Fachanwender zu definieren. Dies umfasst drei Hauptkomponenten. Zum einen ein Rahmenwerk für Designprinzipien, das durch das LPM-Metamodell strukturiert wird. Dieses Rahmenwerk wird für Artefakte definiert, die den Fachanwender bei der Modellierung und Ausführung von Prozessen unterstützen. Die zweite Komponente stellt eine Prozessmodellierungssprache mit definierter Syntax und Semantik dar. Die Sprache enthält zwei Repräsentationsebenen, eine für abstrakte Geschäftsprozesse für die Dokumentation, Kommunikation und den Austausch unter Fachanwendern und die zweite für eine kanonische Darstellung zur Ausführung von Prozessen. Beide Ebenen basieren auf dem LPM-Metamodell. Die dritte Komponente beinhaltet eine technische Architektur und Werkzeuge, die den Fachanwender bei der Modellierung, beim Deployment und bei der Ausführung von Prozessmodellen unterstützen. Dies umfasst prototypische Spezifikationen für Front- und Backend-Werkzeuge, wie einen Prozesseditor und eine Laufzeitumgebung. Zusätzlich wird ein Designprozess definiert, der die Interaktionen der Werkzeuge beschreibt, um abstrakte Prozesse mit Informationen für die Ausführung zu erweitern.

Die Ausarbeitung der LPM und die Beschreibung des Artefakts der Leichtgewichtigen Prozessmodellierungslösung (LPMS) folgt der Design Science als Forschungsmethodik.

1 INTRODUCTION

This section introduces the topic of Lightweight Process Modelling (LPM). Hence, section 1.1 covers a description of the current situation and motivation of this thesis. Section 1.2 formulates the research questions and objectives. The section 1.3 addresses the research methodology, namely the application of the design science to this thesis. Lastly, section 1.5 presents the outline of the thesis.

1.1 CURRENT SITUATION AND MOTIVATION

Business Process Modelling (Williams, 1967) aims at documenting, communicating, analysing, and supporting collaboration between people, often in a business environment. According to Gartner, processes need to be visible to potentially change and improve them (M. Cantara, 2009). In addition, Gartner states that process models form the base for an effective collaboration between business users, business leaders, and IT. Process models serve as well as documentations for auditing or compliance procedures.

Another purpose of process models is to automate processes within an existing infrastructure. According to a Gartner Report (Plummer & Hill, 2009), the design of business systems will be more and more subject to composition and BPM replacing traditional application development. In order to abstract process logic from application logic, Business Process Management Systems (BPMS) (Chang, 2005) allow for the definition and execution of processes by invoking underlying applications or services.

BPM has two main purposes
- Documentation of processes - Automation of the process execution

A study[1] about BPM adoption and usage can be found in (Palmer, 2009). According to this study, the motivation for moving to a BPM architecture is manifold. Besides the utilization of existing infrastructure and software assets in proprietary languages, exposed as services, the use of internal development resources, a faster time to market through reduced deployment time, lower deployment costs through ease of application integration, lower IT support staff costs through on-going application support levels, and hence, lower total cost of ownership are key to the introduction of a BPM solution. In figures, the study revealed that the respondents expect a higher reuse of existing resources (21% of the respondents), lower maintenance costs (19%), faster product development (19%), and freeing up IT resources for other initiatives (16%). Further, this study revealed that many organisations will invest in BPMS training and skills development (27%) and process modelling (19%). Currently however, Microsoft Visio is by far the process modelling tool with the largest number of implementations (Norton, Blechar, & Jones, 2010).

The aforementioned fact that composition and BPM are replacing traditional software development is strongly related to a shift from development within the IT departments to a collaborative approach involving business experts, modellers, analysts, developers, and architects (Plummer & Hill, 2009). Process models are the means of communication between those stakeholders. Hence, according to Plummer and Hill, model-driven approaches are becoming the primary method to develop and maintain software.

Currently, modelling executable processes requires expertise in both business and IT. This renders existing BPMSs and process modelling languages unsuitable for the business user. However, business users build the majority of information workers and deciders. Gartner (Blechar, 2007) and Forrester (Richardson, Moore, & Nicolson, 2009) also state that current tools for business process analysis are not suitable for business users. By business user, information workers are referred to that are 'not

[1] The study was conducted by a consortium of Transformation & Innovation, BPM.com, and Workflow Management Coalition

casual, novice, or naive' (Nardi, 1993) but have got strong domain-specific business skills. Concerning IT they have computational needs but limited IT knowledge and no interest in becoming an IT professional (Nardi, 1993; Quinn, 2005). Furthermore, they usually do not model processes. An example of a typical information worker is a travel agent interacting with customers as well as multiple information systems in order to make reservations.

The business user might also be named non-experienced user or non-technical user. Due to the lack of knowledge, reuse of existing process models is low. Proprietary model representations are created by using office tools, such as Microsoft PowerPoint. These models are neither interchangeable nor executable. Often, there is no common understanding of business process models and the terminology used (Blechar, 2007; Mendling & Recker, 2008). In addition, users have difficulties in delivering process models at a certain quality level and commit typical modelling errors (Koehler & Vanhatalo, 2007). A study conducted by Namoune et al. (Namoune, Wajid, & Mehandjiev, 2009) revealed that although 80% of the participants had been interested in service composition, there had been significant fears about creating errors in process modelling. Furthermore, in the study, composition problems of business users could be clearly revealed. The surveyed users agreed on frequent frustration in the context of service complexity, compatibility, and composition. Process changes are mostly driven by the business making it difficult to keep the IT in synchronisation with these business changes (M. Cantara, 2009).

Current BPMSs

- Provide documentation functionality for business users
- Provide process execution functionality for IT experts
- Leave a gap between process documentation and process execution

A simple way for users to understand business processes is a key success factor to encourage for taking ownership of the process and making process changes (Rosser, 2008). This speeds up process management and lowers costs since experts are only needed in few cases.

The described challenges clearly formulate a need for Lightweight Process Modelling (LPM) implementing a process modelling solution that the business user might use in a *lightweight* way, so required for knowledge expression in general in (Lillehagen & Krogstie, 2008). In this sense, the term *lightweight* concerns the user interaction and means intuitively, abstractly represented, and easy to understand in the context of process modelling and easy to deploy, implement, and execute processes in a tooling context. The process modelling solution should follow the trend of business user enablement similar to the end-user programming paradigm described in (MIT, 1993) in order to cope with these challenges.

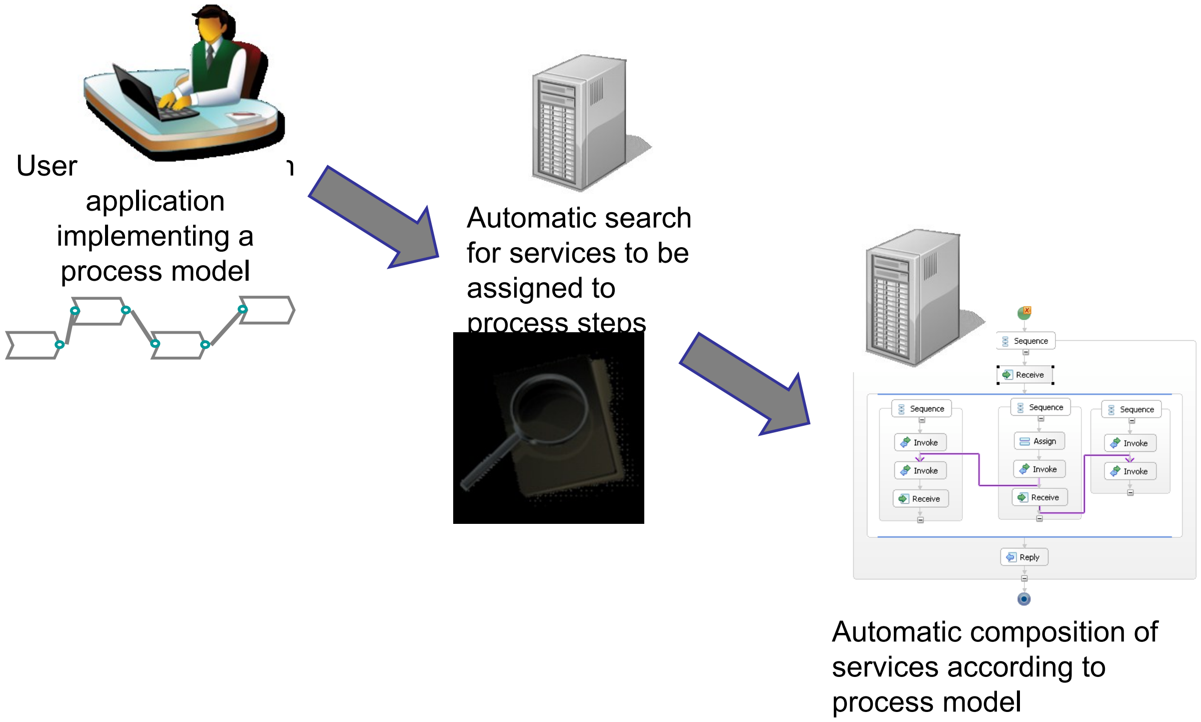

User
application implementing a process model
Automatic search for services to be assigned to process steps
Sequence
Receive
Sequence
Sequence
Sequence
Invoke
Assign
Invoke
Invoke
Invoke
Invoke
Receive
Receive
Reply
Automatic composition of services according to process model

Figure 1: How LPM supports the user in creating an application implementing a process

Figure 1 visualizes the support through LPM on a high level. The user needs to create an application that implements a process model. He therefore models the process. Afterwards, the LPM technologies automatically search for and return services to be bound to the abstract activities in the process model. In the final step, the found services are automatically composed according to the process model.

In order to realize a lightweight user access, sophisticated backend solutions are needed. In particular, the deployment aspect of the lightweightness is important for users or organisations without runtime capabilities.

A need for allowing the business user to document and execute processes has been identified

1.2 Research Questions and Objectives

As described in the introductory section, there's a need to specify the design of a solution enabling non-IT-savvy business users to model and execute their processes in a lightweight way.

The following *Research Questions* and *Objectives* are addressed by this thesis. The thesis aims at answering those questions by describing the design for a solution. The goal of the thesis is not to specify a product-like solution.

Which design principles and meta artefacts are appropriate for LPM supporting the business user in modelling and executing processes?

> Hereby the research objective is to define a metamodel for LPM design principles, the LPM metamodel. This metamodel for design principles is applied to according artefacts, such as a process modelling language and supporting tools.

How should a solution for LPM be designed that enables business users to model executable processes?

> The according research objective is to define the design of the Lightweight Process Modelling Solution (LPMS) targeting the business users.

How should artefacts, such as a language and tools, for the LPMS look like?

> The objectives targeted by this research question are twofold:
>
> The first objective is to create a Lightweight Process Modelling Language (LPML) defining syntax and semantics for the representation of process logic. The language should allow for the representation of abstract business processes for documentation, communication, and collaboration purposes for business users. Further, a representation format is required that might be used to execute process models. The language should balance its expressive power with conceptual simplicity and clarity targeting business users.
>
> The second objective is to create the design of the technical architecture and tools for the LPMS that support the business user in deploying and executing the process models. This comprises a design process defining the user and machine interactions in order to model and enhance abstract processes by execution details. Furthermore, both front- and back-end tools are required, such as a process editor and an execution engine.

How does the LPMS including the LPML reflect these design principles?

The objective of this question is to describe the language elements implementing the LPM metamodel.

1.3 Target Users

The target users addressed by LPM are end-users. They might be differentiated according to their IT-skills. The following user types have been identified, in accordance to Schnabel et al. (Schnabel, Born et al., 2009):

1. *IT experts*: Users that have a significant IT education or significant experience in developing and using software. In the context of this thesis, these IT experts have experience in service programming and usage as well.
2. *Business users*: Users who have strong business knowledge in other domains than IT. These users use IT systems to support their work and to achieve their work objectives. Concerning IT, they have computational needs but limited IT knowledge and no interest in becoming an IT professional (Nardi, 1993; Quinn, 2005). In the public sector use case described in section 1, these users are mainly civil servants and town-hall administrators.
3. *Casual business users*: These users are neither expected to be IT experts nor experts of another business domain. Even if they were experts in any way, they would use the LPMS for purposes that are not related to their main line of work. These users might be seen as the lowest common denominator profile.

1.4 Research Methodology

The goal of this thesis is to develop a metamodel, a language, and tools for LPM. This is a design-oriented goal aiming at the development of an artificial artefact (Simon, 1969). In contrast to behavioural science describing existing phenomena, the design science covers the prescription of artefacts (Hevner, March, Park, & Ram, 2004; March & Smith, 1995). Therefore, the design science is applied as research methodology in this thesis.

1.4.1 Design Science Guidelines

A framework of seven design science guidelines is presented in (Hevner et al., 2004). In the following Table 1, these guidelines are applied to the research work of this thesis.

Table 1: Application of the design science guidelines to this thesis

DS Guideline	Description	Application in this thesis
Design as an artefact	"Design-science research must produce a viable artefact in the form of a construct, a model, a method, or an instantiation" (Hevner et al., 2004; March & Smith, 1995).	LPM and the LPMS are defined as a design artifact. Besides the LPM metamodel, a language and tools for implementing the LPM design principles are presented.
Problem Relevance	Describe important and relevant business problems.	Existing BPM solutions to both model and execute processes are not suitable for business users.
Design evaluation	"The utility, quality, and efficacy of a design artefact must be rigorously demonstrated via well executed evaluation methods" (Hevner et al., 2004).	Evaluation of LPM and the LPMS based on the requirements, such as technical integration, usability, suitability for collaborative modelling, modelling time, or training effort.
Research Contribution	Describe new and interesting contributions.	LPM and the LPMS allow business users to model and execute business processes without specifying detailed execution information.
Research Rigour	Potential scientific methodologies to apply: Case studies, literature research, action research	Research rigour is guaranteed through literature research and a case study for applying the LPMS to the area of the Public Sector.
Design as a search process	Means (set of actions and resources	This thesis is written in the context of the research project SOA4All[2]. Further, a public sector use case is built around the LPMS.

[2] See www.soa4all.eu

	Ends (goals and constraints on the solution)	Simplicity and usability, graphical abstraction, automated service discovery, composition, and execution.
	Laws (uncontrollable forces in the environment)	It is assumed that semantically described service interfaces exist and might be discovered in the web.
Communication of research	Present research work to both technology- and business-oriented people.	LPM is part of the project work in SOA4All. Various deliverables, a scientific conference publication, and a SAP Whitepaper have been published.

1.4.2 The Design Science Research Process

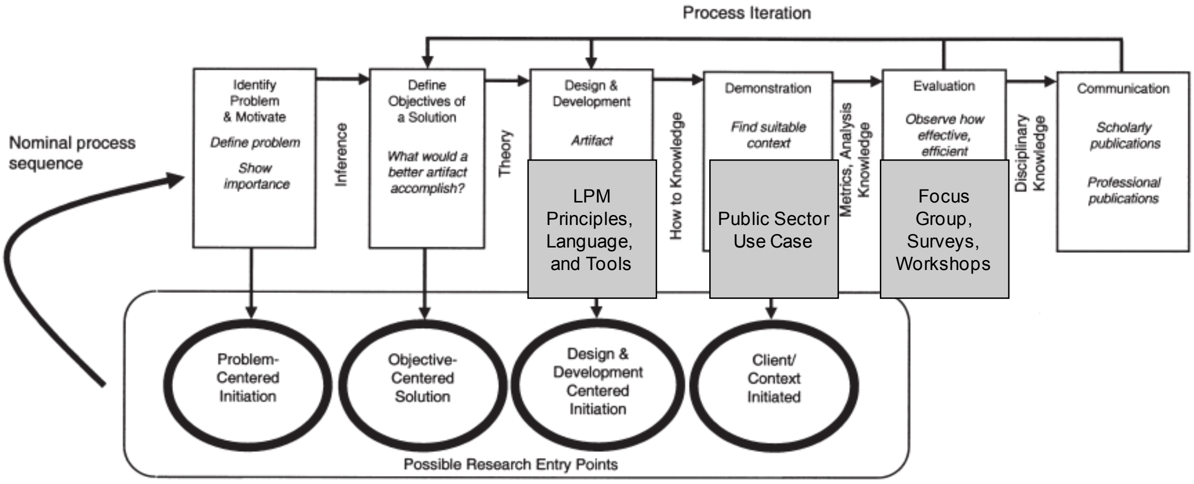

Process Iteration
Nominal process sequence
Identify Problem & Motivate
Define problem
Show importance
Inference
Define Objectives of a Solution
What would a better artifact accomplish?
Theory
Design & Development
Artifact
LPM Principles, Language, and Tools
How to Knowledge
Demonstration
Find suitable context
Public Sector Use Case
Metrics, Analysis Knowledge
Evaluation
Observe how effective, efficient
Focus Group, Surveys, Workshops
Disciplinary Knowledge
Communication
Scholarly publications
Professional publications
Problem-Centered Initiation
Objective-Centered Solution
Design & Development Centered Initiation
Client/ Context Initiated
Possible Research Entry Points

Figure 2: Application of the Design Science Process according to Peffers et al.

Various approaches exist to implement the design science guidelines in a research process (Hevner et al., 2004; March & Smith, 1995; Nunamaker, Chen, & Purdin, 1991; Peffers, Tuunanen, Rothenberger, & Chatterjee, 2008; Rossi & Sein, 2003). Figure 2 depicts the application of the Design Science Process according to Peffers et al. (Peffers et al., 2008) to this thesis. In the following Table 2, the research processes of Rossi and Sein (Rossi & Sein, 2003) and of Peffers et al. (Peffers et al., 2008) are applied to this thesis.

Table 2: Design Science process according to (Rossi & Sein, 2003) and (Peffers et al., 2008)

Process Phase	Addressed by this thesis
Identify a need (Rossi & Sein, 2003) Identify problem and	See section 1.1 (Current situation and motivation) and 3.1 (problem statement): Existing BPM solutions are too complex for business users.

motivate (Peffers et al., 2008)	
Define objective of a solution (Peffers et al., 2008)	See section 1.2: Define an environment for LPM targeting the business user
Build (Rossi & Sein, 2003) Design & Development (Peffers et al., 2008)	See section 1 (Lightweight Process Modelling Principles) and section 1 (Lightweight Process Modelling Solution).
Demonstration (Peffers et al., 2008)	See section 1 about a use case of the LPMS in the public sector.
Evaluate (Rossi & Sein, 2003) and (Peffers et al., 2008)	See section 1 about the LPM evaluation.
Learn and theorize (Rossi & Sein, 2003)	The work of this thesis contributes to the research project SOA4All that is part of the European FP7 Research Program. Further, section 1 covers aspects of the learn-and-theorize phase.
Communication (Peffers et al., 2008)	LPM is the main work in the research project SOA4All comprising deliverables as the means of communication. Further, a scientific publication and a SAP Whitepaper have been published.

1.5 Outline

This thesis is organised as follows. Section 2 gives an overview of related work that is basic to the specification of LPM and the LPMS. The problem statement, the requirements to LPM, an overview of the LPM approach, the research context, and the coverage of parts of the LPM through existing solutions are subject to section 1. The design principles for LPM, the LPM metamodel, and the implementation of the LPM metamodel in the LPML are covered by Section 1. In Section 1, the LPMS is described in detail, comprising the design process to create executable process models and according tools. This section reveals how the user is supported through LPM. How LPM and the LPMS are used in practice in a use case is subject to Section 1. Section 1 reports the evaluation of LPM and the LPMS. Finally a conclusion and an outlook are given in Section 1.

2 RESEARCH FRAME

This section presents basic theories, methodologies, and technologies that influence the LPM environment. Firstly, terms and definitions for this thesis are presented in section 2.1. This comprises the relevant terms in the context of process, service, modelling, and semantics. Afterwards, the trend of end-user empowerment (section 2.2), bottom-up approaches to distribute IT technology (section 2.3), business process management (section 2.4), and semantic technologies (section 2.5) are introduced.

According to the Design Science guidelines, this section covers the Research Contribution guideline.

2.1 TERMS AND DEFINITIONS

2.1.1 Process

In this thesis, a process is defined according to Green and Rosemann (P. Green & Rosemann, 2000; M. Rosemann, Sedera, & Sedera, 2001) as "*a self contained, temporal and logical order (parallel or serial) of those activities, that are executed for the transformation of a business object with the goal of accomplishing a given task*". This definition is based on the definition of Davenport (Davenport, 1993) who states that "*a process is simply a structured, measured set of activities designed to produce a specified output for a particular customer or market. It has a specific order of activities across time and place, with a beginning and an end and clearly identified inputs and outputs with a structure of action.*"

2.1.2 Service

In this section, a definition of the term *service* is presented. Firstly, various definitions based on literature are listed. Afterwards, the common parts of these definitions are analysed in order to create a definition valid for this thesis. Table 3 lists the definitions found in literature.

Table 3: Definitions of the term "Service"

Source	Definition
(Brown & Cantor, 2006; High, Kinder, & Graham, 2005)	A service encapsulates a set of resources that can execute a repeatable task within a process. Services are described by their behavior and interfaces. The service consumer rarely has visibility to the resources; rather,

	the consumer invokes the service through the service interface.
(Plummer, 2005)	In SOA, a service is a modular piece of software (service provider) with a well-described interface that can be activated by another modular piece of software (service consumer). The service consumer doesn't need to understand the technological implementation of the service provider. This implies the concept of "loose coupling" (that is, the service provider can be changed without forcing the service consumer to be changed as well). In this sense, the provider and consumer are loosely coupled, rather than tightly, as in monolithic software architecture leading to a more-modular system that can be changed readily.
(W3C Web Services Architecture Working Group, 2004)	A service is an abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of providers' entities and requesters' entities. To be used, a service must be realized by a concrete provider agent. A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.
(Böhmann, Junginger, & Krcmar, 2003)	[A service is] the application of business and technical expertise to enable organizations to create, manage, optimize or access information and business processes.
(OASIS, 2005)	[A service is] a behaviour or set of behaviours offered by one entity for use by another according to a policy and in line with a service description.

Table 4 covers the analysis of the criteria for the service definitions. An „X“ in the according cell means that the criteria is mentioned in the definition. A comment will be presented, if the criteria are modified or extended.

Table 4: Literature criteria for the service definition

Criteria	(Brown & Cantor, 2006; High et al., 2005)	(Plummer, 2005)	(W3C Web Service Architecture Working Group, 2004)	(Böhmann et al., 2003)	(OASIS, 2005)	(Dostal, Jeckle, Melzer, & Zengler, 2005)	(Zimmermann, Krogdahl, & Gee, 2004)
Encapsulated, modular piece of software	X	X					
Set of resources	X		Abstract resource				
Service provider performs a task	X		X		X	X	X
Within a process	X			Creation, management , optimization , and access to processes		X	X
Services is characterised by its behaviour	X				X		
Interface	X	Well-defined	X				
No visibility for service consumer	X	User doesn't have to understand the technical implemen					

		tation					
User calls service (human or machine)	X	Interface activated through further software			X	X	X
Lose coupling		X					
Application of technical and business expertise				X			
In line with service definition					X		

Based on the criteria extracted out of the definitions listed in Table 3 and Table 4, this thesis uses the following service definition:

"A service is a modular piece of software that is encapsulated behind an interface and performs a specific task. It is offered by a service provider and called by a service consumer. Various services are coupled loosely."

2.1.3 Modelling

According to Brockhaus (Brockhaus, 2002), in computer science, modelling is the proceeding of transforming a real scenario into a model. The model is an image of an object or of an object area that emphasizes the essential characteristics and dismisses aspects that are not considered relevant (Brockhaus, 2002). Like in computer science, modelling has an important role in information management. A definition of modelling in information management can be found in Winter (Winter, 2003). According to Winter, modelling is the proceeding of constructing an image of real or virtual scenarios that is based on the perception of the modeller and is influenced by the modelling purpose.

2.1.3.1 Models and the Model Term

The results of the modelling procedure are models that are images of real or virtual scenarios, influenced through perception and purpose (Schütte, 1998). The model term is characterised by three main attributes. These cover the image attribute, the reduction attribute, and the pragmatic attribute (see (Thomas, 2005) based on (Stachowiak, 1973)). The image attribute covers the fact that models are always images or representations of natural or artificial originals. The fact that a model covers only relevant aspects is addressed by the reduction attribute. The pragmatic attribute refers to the fact that models might not always be assigned unambiguously to their originals.

The model term can be characterised according to their nature (Thomas, 2005). The axiomatic model term represents the semantic or empirical interpretation of a formal system of axioms. The image-oriented model term is widely spread. Main characteristic is the image of the objective real world into a subjective model. The image-oriented model term builds the foundation for process modelling that is described in detail in section 2.4.2. The last model term is the construction-oriented model term. This term is based on the assumption that the real world is not objectively existent and tangible but bound to subjects. A model cognition and creation is not built through the imaging of a reality part but through construction. In terms of construction-orientation, starting point is a problem definition that is structured through the creation of models in order to facilitate the solution process (Bretzke, 1980; Rieper, 1992; Schütte, 1997; Thomas, 2005).

2.1.3.2 Model Type

Models are assigned to model types. Winter (Winter, 2003) differentiates various dimensions that support the structuring of models. An as-is-model represents existing scenarios. In contrast, a target model or a reference model represents desirable or recommended scenarios. A further dimension covers the degree of specialisation of a model. Generic models are valid in a plethora of application contexts whilst specific models are only valid in a certain context. The third dimension describes the abstraction level of the components. Detail or view models concentrate on specific parts. Aggregation models compose multiple aspects.

In terms of syntax and semantics, models are categorized into informal, semi-formal, and formal models. Informal models mostly lack unambiguous description syntax. Semi-formal models follow a specific syntax, however do not follow concrete construction rules of a formal theory. For formal models, the semantics is well-defined, besides the definition of the syntax. An example of formal models are Petri-net-models (Peterson, 1981; Petri, 1962).

The last dimension specifies the dynamicity of models. Static models concentrate on one state, comparatively static models comprise multiple states. Lastly, dynamic models allow for the modelling of flows.

2.1.3.3 Definition of Model

In this thesis, a model is a representation of a real world scenario. This representation is an image of the real world, reduced, pragmatic, and has a specific modelling purpose.

Modelling is the proceeding of creating a model.

2.1.4 Semantics

Semantics is "*the study or science of meaning in language*" (TheFreeDictionary, 2011). According to "The Free Dictionary" it is further "*the meaning or the interpretation of a word, sentence, or other language form*". In computer science, semantics is not restricted to the meaning of words, but is as well applicable to the meaning of languages and their artefacts. Hence, in this thesis, semantics is valid for all artefacts in the context of LPM.

2.2 END-USER EMPOWERMENT

An increasing need for knowledge-intensive work and quick changing user requirements and environments require an effective IT support for users' tasks and processes. Further, managing high quality and competitive knowledge forms the basis for individual action and hence, for effective enterprises in the global competition (Wiig, 2004).

With the emergence of distributed computing, applications are more and more developed by the people that directly need it (Brancheau & Brown, 1993). As a result, end-user computing (EUC) emerged allowing people outside the information systems department to adapt and use information technology in order to develop their own applications. A definition of Brancheau and Brown (Brancheau & Brown, 1993) is referred to describe end-user computing: *"End-user computing is defined as the adoption and use of information technology by personnel outside the information systems department to develop software applications in support of organizational tasks."* Another term for EUC is End-User Development (EUD) that is considered as homonymous to EUC in this thesis.

Early approaches of EUC started in the early 1970s. However, not until the late 70s and early 80s, EUC had been seen as a solution to organizational problems with traditional software development (Brancheau & Brown, 1993; Canning, 1981a, 1981b; Martin, 1982; McLean, 1979).

EUC

- Promises to enable business users to create software
- Promises to overcome the shortage of IT experts
- Is a trade-off between user training costs and the benefits of closing the gap between business and IT

EUC promises to reduce the applications development backlog and to overcome the shortage of IT experts (Brancheau & Brown, 1993). The large majority of the typical workforce of an organization is enabled to participate in development and composition tasks (see Figure 3). Even if a single user is only able to contribute a small part, the composition of these parts might produce significant results. For Lillehagen and Krogstie, EUC is a prerequisite to the active management of enterprise knowledge including the BPM area (Lillehagen & Krogstie, 2008). The authors describe a Knowledge Management Architecture to make knowledge explicit through modelling approaches. Further, the authors see a lack of tools for inexperienced modellers, a lack of scientific methodologies, and a lack of dynamic visual languages as main shortcomings of actual enterprise modelling approaches. They intend to enable users to create and maintain personnel working environments.

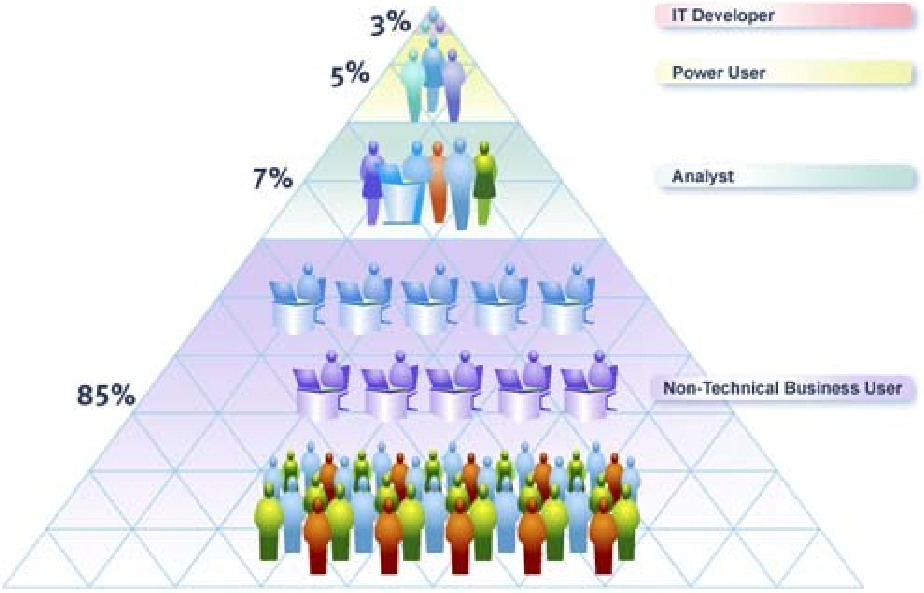


Figure 3: Leveraging the wisdom of the crowds with EUD (Quinn, 2005)

An overview of early EUC approaches can be found in literature (Brancheau & Brown, 1993; Powell & Moore, 2002). In addition, an overview of later EUC approaches has been published by Spahn et al. (Spahn, Dörner, & Wulf, 2008). The motivation for EUC is a balance of benefits and costs (Fischer, Giaccardi, Ye, Sutcliffe, & Mehandjiev, 2004; N. Mehandjiev, Sutcliffe, & Lee, 2006; A. Sutcliffe, 2005; A. G. Sutcliffe, Lee, & Mehandjiev, 2003; Wulf & Jarke, 2004). Hard-coded software is often limited in supporting heterogeneous and changing tasks in a flexible way due to development, installation, and configuration time.

The benefits of EUC are more effective job executions, higher development speed, higher flexibility, more local control, and an avoidance of frequent misunderstandings between business and software specialists. However, costs have to be taken into account, such as selecting appropriate technology, installation and training costs, user guidance, and programming and debugging (Fischer et al., 2004; A. Sutcliffe, 2005). These costs have to be regarded on the individual level as well as the trade-off between cognitive costs of programming and the benefit of achieving a better fitting result (Blackwell, 2002). Non-cognitive costs, such as a loss of personnel time, are recognised as well in literature (A. G. Sutcliffe et al., 2003). Furthermore, issues about control sharing between IT and business departments might occur (Brancheau & Brown, 1993). However, the responsibility shift cannot only be seen as an IT issue but as political, social and cultural issues associated with the users (McBride & Wood-Harper, 2002). A list of benefits and risks of EUC and of supporting actions can be found in a report published by Mehandjiev (N. Mehandjiev et al., 2006).

In the area of BPM, the need for end-user development is agreed as well. A shift from the enterprise view of processes to address the needs of an individual acting in multiple processes is foreseen. This is called the "process of me" in Genovese et al. (Genovese, Comport, & Hayward, 2006) or "activity-centric process" in Hill et al. (C. Hill, Yates, Jones, & Kogan, 2006). An approach for deriving processes from task management is discussed in literature as well (Riss, Rickayzen, Maus, & van der Aalst, 2005; Stoitsev & Scheidl, 2008; Stoitsev, Scheidl, Flentge, & Muehlhaeuser, 2008). To base process models on users' tasks results in consistent, real-life compliant process models. These approaches derive process models based on users' tasks for scenarios where a global process structure does not exist, such as for search processes. The challenge is to make these unstructured processes executable. LPM promises to contribute to this challenge by providing means for knowledge workers – the business users in this thesis - to model and execute those processes in a user-friendly way. Hereby, the basic process models might be derived from task structures as described above and then enhanced via LPM to make the process models executable.

A study about the extent and the attitude towards the costs and benefits of EUC has been performed by Mehandjiev (Nikolay Mehandjiev, 2008). Vogel et al. summarised the main results of the study (Vogel et al., 2009). In this study, business users of large client companies of the SAP AG[3] and of non-technical divisions of SAP have been surveyed, in total, 133 respondents participated in the survey.

A result of the study revealed that the surveyed business users are already performing various EUC activities. 75% of the respondents claimed familiarity with typical EUC functionalities, such as creating rules for filtering or forwarding emails. Furthermore, only 2% of the business users with almost no programming experience indicated that they hadn’t performed any EUC activities.

Another study result has been that the targeted business users trust in the fact that benefits from EUC outweigh its costs. Therefore, the balance of risk and cost expectations against the benefit expectations has been surveyed. The obtained result reveals a value of 0.92 in a -4 to +4 scale. Furthermore, the study revealed that the respondents expect to be able to perform the needed EUC activities (average value of 0.92 on a -2 to +2 scale).

To conclude, the benefits of end-user development mostly outweigh the potential risks which lead to the decision to follow an end-user empowerment approach in this thesis.

2.3 Bottom-Up

In a bottom-up approach
- The workforce is playing the key role in a decision process
- A software solution might be leveraged to become a mainstream solution

Besides approaches for end-user empowerment discussed in the previous section, the paradigm of bottom-up theories influences the LPMS. Basically, in a bottom-up approach, the workforce is playing the key role rather than the top management in a tool purchase or usage decision.

[3] See www.sap.com for more information on the SAP AG

A key success factor for an IT solution targeting business users is the acceptance amongst the target group. Current BPM solutions are focusing a small group of IT or BPM experts. However, in order to leverage the full potential of BPM and to become a mainstream solution, those users have to be focused that possess the business and process knowledge. In order to raise acceptance, the LPMS has to provide value for those users rather than for the management.

An example of a successful bottom-up approach is the business model of Salesforce.com[4]. The provided SaaS model is cheap enough that middle management might consume the provided services within the limits of their own budgets. In contrast, traditional software models covering the whole functionality stack are very expensive and require the approval of the top management. Hence, Salesforce.com solutions slowly penetrate entire organisations through a bottom-up approach. Another example is Skype[5], an IP phone provider. Although blacklisted in a couple of companies Skype is used by the employees and provides value to them. In consequence, Skype might benefit from the additional user base that might be used for advertising. As well, Google[6] increases its user base through the provisioning of a large amount of storage space for its email service Gmail[7]. Often, companies provide less storage space leading employees to use their Gmail accounts for business emails. A larger user base might leverage advertisement incomes for those companies, such as Skype and Google.

As aforementioned, the LPMS envisages providing value to the business users. By achieving a high acceptance amongst the business users, significant business opportunities might arise for the provider of the LPMS. Furthermore, the acceptance degree of the overall set of processes within an organisation will increase since the processes are directly modelled by those users who execute them.

[4] See www.salesforce.com

[5] See www.skype.com

[6] See www.google.com

[7] See www.gmail.com

2.4 BUSINESS PROCESS MANAGEMENT

2.4.1 Business Process Management

Process management theories base the management of an organisation on its core processes. A definition of a process has been stated in section 2.1.1. Business Process Management (BPM) (Elzinga, Horak, Chung-Yee, & Brunner, 1995) has its roots in Business Process (Re)Engineering (Davenport, 1993; M. Hammer, 1990; M. Hammer & Champy, 1993; Michael Hammer & Champy, 1994; M. Hammer & Stanton, 1999; Jablonski & Bussler, 1996) and Workflow Management (Allen, 2001; Hollingsworth, 1995) that is focused on the use of software to run processes. Armistead et al. (Armistead, Machin, & Pritchard, 1997) formulate questions of what is Business Process Management:

"Firstly, is it a series of tools and techniques for improving the performance of business processes whether they be categorised as operational, support or directions setting. Or is it a way of integrating the management of the whole organisation? Secondly, if business process management is the latter, how can it be made to work? Finally, is it a tool for organisational design which needs to be understood by only a few within the organisation?"

In this thesis, the latter scope is clearly recognized. With methods, techniques, and software, BPM includes a broader scope than workflow management to support the handling of processes (van der Aalst, ter Hofstede, & Weske, 2003). BPM is more a kind of management discipline and covers the main aspects of business operations (Zairi, 1997). As is described in this thesis, BPM concerns not only a few expert people within an organisation but all kinds of information workers.

BPM is a management discipline covering
- Process documentation - Automation of process execution

An early summary of BPM literature is provided by (Lee & Dale, 1998). Van der Aalst et al. further define a BPM lifecycle comparing workflow management to BPM. This lifecycle comprises four phases: process design, system configuration, process enactment, and diagnosis. Whilst workflow management concentrates on parts of the design, system configuration, and parts of the enactment phase, the BPM covers all phases.

BPM is characterised by various main principles that can be found in literature (Michele Cantara et al., 2009; Lee & Dale, 1998; Zairi, 1997) and are summarised in the following.

- Customer focus through the linkage of key activities in horizontal processes.
- Quality focus through well-defined procedures that ensure discipline and consistency. Processes are made visible and hence explicit to business and IT people through common models and languages. Further, the quality of each individual process is measured and monitored to ensure a specific service delivery level.
- Business models have to be in line with process execution. Both models and execution systems should allow for the quick adaptation to changing requirements and environments.
- BPM has to be implemented as a continuous approach to improve and optimize procedures and end-to-end processes within and across organisations including partners, suppliers, and customers. Further, an objective is to increase competitiveness.
- Culture change and not only implementation of process-aware systems.
- Integration of process activities, measurement methodologies, rule management, content integration and collaboration
- Empowering business users and analysts to manipulate business process models and instances

The automation of processes is subject to workflow management. A workflow is "*the computerised facilitation or automation of a business process, in whole or part*" (Hollingsworth, 1995). Allen (Allen, 2001) further extends this definition by including the passing of documents, information, or tasks between participants according to procedural rules. Workflows typically describe coarse-grained activities and applications (Alonso, Casati, Kuno, & Machiraju, 2004). Workflow Management Systems (WfMS) allow for the processing of workflows (Allen, 2001). According to Allen, a WfMS is defined as "*a system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools and applications*".

According to a Gartner report (Plummer & Hill, 2009), applications will be more and more replaced by orchestrations of data, processes, and services. In order to explicitly abstract process logic from application logic, Business Process Management Systems (BPMS) (Chang, 2005) allow for the definition and execution of processes by invoking underlying applications or services. A BPMS might be seen as an evolution of the definition of a WfMS.

2.4.2 Business Process Modelling

As described in section 1.2, the goal of this thesis is to describe the design of a solution enabling business users to model and execute processes. Therefore, an executable process modelling language is required comprising a graphical abstraction the business user might understand. Further, the language has to allow for the specification of semantic annotations. In the following, a general introduction to business process modelling is given. Afterwards, existing process modelling languages are investigated in terms of their suitability for business users, ability to include execution information, and ability to include semantic annotations.

Business Process Modelling (Williams, 1967) has been introduced in order to document, communicate, analyse, and support collaboration in pursuit of business needs (Davis, 2001; Harrington, 1991). Scholz-Reiter and Stickel (Scholz-Reiter & Stickel, 1996) emphasise in their definition the transformation of knowledge about business systems into process models. Explicit process models might be required for quality objectives or compliance. Another purpose of process models is to automate processes within an existing infrastructure. Van der Aalst and van Hee (van der Aalst & van Hee, 2004) see the definition and selection of appropriate tasks as part of workflow modelling. The authors suppose a task library where the tasks might be taken from. In this thesis, the selection of tasks is performed in the web which requires more sophisticated mechanisms. Furthermore, van der Aalst and Hee see sequencing of the tasks to satisfy dependencies, allocation of resources and agents to the tasks, scheduling of tasks considering concurrency, and validating and verifying the model as part of workflow modelling.

Business process modelling
- Serves for the documentation of processes - Allows for automation through executable process models - Might be structured through functional, operational, behavioural, organizational, and informational perspectives

A model has to provide information about activities, who is performing those activities, when and where the activities are performed, how they are executed, and the data elements manipulated by the activities (Giaglis, 2001). Process models might be differentiated according to the following perspectives defined by Jablonski and Bussler in the *Mobile Framework* (Jablonski & Bussler, 1996) and refined by Curtis et al. (Curtis, Kellner, & Over, 1992).

- The *functional* perspective describes *what* process elements (activities) are being performed.
- The *operational* perspective describes which operations are supported.
- The *behavioural* perspective describes *when* activities are performed (sequence or parallel) as well as aspects of *how* they are performed through feedback loops, iteration, decision-making conditions, entry and exit criteria, etc.
- The *organizational* perspective represents *where* and *by whom* activities are performed, the physical communication mechanism used to transfer entities, and the physical media and locations used to store entities.
- The *informational* perspective represents the information entities (data) produced or manipulated by a process and their interrelationships.

Business Process Modelling is a part of enterprise modelling. An approach for a business modelling framework has been developed by Lo and Yu (Lo & Yu, 2008). Like enterprise modelling, process modelling has to cope as well with the balance of allowing freedom degrees and a reasonable degree of quality. This balance is described by the term "Generally Accepted Modelling Principles" (GAMP) (J. Becker, Rosemann, & Schütte, 1995; Scheer, 1998). The GAMPs cover principles for correctness, relevance, efficiency, clearness, comparability, and systematic structure. In this thesis, these principles are applied to the creation of the LPML and the according Process Editor. The editor has to make sure that only correct models might be created, that the creation is efficient, that the models are clear in order to be executable, and follow a systematic structure. The principles of relevance and comparability are subject to the process models themselves and less to the modelling language and tools.

A need for supporting the creation of correct, sound, meaningful, and commonly understandable process models could be identified.

However, creating sound models is not sufficient to enable an alignment and a common understanding of processes as well as to create a homogeneous set of processes on the same granularity level within an organisation. Meaningful, understandable, and maintainable process models have to be created. Born et al. (Born, 2009; Born et al., 2009) describe a proceeding for the alignment of process models. The proceeding is based on a common naming convention for process elements and a business term repository containing activity names and descriptions in order to reveal their relations. The goal is to enforce the users applying the naming convention and to use the terminology provided by the repository. In particular, business users benefit from a quick understanding of process models with terminology that is common in their context. By implementing these aspects, further, a context-driven recommendation system might be implemented. Section 4.4 covers context-awareness of the LPMS.

Since the LPMS deals with services, the process modelling procedure might be seen as a service composition task. In general, three approaches exist to compose services: Modelling the control flow, modelling the data flow, and modelling through an assisted approach. The assisted approach provides a composition template and supports the user in selecting services from a predefined list. Wajid et al. (Wajid, Namoune, & Mehandjiev, 2010) compared these three approaches of service modelling by business users through a modelling workshop. The result of the workshop revealed that users liked the assisted approach best. Although the author of this thesis agrees to the workshop result to provide an assisted modelling approach, the LPMS has to keep flexibility in order to model and execute any kind of processes. Hence, the LPMS primarily focuses on supporting the user in modelling data and control flow. However, for a specific domain, it is reasonable to provide an existing set of services and support the user in selecting one of them.

2.4.2.1 Process Modelling Languages

The user interface of BPM is a process modelling language allowing the user to orchestrate tasks and activities in a process structure. The language should provide a graphical abstraction layer and an executable format, such as formats for workflow modelling. The basis for workflow modelling are Petri nets (Pankratius & Stucky, 2005). Petri nets are a formal workflow modelling language and allow for the creation of a workflow model out of various models in terms of algebraic operations. The formal notation of Petri nets might be used to define other formal workflow notations. This section starts with an overview of graphical process modelling languages. Afterwards, executable process modelling languages are described.

Graphical process modelling languages

Currently, a couple of graphical process modelling languages exist, an overview is provided by Ko et al. (Ko, Lee, & Lee, 2009). The most important languages for LPM are described in the following. A common language for documenting process models is the Event-Driven Process Chain (EPC) (G. Keller, Nüttgens, & Scheer, 1992; Richter-von Hagen & Stucky, 2004; Scheer, 2001; Staud, 2001). EPCs are a semi-formal notation for the enterprise and process modelling. They allow for the graphical documentation of the control flow, concurrency, conditional splits, joins, and loops, data flow, involved organization units, and information systems. The explicit documentation of the data flow however, is only provided if the control flow is equal to the data flow between these artefacts (Jablonski, Böhm, & Schulze, 1997). In detail, EPCs comprise functions as active components transforming objects from a start state into an end state and events as passive components revealing system states and business conditions. Furthermore, EPCs contain connecting operators, process interfaces, hierarchical functions, and further object types, such as external persons, organizational units, or swim lanes (Hoffmann, Kirsch, & Scheer, 1993; Klein, F., & A.-W., 2004; Richter-von Hagen & Stucky, 2004; Staud, 2001). EPCs do not comprise specific information for the execution of process models. In order to execute the EPC models, these have to be transformed into an executable process modelling language. For this thesis, EPCs are not considered as real alternative due to the lack of execution focus.

The most common languages that are related to process automation are the Yet Another Workflow Language (YAWL) (van der Aalst & ter Hofstede, 2005) and the Business Process Modelling Notation (BPMN) (OMG, 2006).

YAWL is a business process modelling language founded on workflow patterns and Petri nets (van der Aalst & ter Hofstede, 2005; van der Aalst, ter Hofstede, Kiepuszewski, & Barros, 2003). The goal of YAWL is to provide an intermediate language for various workflow systems. Hence, the foundation on Petri nets which allows for better performance for executing state-based workflow patterns. YAWL has a formal semantics and comprises a graphical syntax supporting higher-level modelling activities. YAWL is implemented in the so called YAWL System as described in van der Aalst et al. (van der Aalst, Aldred, Dumas, & ter Hofstede, 2004).

The main purpose of BPMN models is to facilitate the communication between domain analysts and the strategic decision-making (Jorg Becker, Rosemann, & Kugeler, 2003; J. C. Recker, Indulska, Rosemann, & Green, 2005). BPMN models are also used as a basis for specifying software system requirements and providing input to software development projects.

The modelling procedure in BPMN is performed with a small set of graphical elements, in particular, flow objects, connecting objects, swimlanes, and artefacts enabling the stakeholders to construct a Business Process Diagram (BPD). BPMN itself is not executable. To execute the process models the BPMN models have to be transformed into an executable language.

Executable process modelling languages

A commonly used executable process modelling language for service orchestration is the Web Service Business Process Execution Language (WS-BPEL) (OASIS, 2006). The services are integrated into the process model by referencing their Web Service Definition Language (WSDL) (Christensen, Curbera, Meredith, & Weerawarana, 2001) service descriptions. WS-BPEL combines the features of a block structure language with those for directed graphs. Both abstract and executable processes might be modelled in WS-BPEL. An abstract, non-executable process is a business protocol specifying the message exchange behaviour between different parties. The internal behaviour for the involved parties is not covered. An executable process specifies the execution order between a number of activities constituting the process, the partners involved in the process, the messages exchanged between these partners, and the fault and exception handling.

In order to make BPEL independent of the *Web Service Standards*[8] including WSDLs, an extension mechanism has been defined called BPEL-Light (Jörg Nitzsche, van Lessen, Karastoyanova, & Leymann, 2007b). Therefore, new elements have been defined replacing those elements that reference to WSDL-based elements or elements referencing WSDL interfaces. Another BPEL extension mechanism is BPEL4SWS (Jörg Nitzsche, van Lessen, Karastoyanova, & Leymann, 2007a). This BPEL dialect defines references to goals defined by the Web Service Modelling Ontology (WSMO) (Roman, Lausen, & Keller, 2006) and to interfaces described in OWL-S (W3C, 2004b).

2.4.2.2 Structuring Unstructured Processes

One objective of the LPMS is to increase the amount of explicitly structured processes. This helps organisations in managing effectively and efficiently their core competencies and the according support tasks. Unstructured processes mostly evolve

[8] See www.w3.org

around stakeholder communication, such as email exchange (Ukelson, 2009). Gartner (Michele Cantara et al., 2009) further details the definition:

"*Unstructured process describes work activities that are complex, nonroutine processes, predominantly executed by an individual or group highly dependent on the interpretation and judgment of the humans doing the work for their successful completion.*"

According to a Gartner Hype Cycle (Michele Cantara et al., 2009), the management of unstructured processes is one of the more important topics within the next 5-10 years. Common areas that need to be structured in terms of explicit processes comprise regulatory and compliance processes, exceptions and escalation processes, decision implementation processes, audit processes, and complex project management processes (Ukelson, 2009). Various approaches exist to structure unstructured processes, such as process mining or deriving process models from task structures. Stoitsev et al. (Stoitsev & Scheidl, 2008; Stoitsev et al., 2008) present an approach of deriving processes from task management. The authors propose to generate weakly-structured process models from data on personal task management. These process models are then transferred to process designers or software developers that transform the process models into executable models. The approach assumes the involvement of IT experts that execute the tasks. However, the LPMS envisages the direct enabling of business users to model and execute processes.

Unstructured processes might be made explicit through task management, mashups, or the LPMS

Another emerging approach of structuring unstructured processes is the generation of screenflows through mashup technologies (Benslimane, Dustdar, & Sheth, 2008; Hoyer et al., 2009; Hoyer & Stanoevska-Slabeva, 2009). The mashup technologies focus on the data flow between services and are complementary to a process modelling solution. An integration of mashup technologies into the LPMS is however not performed in this research work. Further, approaches exist using Artificial Intelligence (AI) technologies to derive processes (Blythe, Deelman, & Gil, 2003; Madhusudan, Zhao, & Marshall, 2004). The approach of Madhusudan et al. for example, uses task pre- and postconditions to develop plans to fulfil a business need. Later, these plans are transformed into workflow models.

2.4.3 Business Process Execution

As already introduced at the beginning of section 2.4, the automation of processes is subject to workflow modelling. Workflow Management Systems (WfMS) allow for the processing of workflows (Allen, 2001), Business Process Management Systems (BPMS) (Chang, 2005) are an evolution of WfMSs and allow for the definition and execution of processes by invoking underlying applications or services.

Traditional WfMSs have got a couple of characteristics that make the use in the service context difficult. Such characteristics comprise expensive software licenses, complex installation and operation, or long development cycles to automate real business processes (Alonso et al., 2004). Alonso et al. further criticize that WfMSs had to provide an according necessary runtime environment, such as a complete middleware platform. Thus, WfMSs became heavyweight platforms that had been difficult to operate and maintain. Lastly, WfMSs had been most useful with predefined processes in a specific structure that had often already been implemented by conventional middleware.

In the meantime, WfMSs had mostly been renamed to BPMSs and acquired flexibility. This thesis refers to the following definition of flexibility given by Regev et al. (Regev, Bider, & Wegmann, 2007): "*Flexibility is the capability to change without loss of identity*". Further, Regev et al. define business process flexibility as the capability of a process to adjust to environmental changes by adjusting only aspects of a process that are impacted by these changes while keeping the rest of the process stable.

However, the terms WfMS and BPMS are treated differently in literature. Whilst a couple of BPM experts don't see any difference (e.g. (Gadatsch, 2005)) others distinguish the scope of WfMSs and BPMSs (e.g. (Chang, 2005; J. B. Hill, Cantara, Kerremans, & Plummer, 2009)). Often, a BPMS is seen as an extension to a WfMS through providing functionality for process design and improvement besides the pure execution. Gartner (J. B. Hill et al., 2009) defines a BPMS as an "*integrated collection of software technologies that enables the control and management of business processes*".

Recently, web-based BPMSs avoiding heavy-weight installation and a complex runtime environment have emerged. These solutions are often offered as services and allow for the integration of services from the cloud. For example, Lombardi Blueprint offers a BPM solution for process design and documentation as a web-based tool (Lombardi-BPM, 2010; Richardson et al., 2009). The processes modelled through

Lombardi technology are in line with the Business Process Definition Metamodel (BPDM) (J. H. Frank, Gardner, Johnston, White, & Iyengar, 2004; OMG, 2004) and might be transformed into an executable process modelling language. Furthermore, Appian and Cordys offer technologies for process modelling and execution as a service (Appian, 2010; Cordys, 2010). Both use BPMN as modelling language, however, requiring the user to have both business and IT skills.

An environment for process execution requires a set of modelling abstractions that support potential system failures. Alonso et al. (Alonso et al., 2004) present forward recovery, backward recovery, exception-handling languages, and deadlines as means to support system failures. Since this thesis focuses on the holistic design of the LPMS with a special emphasize on the language, the process execution environment is not described in detail.

2.5 SEMANTIC TECHNOLOGIES

In this section, semantic technologies and their application to the web and to services are described. Further, semantic annotations are introduced that describe services from a semantic perspective. For this thesis the following definition of semantic annotations is valid: *Semantic annotations are well-structured and categorized metadata that describe an artefact*. The semantic annotations support the discovery and composition of services in processes.

Key requirements
- The LPMS uses semantic annotations for service discovery and composition in processes - The according process modelling language has to provide elements to handle these semantic annotations.

In order to interoperate with web services, the service provider and user agree on a common interface that is in most cases decided by the provider. This interface provides information about messages to be exchanged. The service user then has to understand the interface description given by the provider. For the syntactical interface description commonly agreed standards exist, such as XML Schema and WSDL. XML schema describes the message types while WSDL specifies the operations to exchange messages and detailed message serialization information. However, the syntactic information is not sufficiently precise to always ensure a common understanding of how the service works and what the service details mean. In particular, for a machine-to-machine communication the syntactic descriptions are not sufficient. Furthermore, the human users are often not capable to give precise syntactic descriptions of the services. Existing semantic descriptions are mostly unstructured on web sites describing the web services.

Semantic annotations

An approach to bridge the gap between business specification and execution details, to support the provisioning of execution-related information for services, and to enable a machine-to-machine communication is to attach semantic annotations to process elements. These annotations provide information about discovery, composition, and execution of services. Furthermore, semantic activity descriptions allow for the definition of requirements rather than specifying concrete services. These annotations are thus more intuitively understandable for business users. In this thesis, this approach of using semantic annotations is followed. Rolland et al. (Rolland, Kaabi, & Kraiem, 2007; Rolland & Kaabi, 2007) have already described an approach of defining intuitive semantic service descriptions rather than technically oriented functional descriptions. The service composition however, is defined in a declarative way. In this thesis, the semantic service descriptions are used to describe activities that are part of a well-defined process-order. Current process modelling languages, e.g. BPMN, allow only for the rudimentary provisioning of semantic annotations through text annotations.

Currently, REST and WSDL/SOAP services are wide-spread syntactic description standards for service interfaces. In order to add semantic information, the interfaces of REST services might be described by hRESTS (HTML for RESTful services) (Kopecky, Vitvar et al. 2009) in order to ensure machine-readability. Like WSDL, hRESTS is a service description language and structures the information that is mostly provided in an unstructured manner in HTML on web sites. A description language for hRESTS can as well be found in Kopecky et al. hRESTS uses tags to mark information relevant for the service API, such as tags for the entire service description, for the HTTP methods, the operations, or the input and output.

In order to integrate semantic annotations in hRESTS descriptions, MicroWSMO has been defined (Kopecky, Vitvar et al. 2009). MicroWSMO defines annotations for hRESTS as SAWSDL annotations do for WSDL (W3C, 2007a). MicroWSMO annotations are based on the WSMO-Lite ontology that is described below. This ensures that the annotations of RESTful services and WSDL/SOAP services can be mapped.

Ontologies

Metadata provide semantic information for resources they are attached to. In order to structure this metadata, ontologies provide a schema for the metadata. This thesis refers to Gruber (Gruber, 1992) for a definition of ontologies: "*An ontology is a specification of a conceptualization*". Gruber details this statement as follows. An ontology is a formal specification of the objects, concepts, other entities, and relationships that might exist for an agent or a community of agents. In the context of AI and as well in the context of BPM, the ontology is a set of representational terms. Definitions associate the names of entities in the common knowledge space, such as classes, relations, functions, or other objects, with human-readable text explaining the meaning of the text. Furthermore, formal axioms are associated that constrain the interpretation and use of these terms. In other words, according to Gruber, an ontology is the statement of a logical theory. Hence, by basing semantic annotations on an ontology, a common understanding is ensured.

An existing ontology and hence, a conceptual model for semantically describing Web Services is WSMO. WSMO provides information about functional and behavioural aspects of web services. It is based on techniques on logics and knowledge representation. Service *capabilities* describe aspects for service discovery and composition, while *interface* and *choreography* information specify how to bind a discovered service.

Two ontologies for WSDL-based services, WSMO-Lite (Vitvar, Kopecky, & Fensel, 2008), and for REST services (Fielding, 2000), MicroWSMO (Kopecky, Vitvar, Fensel, & Gomadam, 2009), are complementary to WSMO. The Business Process Modelling Ontology (BPMO) (Dimitrov, Simov, Stein, & Konstantinov, 2007) provides a framework for describing processes by ontologies. BPMO is based on WSMO. However, BPMO doesn't support reasoning about state changes and temporal behaviour, variable semantics are neither supported. Parts of the full-fledged BPMO framework are used in order to attach semantic concepts to the LPML. In order to keep the LPML really lightweight, semantic annotations are used mainly for activities and the process as a whole.

Grounding

The semantic information of SWS doesn't contain any invocation information. This information is given through WSDL, XML, or REST data and has to be figured out through a grounding mechanism (Kopecky, Moran, Vitvar, Roman, & Mocan, 2007). The grounding is implemented in order to transform the invocation-relevant information from a semantic description language into XML and WSDL, respectively REST. The syntactic service descriptions in terms of XML and WSDL and the semantic descriptions in terms of WSMO or in another language respectively in REST and MicroWSMO are linked. A corresponding lowering (from an ontology description language to XML-based data) and a lifting mechanism (from XML-based data to an ontology description language) implement the grounding. The lifting and lowering schemas have to be predefined.

It is assumed that existing service descriptions in WSDL contain references to semantic descriptions through SAWSDL annotations (W3C, 2007a). Further, the information contained in WSDL and XML might be lifted in order to be described in a semantic annotation language, such as the Resource Description Framework (RDF) (W3C, 2004c). In SAWSDL, the *modelReference* attribute references domain-specific ontologies. The lifting mechanism is implemented through the *liftingSchemaMapping* attribute and the lowering mechanism through the *loweringSchemaMapping* attribute.

Various notations exist to formalize ontologies, such as RDF, RDFS (W3C, 2004d), OWL (W3C, 2004a), and WSML (WSMO-Working-Group, 2008). RDF represents entities and binary relationships between entities. The entities might be identified by a unique identifier. Two entities and a relationship between them are called a triple in RDF. To visualize an RDF structure, the source entity of the relationship might be represented as subject, the relationship as predicate, and the target entity as object.

RDFS defines the primitives to describe classes, instances, and relationships and restricts RDF to a formal notion of meaning. This might be seen as the part of RDF that is common to all other accounts of meaning.

Reasoning

Often, the explicitly given semantic descriptions are not sufficient to gather all required information. Reasonners have been developed to overcome this issue and derive some information from existing knowledge. Reasoning is based on a formal, logical system and uses the meaning of facts. An example for such a logical system is the description logics. Two main categories of reasoning exist, the deductive and inductive reasoning. Deductive reasoning covers aspects that follow from given facts. Inductive reasoning covers deriving a reliable generalization from existing observations. Reasoning is closely related to semantic technologies and hence to semantic annotations of services used in this thesis. The user however, doesn’t get in touch with reasoning functionalities. Reasoning works in the background according to decidability and complexity constraints.

Reasoning for BPM

The reasoning for BPM is mostly based on AI planning technologies. An overview of AI planning can be found in Russell and Norvig (S. J. Russell & Norvig, 2003). AI planning has as well been used for designing software at higher abstraction levels, such as process modelling (Linden, 1991). Blythe et al. (Blythe et al., 2003) have proposed an approach using AI planning for workflow management. According to Blythe et al., a declarative specification of a process model might be seen as a plan to fulfil a business need. Madhusudan et al. (Madhusudan et al., 2004) propose an approach to reason about interactions between pre- and postconditions of tasks to develop plans that are equivalent to process models.

Semantic Execution Environment (SEE)

In order to process SWS, Semantic Execution Environments (SEE) have been specified. The SEEs support functionalities for the (semi-)automated discovery, selection, composition, mediation, execution, and monitoring of SWS (Pedrinaci, Grenon, Galizia, Gugliotta, & Domingue, 2010). The current approaches on SEEs are closely coupled to WSMO forming the theoretical model. Two reference implementations currently exist, WSMX (Cimpian, Moran, Oren, Vitvar, & Zaremba, 2005) and IRS-III (IRS, 2010).

Orchestration of SWS

Barricelli et al. describe an approach to create workflows through the semantic orchestration of web services. (Barricelli et al., 2010). The authors describe how a common understanding of business and IT experts might be supported through semantic descriptions of services. Furthermore, in their work, there has been created an editor visualizing the workflows and the according semantic annotations. In this thesis, a similar approach about process modelling is proposed. However, the goal is to directly enable the business users to model executable processes themselves instead of making their process knowledge explicit to be used by IT developers.

Another approach using case-based reasoning has been proposed by Madhusudan et al. (Madhusudan et al., 2004). This approach uses text query mechanisms to find appropriate cases from a repository. Semantic technologies are used to adjust, modify, or compose existing cases to fulfil different requirements. The cases are described by preconditions, postconditions, and applicable tasks. The described approach focuses on case representation and retrieval. The composition of cases or included tasks however, is not yet investigated.

To conclude, the existing semantic technologies are sufficient to achieve the goals of the LPMS. In particular, WSMO and its related ontologies, WSMO-Lite and Micro-WSMO provide means to support the instantiation of process steps with services through discovery and selection functionalities.

3 Problem Statement, Requirements, And Approach

After having introduced basic theories, methodologies, and technologies, this section starts with the problem statement in section 3.1. In section 3.2, the requirements for the envisaged solution approach are presented that are derived from the problem statement. Afterwards, the LPM approach is presented in section 3.3. This approach describes how the business user might create executable process models without getting in touch with execution details. Section 3.4 is dedicated to the research context of this thesis that is the research project SOA4All (SOA4All, 2010). SOA4All not only embeds the LPM topic but implements as well important technologies that are necessary for the functioning of LPM. The section closes with a summary of the fulfilment of the requirements for the LPM approach and in particular, for the LPML through existing approaches in section 3.5.

According to the Design Science guidelines, this section covers again the Research Contribution guideline. In terms of the Design Science phases, the phase of the need identification is covered.

3.1 Problem Statement

To model and execute processes multiple languages and technologies exist. On the one hand, languages exist to design processes for business purposes. These process models are created by business users mostly. On the other hand, languages and technologies exist for process execution. These languages are mostly too complicated for the average business user. A lack of a coherent approach allowing business users to model and execute processes has been identified.

The creation of executable processes might be seen as application development. The means and tools to create those processes are applications that allow for the development of other applications and that have to fit to the users' skills. The problems that are described in the following, are categorized according to their affiliation to application development, application use, limitation of Information Technology (IT) resources, and the cognitive gap between business and IT. These categories apply in general to the support of the business through IT. In particular, for process modelling and execution these issues occur which is described after the introduction of the problem categories.

Application development

Systems and applications have to adapt to environments that change often and in short timeframes. Applications have to avoid long release cycles and build the capability to adjust quickly to those changing environments (Lieberman, Paterno, & Wulf, 2006). However, currently, application adaptations are often done in long release cycles. Systems cannot easily be tailored for heterogeneous user groups. And, a trend appears revealing that software development is more and more shifted from traditional application development to composition of existing pieces of software. In particular, applications that implement a process are more and more compositions of existing software functionalities. However, systems allowing for the coherent process and application composition from the business to the execution level do rarely exist.

Application use

An IT system should be tailored for various user groups according to a level of complexity that meets their skills (MacLean, Carter, Loevstrand, & Moran, 1990). The better the system functionality fits to users' needs, the better the users are satisfied. Since users' needs are often heterogeneous, the fit is a trade-off between generality and specialisation. Sutcliffe (A. Sutcliffe, 2005) summarises that as follows: "*The user motivation to accept an end-user development technology will be inversely proportional to product complexity and variability in the user population.*" Furthermore, for cost and time reasons, the adaptation procedure should be done by the business user that needs that kind of change in order to optimally meet the user requirements (Lieberman et al., 2006; A. Sutcliffe, 2005). Hereby, the knowledge of a large number of business users should be made explicit through user-friendly building blocks and interfaces, such as Mashups (Hoyer & Stanoevska-Slabeva, 2009). Current applications mostly address only one user group requiring specific usage skills.

Limitation of IT resources

Software development and adaptations are often done by the IT department which is time consuming and costly. The large amount of business users is not enabled to create software or to support at least the software development process. Due to the lack of appropriate means for software development dedicated to business users, the creative energy of those business users is not used. In particular, for execution-oriented process modelling, the IT department has to be involved mostly.

Cognitive gap between business and IT

Table 5: Problem statement in the context of BPM

Problem statement	**Application to BPM**
Application development	Existing business process modelling stacks comprising a graphical abstraction for business users and a formal, executable format require manual, sophisticated, and error-prone transformation mechanisms.
Application use	Current BPM suites and tools either target business users or IT experts
Limitation of IT resources	- Business process models, in particular executable process models, are developed by the IT department. - The transformation from an abstract process model into an executable model is performed by IT experts
Cognitive gap between business and IT	- Current BPM solutions for process execution – both languages and suites – require a high level of expertise in business and IT. - Business process modelling languages contain elements that are not easily understandable for business users. - There’s no common understanding of business process models and the terminology used. - Modelling errors occur. - Due to the lack of common understanding reuse of process models is low. - The business knowledge is not made explicit in process models, the wisdom of the crowds is not used.

Still, misunderstandings might occur between business people and IT experts developing and maintaining applications. Malone calls this the “*cognitive distance*” between designing and using an application (Malone, Lai, & Fry, 1992). While the development of an application requires knowledge in formal theories and models, the simple use requires a much lower level of formality (Morch & Mehandjiev, 2000). Hence, people that have got the business knowledge often lack the skills to develop an according application or at least to communicate on a formal level. This produces a gap between the business specification and the application development. Overall, there’s often a bad fit between systems and the requirements of the users leading to low user satisfaction (A. Sutcliffe, 2005).

These general issues occur as well in the area of Business Process Management. Table 5 reveals the problem statements applied to the BPM context.

3.2 REQUIREMENTS

The goal of this thesis is to provide a design principles framework, the design of a language, and of tool specifications allowing the business user to model and execute processes. This results in the objective to design the LPMS and in particular, the LPML. The target group is thus people that have computational needs but limited IT knowledge and no interest in becoming an IT professional (Nardi, 1993; Quinn, 2005).

The requirements to the LPMS might be categorized into technical, individual, organisational, and economic requirements. Technical requirements refer to the technical design of the LPMS. Individual requirements focus on the single user whilst organisational requirements concentrate on the interplay of the users. The economic requirements cover the economic perspective on the LPMS. Table 6 assigns these requirement categories to the problem statement.

Table 6: Relation of requirement categories to the problem statement

Relation of requirement categories to the problem statement	**Requirements**			
	Technical	Individual	Organisa-tional	Economic
Application development	●			
Application use		●	●	
Limitation of IT resources			●	●
Cognitive gap between business and IT	●	●	●	●

Table 7, Table 8, Table 9, and Table 10 present the requirements according to the category. The column “source” at the right hand side of these tables assigns the requirements to their relation to the problem statement. General requirements are applied as well to all artefacts in the context of a modelling solution. The general requirements to a modelling solution are based on literature on conceptual models, reference models, modelling languages and methods, and domain-specific languages. A summary of these requirements can be found in literature (Kurpjuweit, 2009). This thesis mainly requires aspects described for modelling languages and methods that are based on literature (Brinkkemper, Saeki, & Harmsen, 1999; U. Frank, 1998; Greiffenberg, 2003; Kiper, Howard, & Ames, 1997; Paige, Ostroff, & Brooke, 2000; Remme, 1997; Sinz, 1998; Zelewski, 1996). These aspects cover syntactic and semantic correctness, consistency, suitable abstraction and granularity, simplicity, understandability, stability, longevity, cost-effectiveness, and transformability. These requirements are integrated into the following description of the requirements for the LPMS.

From a business and technical perspective, the LPMS should mainly comprise a coherent design process of creating executable process models. Further, it should be capable of integrating heterogeneous services and of handling semantic annotations. On an individual level, the main requirement is to provide process modelling functionality for a broader user-base. Hence, the LPMS should be designed according to the principles of usability and simplicity. The LPMS should provide various abstraction views on process models allowing for keeping the business user free from execution details, such as service binding and composition. The main requirements on organisational level cover the suitability of the LPMS to be used to perform the users' tasks and the fit to the existing IT infrastructure and to the organisation. The economic requirements concern mainly cost savings, freeing IT experts from business modelling tasks, and reusing the LPM infrastructure as well as process model parts. Enabling business users to model and execute their processes themselves reduces overall modelling costs by decreasing the workload for expensive IT experts. By providing process information in a language the user might understand, the degree of reuse of process models increases.

Table 7: Technical requirements for the LPMS

Requirement	**Source**			
	Application development	**Application use**	**Limitation of IT resources**	**Cognitive gap between business and**
Design of a consistent BPM language stack comprising a graphical abstraction and an executable layer for process models. The abstract layer has to provide sufficient	●			

information for service discovery and selection. The executable layer has to provide sufficient information for service selection and composition.				
Definition of a consistent design process creating an executable process model in various steps. For each step the actions, the needed input, the output, and the performer have to be aligned.	●			
Handling of semantic information: Semantic informations provide a common understanding between users and users, users and machines, and machines and machines.				●
Integration of heterogeneous services	●			
Service selection, binding, replacement, and adaptation at various stages (design time, runtime, and in-between)	●			

Table 8: Individual requirements for the LPMS

Requirement	Source			
	Application development	Application use	Limitation of IT resources	Cognitive gap between business and
Provide executable BPM functionality suitable for business users that fits into the existing environment. The functionality requirement is based on (Hevner et al., 2004) and (Moody & Shanks, 1994)		●		
Increase number of users by people without IT knowledge: Current existing BPM solutions address IT experts.		●		
Usability, simplicity and understandability: Provide a simple and understandable BPM solution for business users. Further, usability, simplicity, and understandability are general requirements to software solutions (U. Frank, 1998; March & Smith, 1995; Paige et al., 2000).		●		●
Provide various abstraction views in order to keep the user free from execution details. Generally, a modelling		●		

language should comprise several abstraction layers in order to be usable by heterogeneous user groups (Paige et al., 2000).				
Facilitate information search: Current process models and its elements are often not unambiguous.		●		
Facilitate taking process ownership: Avoid that users hesitate of taking the process ownership due to a lack of understanding.		●		

Table 9: Organisational requirements for the LPMS

Requirement	Source			
	Application development	Application use	Limitation of IT resources	Cognitive gap between business and IT
Increase the number of users according to their task duties: Keep modelling experts and IT experts free from performing modelling tasks for business users.		●	●	●
Achieve community acceptance and suitability for collaborative modelling: In order to foster the use of the wisdom of the crowds, the LPMS has to be accepted by the modelling community.		●	●	●

Table 10: Economic requirements for the LPMS

Requirement	Source			
	Application development	Application use	Limitation of IT resources	Cognitive gap between business and
Generality and applicability to various scenarios: In order to be reused, the LPMS has to be applicable to various settings. In general, a software solution has to be designed to achieve generality (March & Smith, 1995).		●		
Increase degree of reuse: Foster reuse of process models and its parts in order to avoid modelling from scratch.		●	●	
Reduce modelling time: Reduce modelling time through an intuitively understandable modelling language, tool support, and reuse of models and its parts.		●	●	●
Lower needed expertise and training effort		●	●	●
Utilization of existing infrastructure and software assets: Integrate existing assets in order to avoid costly procurements (Palmer, 2009). In the LPM setting, this means the ability to integrate heterogeneous services.		●	●	
Use of internal development and support resources and avoiding high workload for IT resources: Enable business users to model and execute processes themselves. Further, establish problem solving mechanisms for business users. This will free resources of the IT department to be spent for other initiatives (Palmer, 2009).		●	●	
Faster time to market: Achieve faster time to market through quicker process modelling, deployment, and execution (Palmer, 2009).			●	
Lower initial application integration costs: Easy integration of existing assets (Palmer, 2009).			●	

The requirements to the LPML cover correctness, completeness, and expressivity, adaptability and extensibility, efficiency and effectiveness, and usability. Table 11 presents the requirement according to the categories. As part of the usability requirements, different abstraction views are an important aspect for freeing users from execution details. The various LPML views however, should be based on a common canonical format of the process models in order to guarantee a model coherency without sophisticated transformation mechanisms and potential loss of information. Graphical symbols on the abstract level should be simple. The LPML should be further extendable in order to adjust to specific domains or environments. Furthermore, the integration of metadata should be allowed in order to provide machine-readable documentation. Other general requirements to modelling languages, such as stability and longevity, are not considered important in this thesis describing the design of the LPML.

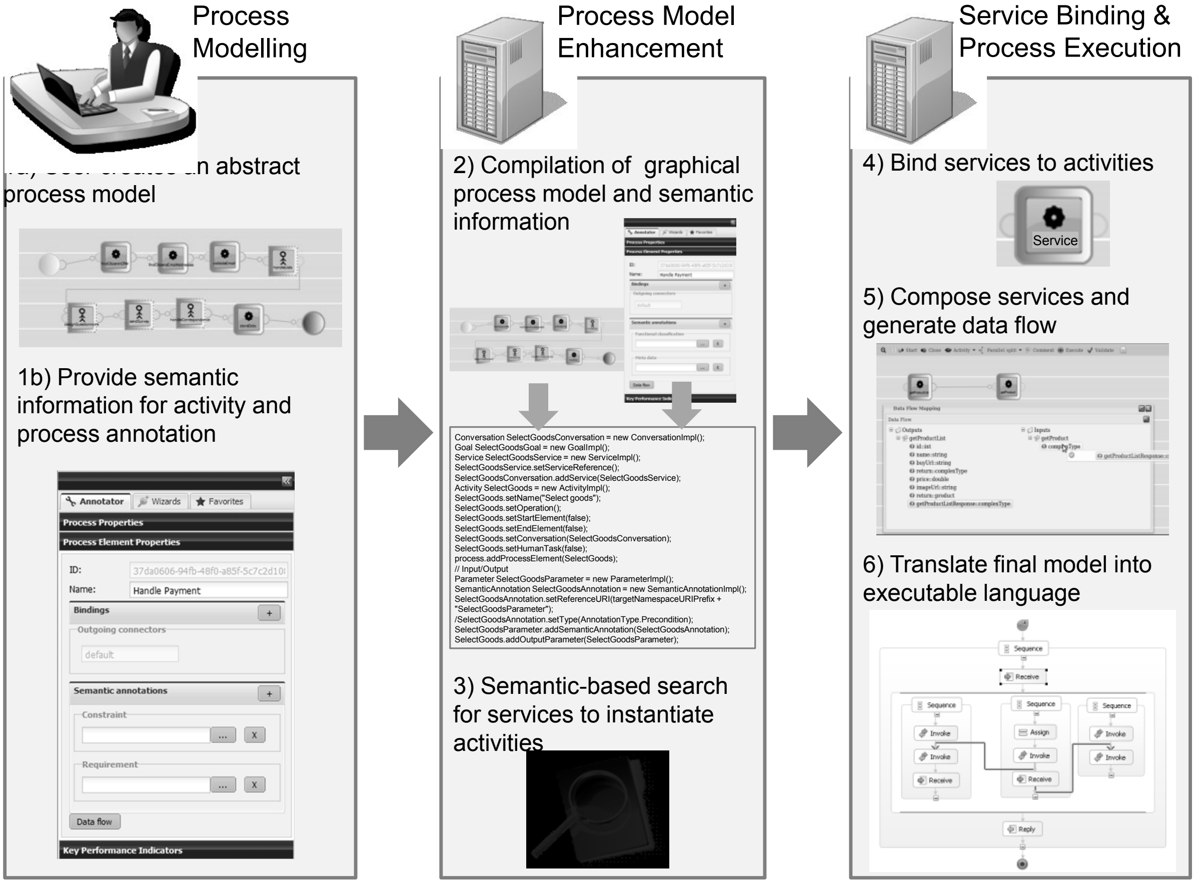
Table 11: Requirements for the LPML

Requirement category	Requirement	Source
Correctness	Syntactic correctness	Literature
	Semantic correctness	Literature
	Uniqueness and canonical, exchangeable format	Literature
	Coherency of different layers	Literature
Completeness and expressiveness	Ontological completeness	LPM
	Pattern-based completeness	LPM
	Use case scenario coverage	LPM
Adaptability and Extensibility	Optional extensions and adaptations	Literature

3.3 APPROACH

In this thesis, LPM is introduced seeking to lower the entrance barrier for business users to model processes. It is investigated how business users might be enabled to express activity requirements rather than to specify services and be kept free from execution details. Figure 1 in section 1.1 depicts an application scenario how the user is supported to perform a task. The user expresses the task to be fulfilled in terms of a process model. Afterwards, services that might be bound to the steps in the process model are automatically retrieved. Lastly, an executable process is created through composing the retrieved services.

In order to realize such scenarios, the process models created by the user have to contain sufficient information in order to allow for automatic search for services and composition of those services. The process models have to be enhanced step by step to generate executable process models. In the following, these required enhancement steps are described. Hereby, actions the user has to perform and actions that are automatically performed by tools are differentiated.


Process Modelling
1a) User creates an abstract process model
1b) Provide semantic information for activity and process annotation
Annotator
Wizards
Favorites
Process Properties
Process Element Properties
ID:
37da0606-94fb-48f0-a85f-5c7c2d10
Name:
Handle Payment
Bindings
Outgoing connectors
default
Semantic annotations
Constraint
Requirement
Data flow
Key Performance Indicators
Process Model Enhancement
2) Compilation of graphical process model and semantic information
3) Semantic-based search for services to instantiate activities
Service Binding & Process Execution
4) Bind services to activities
Service
5) Compose services and generate data flow
6) Translate final model into executable language
Sequence
Receive
Invoke
Assign
Reply

```
Conversation SelectGoodsConversation = new ConversationImpl();
Goal SelectGoodsGoal = new GoalImpl();
Service SelectGoodsService = new ServiceImpl();
SelectGoodsService.setServiceReference();
SelectGoodsConversation.addService(SelectGoodsService);
Activity SelectGoods = new ActivityImpl();
SelectGoods.setName("Select goods");
SelectGoods.setOperation();
SelectGoods.setStartElement(false);
SelectGoods.setEndElement(false);
SelectGoods.setConversation(SelectGoodsConversation);
SelectGoods.setHumanTask(false);
process.addProcessElement(SelectGoods);
// Input/Output
Parameter SelectGoodsParameter = new ParameterImpl();
SemanticAnnotation SelectGoodsAnnotation = new SemanticAnnotationImpl();
SelectGoodsAnnotation.setReferenceURI(targetNamespaceURIPrefix +
"SelectGoodsParameter");
/SelectGoodsAnnotation.setType(AnnotationType.Precondition);
SelectGoodsParameter.addSemanticAnnotation(SelectGoodsAnnotation);
SelectGoods.addOutputParameter(SelectGoodsParameter);
```

Figure 4: LPM approach to create an executable process model

Gil (Gil, 2006) proposes an approach comprising three steps to create executable workflows using templates. The first step is to define workflow templates that are data- and execution-independent specifications of computations. In the following second step, workflow instances are created specifying the data needed on an activity level. In addition, the data flow is specified. However, the second step is still execution-independent. In the final step, the executable workflows are created by assigning resources that exist in the execution environment.

In this thesis, a similar approach is followed. The business user defines a series of activities, the functionality of each activity, and the control-flow between them. In the following, the further steps needed in order to instantiate the activities with services and hence, make the processes executable, is presented. This procedure is performed in six steps as depicted in Figure 4 and described in Table 12.

The following steps have to be performed to achieve an executable process model:

Table 12: Main steps to create an executable process model through the LPM approach

Step	Activity	Needed artefacts	Created and described in
1a	Graphically modelling processes in a language the business user understands.	A language for LPM integrating semantic annotations	See section 4 and 5.1 of this thesis
1b	Providing semantic information for activity and gateway specifications	A language for LPM integrating semantic annotations. Means to support the user in providing semantic annotations, e.g. through a separate data area in the process editor, templates to fill, existing requirements and constraints to click on	See section 4 and 5.1 of this thesis
2	A Process Editor compiles and transforms the graphical model into a textual model	Functionality compiling the graphical model into a textual model	Implemented in SOA4All and summarized in section 5.2.2
	The activity and gateway specifications given by the user and context information are compiled into formal, ontology-based semantic annotations for functional classification, preconditions, postconditions, non-functional properties.	Common ontologies to be referenced by semantic annotations	Implemented in SOA4All and summarized in section 3.4.2
3	Search for services to instantiate the activities	Semantically described services Common ontologies to be referenced by semantic annotations A public service repository A service search engine	Implemented in SOA4All and summarized in section 3.4.2

4	Select services based on semantic annotations and preferences	Functionality to select services from results	Implemented in SOA4All and summarized in section 5.2.3
5	Compose services and generate data flow	Component to map the data flow	Implemented in SOA4All and summarized in section 5.2.3
6	Translation of the process model into an executable language, such as BPEL	Process translation and execution engine for LPML processes	Implemented in SOA4All and summarized in section 5.2.4

To summarize, the contribution of this thesis to the steps of creating an executable process model covers the approach as a whole and the process modelling language, the LPML.

Those parts that are prerequisites for the LPM approach are described in the following section 3.4. Especially, section 3.4.2 describes the parts that are developed in the context of the project SOA4All.

3.4 Research Context

3.4.1 The Research Project SOA4All

This thesis has been created in the context of the research project SOA4All. In the following, SOA4All is introduced and revealed how this thesis is integrated into the research work.

The FP7 research project SOA4All funded by the European Commission "*aims at realizing a world where billions of parties are exposing and consuming services via advanced Web technology: the main objective of the project is to provide a comprehensive framework that integrates complementary and evolutionary technical advances (i.e., SOA, context management, Web principles, Web 2.0 and semantic technologies) into a coherent and domain-independent service delivery platform*" (SOA4All, 2010).

The project is based on the following main building blocks, taken from the SOA4All web site (SOA4All, 2010):

- SOA as the emerging dominant paradigm for application development which abstracts from software to the notion of service.
- Context management, i.e., adapting services to meet local environmental constraints, organizational policies and personal preferences.
- Web principles to scale SOA to a world-wide Web communications infrastructure.
- Web 2.0 as a means to structure human-machine cooperation in an efficient and cost-effective manner.
- Semantic Web technologies to automate service discovery, mediation and composition.

The main research objectives of SOA4All are as follows:

- Definition and implementation of a service web architecture to bring web services and SOA to a web scale. This comprises a framework for defining an architecture for web services, approaches for grounding semantic web services (SWS) into existing syntactical description standards and protocols, a service bus as infrastructural backbone, and a test-bed to validate the research results.
- Specification and implementation of a user interface allowing business users to manage web services. The user interface blurs the differentiation between service providers and consumers by providing methods and functionality for the provisioning, composition, bundling, consumption, and analysis of services.
- Provisioning of service annotation and reasoning functionality. This allows for improving service discovery and composition by using semantics to reach a common understanding of service descriptions. In more detail, the objective comprises the specification of lightweight versions of existing semantic description frameworks, a scalable reasoning system, and an ontology instantiation and mapping to ontology tag clouds.
- Specification and implementation of a framework for managing and applying context information to adapt services and processes.
- Definition and implementation of a service discovery functionality allowing for the retrieval of services that fulfil the users' requirements. This comprises a service crawler to collect service information that is distributed in the web, a service discovery tool to find the appropriate services, and an RDF access to crawled data.
- Definition and implementation of tools enabling automatic service and process construction. This comprises the definition of a lightweight process modelling language and tools allowing the business user to model and execute processes based on service compositions.

In addition, SOA4All targets at validating the research objectives in use cases as business proof of concept. SOA4All aims as well to contribute to standardization efforts and train business users to benefit from the developed concepts and tools.

This thesis has been created in the context of task 6.3 of the work package 6, Service Construction, of the project SOA4All. This work package covers the creation of a modelling language and tools for process model adaptation and execution allowing business users to model and execute processes. In particular, the task 6.3 deals with the specification of a "*Lightweight, Context-aware Process Modelling Language*". This language specification has resulted in the LPML which is the main subject to this thesis. I lead the task 6.3 in the SOA4All project and have been the architect and responsible of the LPML. Hence, my contributions had been the specification of the principles for lightweight process modelling, the LPML metamodel comprising all LPML elements, the design process to model and execute a process, and the evaluation of these research results. I contributed as well to the tools for process modelling, adaptation and execution, however, the main work for the tools had been done by project partners. I had further been responsible for applying the research results related to lightweight process modelling to the public sector use case. The use case itself had been specified in another work package in the SOA4All project.

3.4.2 Integration into SOA4All

This section describes the artefacts that are needed for the functioning of the LPM approach and that have been developed in the context of the SOA4All project. Those artefacts that are part of the LPM approach and have been developed in SOA4All are described in section 5.2.

Ontologies:

A prerequisite for the functioning of LPM, and hence the steps 2 and 3 as described in section 3.3, is that services not only are semantically described but that the semantic annotations refer to ontologies as well. And again a prerequisite is the existence of a global ontology or the existence of multiple ontologies that can be mapped to each other.

Currently, most of the services either lack completely a semantic description or lack a reference to a common ontology. The research objectives in SOA4All address these shortcomings by providing means to facilitate the semantic description of services and by defining ontologies the services might refer to.

In SOA4All, a couple of ontologies have been defined. These ontologies are described on the SOA4All website (SOA4All, 2010) and summarized in the following.

- SOA4All Functional Classifications ontology. This ontology defines concepts for the functional classification of services. It comprises two parts, namely a formal specification of the NEXOF-RA functional view and SOA4All-specific extensions that cover functionalities for tools defined in SOA4All and processing semantic information.
- SOA4All Execution ontology. This ontology is a set of ontologies representing concepts for annotating SOA4All web services with execution-related information.
- Further ontologies for auditing logs that supports recommendations for the users, for describing contextual information (applying contextual information to LPM is described in section 4.4), or for eGovernment (see section 1 for an eGovernment scenario for LPM).

Besides the SOA4All ontologies, general ontologies for describing services exist, such as WSMO, WSMO-Lite, Micro-WSMO, or hRESTS (see section 2.5 for more information on these ontologies). Further, the ontology yellow pages[9] list existing ontologies of other areas.

Semantically described services:

Another prerequisite for step 3 of the LPM approach (see section 3.3) is the existence of semantically described services that are publicly available. Currently, most of the services are only described syntactically through WSDL/SOAP descriptions. Semantic descriptions of the web services on the web sites are mostly unstructured. As introduced in section 2.5, semantic description formats exist to structure the semantic information. In order to increase the number of semantically described services, in the project SOA4All, editors for service providers to describe their services have been developed. SWEET (Semantic Web sErvice Editing Tool) is an editor for supporting the semantic annotation of web APIs and RESTful services. A description can be

[9] See http://wg.sti2.org/semtech-onto/index.php/The_Ontology_Yellow_Pages

found in Maleshkova et al. (Maleshkova, Pedrinaci, & Domingue, 2010). We can expect that the amount of semantically annotated services will increase significantly through tools, such as SWEET.

Service repository

In order to make the services publicly available, a service repository has to exist. In the context of the project SOA4All, IServe[10] has been developed as a global service repository. IServe is able to deal with heterogeneous annotation mechanisms, such as SAWSDL, WSMO-Lite, MicroWSMO, and OWL-S. The repository therefore abstracts these annotation mechanisms through the use of a Minimal Service Model that describes services in RDF. IServe provides a functionality to automatically generate the RDF statements based on the Minimal Service Model. These RDF statements are exposed as linked data. Hence, semantic web services might be published in an interoperable format.

Service discovery

Besides browsing or searching in a service repository, web services might be searched in the web. An existing web service search engine[11] is provided by SEEKDA. Currently, more than 28000 WSDL-described services from more than 7000 providers are crawled, indexed, monitored, and categorized. Seekda allows for the search by tags, keywords, and various categories, such as country or provider.

Reasoning

The main feature of a reasoner in the context of LPM is the resolution of the semantic annotations. The resolution of the semantic annotations is needed for binding goals and services to activities and for composing services in terms of data flow mappings. The reasoning has to be supported for various ontologies, such as WSMO, WSMO-Lite, Micro-WSMO, or domain-specific ontologies. Furthermore, the reasoning has to support various representation formats. In the context of SOA4All, various reasoners

[10] See (http://iserve.kmi.open.ac.uk/) for more information

[11] See (http://webservices.seekda.com/browse) for more inhe formation

have been developed. For example, IRIS[12] is an extended reasoning engine for rule-based languages, ELLY[13] is a reasoner for entailment and satisfiability checking of ELP knowledge bases, and WSML2[14] is a modular framework comprising validation, normalization, and transformation algorithms allowing for translating ontology descriptions in WSML to underlying reasoning engines (SOA4All, 2010).

3.4.3 Implementation of the LPM Design

Besides the LPM design, SOA4All comprises as well the implementation of the LPMS. This thesis has been written in the early design phase of the SOA4All project and describes the LPM design as the first research step. The following step, the prototypical implementation of the LPM design in terms of the LPMS, has been performed later in the project. At the point in time when the thesis had been written, the prototype hasn't been implemented yet and hence, there's no information about its realization. However, the envisaged prototypical implementation reveals the relevance and usability of the LPM design.

3.5 Fulfilment of Requirements and Conclusion

In section 2, basic theories, concepts, and state-of-the-art technologies have been presented. For the LPM approach, the following Table 13, Table 14, Table 15, and Table 16 in section 3.5.1 indicate for each requirement the fulfilment through state-of-the-art (SOTA) technologies. The column on the right hand side indicates the contribution to the research community according to the design science guideline (Hevner et al., 2004). Section 3.5.2 covers the fulfilment of LPML requirements through existing languages and technologies.

[12] See http://iris-reasoner.org/ for more information

[13] See http://elly.sourceforge.net/ for more information

[14] See http://tools.sti-innsbruck.at/wsml2reasoner/ for more information

3.5.1 Fulfilment of the LPM Approach Requirements

Table 13: Fulfillment of technical requirements through SOTA

Requirement	Fulfillment through SOTA	Contribution to the research community
Design of a consistent BPM language stack	Not yet covered by existing approaches	Creation of a new, consistent BPM language stack comprising an abstract graphical layer and a canonical format.
Definition of a consistent design process	Not yet covered by existing approaches	Definition of a new design process specifying how to generate an executable process out of abstract semantic service and process information. The design process further indicates how to free the user from providing execution information through new principles, such as abstraction and the usage of semantic annotations.
Handling of semantic annotations	Reuse and adjust concepts of semantic technologies	Reuse of approaches applying semantic annotations to entities in process modelling. The new thing is the definition of semantic annotations that might be handled by business users and that are relevant to discover, select, and compose services.
Integration of heterogeneous services	Partly covered by existing approaches, not yet covered for service orchestration in processes	Definition of a new, abstract service description allowing for the integration of heterogeneous services, such as SWS, WSDL, or REST services.
Service selection, binding, replacement, and adaptation at various stages	Partly covered for design time and runtime, not yet for dynamic selection of the stage	Provisioning of new means and semantic descriptions to select, bind, replace, and adapt services at modelling time, design time, deployment time, and runtime.

Table 14: Fulfillment of individual requirements through SOTA

Requirement	Fulfillment through SOTA	Contribution to research community
Provide executable BPM functionality suitable for business users	Not yet covered by existing approaches	The design of a new process modelling solution for business users is provided. This comprises user support through new design principles, such as abstraction, semantic annotations, context-awareness, reuse, abstract service descriptions, and data flow support.
Increase number of users by people without IT knowledge	Not yet covered by existing approaches	
Usability, simplicity and understandability	Not yet covered by existing approaches	The process modelling solution is built according to principles of usability, simplicity, and understandability newly targeting business users.
Provide various abstraction views	Not yet covered by existing approaches	The process modelling solution for business users provides new, consistent layers, namely a graphical abstraction and a canonical format of the process model. In addition, a new concept for abstract service descriptions is provided.
Facilitate information search	Not yet covered by existing approaches	The process modelling solution for business users allows for the definition and provisioning of new semantic annotations and for context-awareness to support the information search.
Facilitate taking process ownership	Requirement to integrate user support in the LPMS	Through new support means to create understandable processes, the business users are encouraged to take the process ownership.

Table 15: Fulfillment of organisational requirements through SOTA

Requirement	Fulfillment through SOTA	Contribution to research community
Increase number of users according to task duties	Not yet covered by existing approaches	The design of a new process modelling solution for business users is provided. This comprises user support through new design principles, such as abstraction, semantic annotations, context-awareness, reuse, abstract service descriptions, and data flow support.
Achieve community acceptance, suitability for collaborative modelling	Not yet covered by existing approaches	

Table 16: Fulfillment of economic requirements through SOTA

Requirement	Fulfillment through SOTA	Contribution to research community
Generality and applicability to various scenarios	Covered by most existing approaches	The design of an extensible process modelling solution for business users is provided applying new principles of abstraction, context-awareness, and reuse to achieve generality.
Increase degree of reuse	Partly covered by existing approaches for case-based reasoning	A new design principle to increase the reuse of process models is defined.
Reduce modelling time	Requirement to integrate user support in the LPMS	Through abstraction, context-awareness, reuse, and data flow support the user saves time in modelling executable processes and reduces the training effort.
Lower needed expertise and training effort	Requirement to integrate user support in the LPMS	
Utilization of existing infrastructure and software assets	Use of service technologies	This thesis provides no significant contribution to the improvement of service technologies.
Use of internal development and support resources and avoid high workload for IT resources	The LPMS follows an approach of end-user empowerment	The design of a new process modelling solution for business users is provided. This comprises user support through new design principles, such as abstraction, semantic annotations, context-

Faster time to market	The LPMS follows an approach of end-user empowerment	awareness, reuse, abstract service descriptions, and data flow support.
Lower initial application integration costs	Use of service technologies	This thesis provides no significant contribution to the improvement of service technologies.

To conclude, there's currently no coherent BPM solution, the business user might deal with in order to model and execute processes. In this thesis, an approach of attaching semantic annotations to process elements is followed. The semantic annotations are intuitively understandable by the user and might be easily provided. In the backend, these semantic annotations are used in order to bind services to process elements and orchestrate them. In the following section the design principles for supporting the user in modelling executable business processes is presented.

3.5.2 Fulfilment of LPML Requirements

In this section, the fulfilment of LPML requirements through existing languages and technologies is checked. In particular, this comprises a critical view on graphical process modelling languages in terms of complexity for business users and executable process modelling languages in terms of flexibility.

A graphical process modelling language for business users

Key requirements
A graphical process modelling language for the LPMS has to - be usable and understandable by business users - provide sufficient information for service discovery and selection

With respect to the target user group of LPM, a process modelling language has to be provided that copes with the tension of allowing for familiarity and simplicity for business while providing expressiveness and semantic precision for the process execution (Silver, 2009). Hence, a minimal set of BPM language elements is defined. This set comprises only elements the business user might understand. As a study by Recker and Dreiling revealed, a user understanding one process modelling language will easily understand another one (J. Recker & Dreiling, 2007). Hereby, any structural differences between the languages, such as between EPCs and BPMN, do not matter according to Recker and Dreiling. The study did not show any results about creating process models and about modelling execution-related aspects, though. This is another aspect to be regarded in this thesis. In the following, the previously introduced languages YAWL and BPMN are considered with respect to business users.

YAWL criticism

Specifying process models in YAWL requires high expertise in IT and formalisms. According to Havey (Havey, 2005), YAWL targets the support of complicated patterns that are rarely used rather than to facilitate modelling, provide expressiveness, system integration capabilities, and business analyst savvy. Recker and Dreiling state that YAWL seems to be a suitable language from an academic perspective, however, due to the required expertise, misses acceptance and application by a broad user base (J. Recker & Dreiling, 2007). In this thesis, as well, the target users are not supposed to have that kind of knowledge. Hence, using the YAWL formalism is not appropriate for the business users.

BPMN criticism

Like YAWL, BPMN as well is too complicated for the target users. An analysis of BPMN models revealed that only 20% of its vocabulary is regularly used (zur Muehlen & Recker, 2008). Another study by Recker (J. Recker, 2008) revealed that 36% of respondents only use the core BPMN set to create their process models, that 37% use an extended set of BPMN symbols, and that the remaining 27% use the whole bunch of symbols and expressiveness. These studies clearly revealed the gap between the BPMN surface and the expressiveness in the backend. While BPMN has only the three shapes *activity*, *gateway*, and *event* on the graphical modelling level, the expressiveness and precision for execution purposes allows for a myriad of subtypes of each. The subtypes are distinguished by detailed graphical aspects, such as border style, the symbols inside, and the placement in the diagram. Hence, although the surface seems to be rather simply usable it is complex (Silver, 2009).

Recker (J. Recker, 2008) further states that a formal education for BPMN is required. Currently only about 14% of the BPMN modellers took part in a training. In addition, in his work, a statistics is provided indicating the uselessness of certain symbols. Furthermore, the event concept is criticised due to too much different types that are difficult to understand from a user perspective (J. Recker, 2008; J. Recker, Indulska, Rosemann, & Green, 2008). Although BPMN has been created to close the gap between describing and executing processes, in practice the models often lack the execution focus (Silver, 2009). Recker (J. Recker, 2008) states in his study that about half of the users model processes for documentation purposes. To document and describe processes with a language dedicated for execution overstrains business users.

And BPMN doesn't provide any support for translating the process documentation into execution aspects.

To summarize, like YAWL, the BPMN formalism is not suitable for the business users targeted in this thesis. Even creating an additional abstraction layer on top of YAWL or BPMN reducing the myriad of symbols and simplifying the modelling proceeding, is not an appropriate approach. At the end of the day, the user has to provide information about execution details he is not capable to provide. The user requires rather support in translating semantic information about business needs into execution details. Therefore, a language is needed that is able to understand this semantic information and translate it into execution information.

An executable process modelling language for business users

Key requirements
An executable process modelling language for the LPMS has to - be usable and understandable by business users - allow for the management of semantic annotations - provide sufficient information for service selection and composition

In order to account for the expressiveness and semantic precision regarding execution requirements, the aforementioned minimal set of language elements is not sufficient in order to build executable processes. Stein et al. (Stein, Kuehne, & Ivanov, 2009) see this as the semantic gap between business requirements and the technical implementation. To enhance the minimal set with execution details, tooling support is needed. Thus, the minimal language element set has to be extended by elements the tools might manipulate for gathering execution details. In the following the existing process modelling languages BPMN and BPEL are investigated with respect to the mentioned purpose. Since YAWL is already too complicated on the graphical process modelling layer, as well the execution aspects are not regarded as an option to be reused or extended in this thesis.

In order to support the user in creating executable processes, various approaches exist. The goal is to automate as many of the transformation steps as possible in order to avoid error-prone, time-consuming, and cost-intensive manual mappings (Stein et al., 2009). One approach is to specify the models in EPCs or BPMN and then transform it into BPEL. Various approaches exist to transform EPCs into BPEL code (Fötsch, Speck, & Hänsgen, 2005; Stein & Ivanov, 2007; van der Aalst & Lassen, 2005; Ziemann & Mendling, 2005). As well, for the transformation of BPMN into BPEL a couple of approaches exist. The approaches are however often limited in terms of complexity reduction, transformation power, or transformation processing (Stein et al., 2009). Furthermore , the transformation approaches might be classified according to the transformation implementation strategy (Mens, Czarnecki, & Gorp, 2005), the level of abstraction (Visser, 2001), or the refinement strategy (Czarnecki, Eisenecker, Glueck, Vandevoorde, & Veldhuizen, 2000; Greenfield, 2004). For example, Gao (Gao, 2008) proposes an approach to map BPMN to BPEL through a two-phase transformation comprising the application of a static token flow analysis to parse BPMN models into sub-flows and the transformation of the subflows to BPEL, based on a pattern. Roser et al. (Roser, Lautenbacher, & Bauer, 2007) propose a framework to generate workflows.

However, currently, there's no method that is able to transform all kinds of EPC or BPMN models into BPEL code due to structural differences of EPC respectively BPMN and BPEL. While EPCs and BPMN are graph-structured, BPEL is block-structured. Although approaches have been described for the transformation of graph-structured into block-structured languages (Mendling, Lassen, & Zdun, 2005; Ouyang, Dumas, ter Hofstede, & van der Aalst, 2007), issues still remain. Silver (Silver, 2008) describes interleaved flows and attached events as examples of where BPMN code cannot be transformed easily into BPEL code. And even for simple models that might be transformed, the specification of the transformation script again requires high experience in IT.

For these difficulties in transforming BPMN models to BPEL models, an approach including a closer model transformation is favoured in this thesis. A transformation risks the loss of information and errors during the transformation. Hence, extending BPMN by elements supporting the user in providing execution information is not an option for this thesis.

To summarize, there exists no process modelling language that allows for supporting the business user both in modelling and executing processes. This thesis proposes the use of semantic annotations to gather information from the user that might be used for process execution.

An execution environment for the LPMS

The LPMS aims at keeping the business user free from execution details while modelling yet executable processes. Hence, the user specifies abstract descriptions of the process steps. The process steps are then instantiated with services. In order to cope with changing requirements the instantiation of the process steps has to be dynamic, flexible, and adaptable. In the following, existing technologies are investigated in terms of the suitability for the LPMS.

Key requirements
An execution environment for the LPMS has to allow for - handling semantic annotations - the dynamic selection, replacement, and adaptation of services.

As aforementioned, BPEL is the de-facto standard as process execution language. Hence, BPEL engines have come up being able to execute BPEL processes. A BPEL process works fine within a closed environment where services are harmonized. However, in the heterogeneous, open web the configuration and composition of services require more sophisticated mechanisms. Services are difficult to find. Different data types have to be aligned syntactically and semantically. Further, flexible process and service instantiation is required in order to achieve more flexibility in case of changing requirements or failures. In this thesis, it is envisaged to allow for selecting, binding, and replacing services during modelling, deployment, and execution.

Rao and Su (Rao & Su, 2005) provide a survey of existing approaches to dynamically and flexibly compose web services. Furthermore, Andrikopoulos et al. (Andrikopoulos et al., 2008) provide an overview of all areas related to the engineering of service-based applications with particular focus on the capability to dynamically adapt to different scenarios. An example adaptation mechanism is provided by Brogi and Popescu (Brogi & Popescu, 2007) that adapt existing services to changing client requests. In general, the adaptation approaches are mainly based on Artificial Intelligence (AI) techniques. Those approaches to reason about web services are applied by the LPMS. However, the reasoning procedure is not described in detail in this thesis. It is referred to the work performed in the project SOA4All for further information.

An approach by Casati et al. (Casati, Ilnicki, Jin, Krishnamoorthy, & Shan, 2000) is the eFlow solution providing a number of features in order to support service and process specification and management. This comprises a service composition language, events and exception handling, ACID (atomicity, consistency, isolation, and durability) service-level transactions, security management, and monitoring tools. According to the authors, the eFlow model allows for specifying processes that automatically configure themselves at run-time according to the user needs and features of services that are available in the web. However, the selection mechanism is based on predefined selection rules. Furthermore, well-defined service descriptions and mapping rules have to be defined in advance. Even for the so called generic nodes serving as service templates for a set of similar services, the list of services and their descriptions have to be specified. The LPMS requires an even more dynamic selection in an environment where selection rules and service descriptions are not available and mapping rules are created on the fly.

To implement some of the aforementioned requirements for the LPMS through BPEL engines, Colombo et al. (Colombo, Di Nitto, & Mauri, 2006) have developed SCENE, a BPEL engine supporting dynamic binding and self-adaptation disciplined through rules.

Finally, similar to YAWL and BPMN, the execution details in BPEL are hard to understand for non-IT-experts. The same applies to SCENE that requires high effort spent by system integrators in defining adaptation rules.

To summarize, currently there exists no execution environment supporting the user in providing execution information, dynamically adapting processes, and dynamically selecting and replacing services.

4 LIGHTWEIGHT PROCESS MODELLING PRINCIPLES

The aim of LPM is to simplify the work of a process designer by hiding technical aspects, performing automatic optimizations, allowing for the late binding of concrete services, and substituting services at runtime. In this section, the core design principles of LPM are described that implement this user support. These design principles comprise abstraction, semantic annotations, context-awareness, patterns and templates, goals, and data flow support. The patterns, templates, and goals support the user in modelling the control-flow of its processes. The data flow aspect highlights the process model from a data exchange perspective. Semantic annotations and context might be applied to both the control and data flow perspective. The design principles are formalized and structured by the LPM metamodel that describes the necessary elements for LPM and their relations.

For example, through LPM, the user is supported in specifying process steps not as operations bound to concrete services but as a set of requirements. The requirements express desired functionalities and characteristics and are described according to a shared conceptualization, such as an ontology. This ensures a common understanding of the descriptions.

As aforementioned, this section specifies the LPM metamodel in section 4.1 and its implementation in two abstraction layers of the LPML. For both abstraction layers a specific metamodel is defined and presented in section 4.2.2.1 for the canonical format of the LPML and in section 4.2.2.2 for the graphical representation. Since both abstraction layers are based on the LPM metamodel, the equivalence of the two LPML metamodels is guaranteed. While the canonical format comprises sufficient information for process execution through tools, the graphical layer takes into account the targeted business users. The automatic process enhancement for execution and the processing of the canonical format through tools is described in section 1. An evaluation of the usability of the graphical LPML layer with respect to business users is covered by section 7.3.5.

The LPML comprises new process modelling concepts for the LPM design principles that realize the business user support and are not yet implemented in existing process modelling languages. Selected concepts of existing process modelling languages, such as the Business Process Modelling Notation (BPMN) (OMG, 2006) and the Web Service Business Process Execution Language (WS-BPEL) (OASIS, 2006) complement the LPML.

For each design principle, the benefit, the structure, the application to LPM, and the implementation in the LPML is described. Hereby, the implementation in the two abstraction layers of the LPML is detailed. Parts of the description of the design principles have already been published in Schnabel et al. (Schnabel, Born et al., 2009).

The LPML metamodel describes the elements, their properties, the relationships between each element and the constraints applicable in their usage. Some elements of the LPML are not provided directly by the process designer but might be derived automatically by tools exploiting predefined semantic annotations of services and goals.

For the LPML metamodel of the canonical format, the element descriptions are presented in tables comprising information about the elements included, the attributes or literals, a potential reference to a context file, and semantic annotations.

This section starts with the presentation of the LPM metamodel in section 4.1. Afterwards, the design principles for LPM are described in detail. These principles are abstraction (section 4.2), use of semantic annotations (section 4.3), context-awareness (section 4.4), patterns and templates (section 4.5), goals (section 4.6), and data connectors (section 4.7).

In the context of the Design Science, this section covers the first part of the design of the solution comprising constructs and the conceptualisation.

4.1 THE LPM METAMODEL

This section presents the LPM metamodel comprising the artefacts that are needed for modelling and executing processes in a lightweight way and the relations between those artefacts. The metamodel contains elements implementing the LPM design principles. Figure 5 depicts the LPM metamodel. The following Table 17 gives an overview of the elements and references the sections where those elements are described in detail.

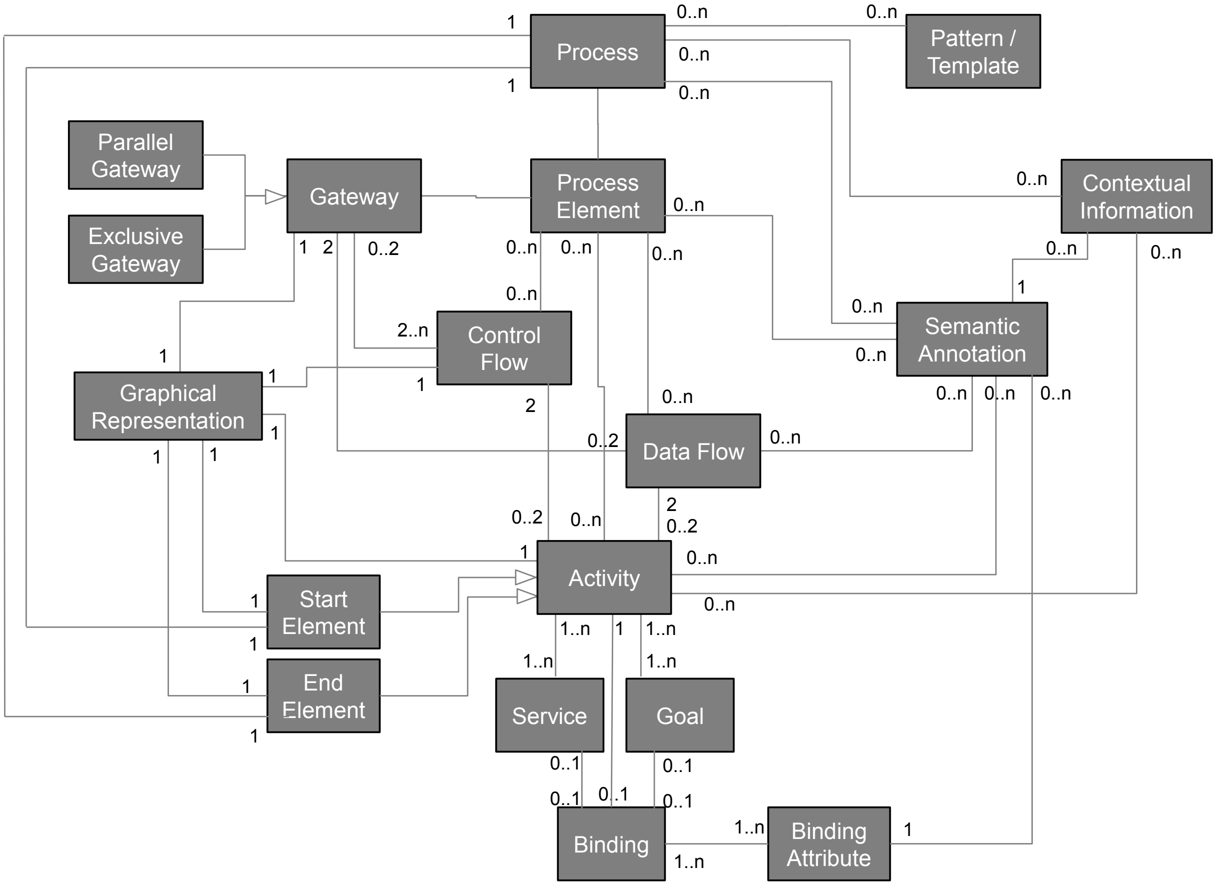


Figure 5: LPM Metamodel

Table 17: Elements of the LPM metamodel

Metamodel element	**Description**	**Described in detail in section**
Process	Container of all process elements	4.2.2.1
Process Element	ProcessElement is a generic construct for the abstraction of flows, gateways, and activities.	4.2.2.1
Control Flow	Control Flow is an association of two process elements determining the sequence of those elements	4.2.2.1
Data Flow	Association of two process elements in terms of output and input data	4.7
Activity	Unit of work in a process	4.2.2.1
Start Element	Starts and instantiates the process	4.2.2.1
End Element	Ends the process	4.2.2.1
Service	Instantiation of an activity	4.6
Goal	Kind of category for an activity	4.6

Binding	Specification of binding a service and a goal to an activity	4.6
Binding Attribute	Attributes to characterize the binding	4.6
Gateway	Represents a process split or merge according to a specific condition.	4.2.2.1
Exclusive Gateway	Gateway for the fulfilment of exactly one condition	4.2.2.1
Parallel Gateway	Two or more conditions might be fulfilled and the according flows followed	4.2.2.1
Semantic Annotation	Semantic information for various elements	4.3
Pattern / Template	Predefined process parts to be reused as a whole	4.5
Contextual Information	Container for environment information	4.4
Graphical Representation	Element specifying the graphical symbolization in a process editor	4.2.2.2

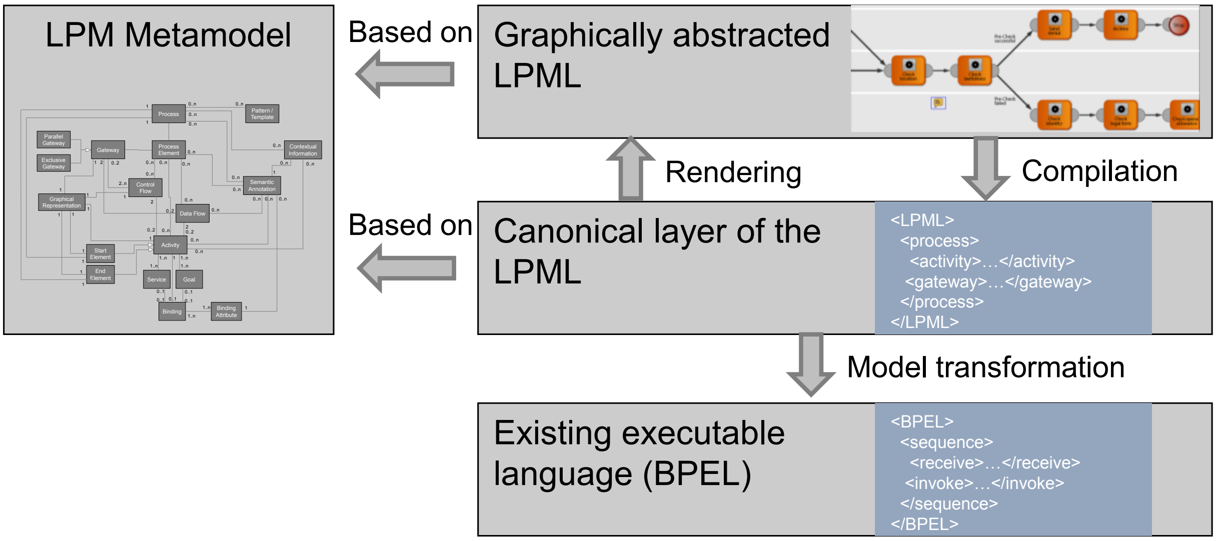
4.2 Abstraction Layers for the LPM Metamodel

4.2.1 Benefits of Abstraction

In order to keep the business user free from execution details, a specific abstracted view of the process model has to be implemented. Hence, besides a canonical format for the machine communication, a format has to be provided that the business user is able to cope with. This is in line with the requirement to create a new, consistent BPM language stack comprising an abstract graphical layer and a canonical format. Further, the requirement to include various abstraction views is addressed. If required, any IT experts might manually manipulate the process models in the canonical representation. Thus, the LPM process models contain the necessary information for machine communication and can simultaneously be read by business users.

The presented LPM approach is based on a common metamodel for all representation formats. This common metamodel has been presented in the previous section.

4.2.2 Structuring the Abstraction and Applying it to LPM


LPM Metamodel
Based on
Graphically abstracted LPML
Rendering
Compilation
Based on
Canonical layer of the LPML
Model transformation
Existing executable language (BPEL)

```
<LPML>
  <process>
    <activity>…</activity>
   <gateway>…</gateway>
  </process>
</LPML>
```

```
<BPEL>
  <sequence>
    <receive>…</receive>
   <invoke>…</invoke>
  </sequence>
</BPEL>
```

Figure 6: Languages and representations for LPM

LPM mainly comprises two abstraction layers that are represented through the LPML (see Figure 6). The graphical abstraction layer is the modelling interface for the user to model its processes, relies on the ability to design process models and their elements with a minimal set of information, contains additional rendering information, and is represented by the upper layer in Figure 6. It is designed for non-modelling experts, is kept very simple, and abstracts from sophisticated execution aspects. The graphical abstraction layer is described in detail in 4.2.2.2. A specific process editor designed for the LPML has been implemented. The middle layer in Figure 6 contains the canonical, executable representation of the process model that is semi-automatically created from the abstract model. The canonical layer is presented in detail in section 4.2.2.1. Since the graphical and the canonical LPML layers both use the common LPM metamodel, the compilation and rendering of the process models to transform one model to the one on the other layer might be easily implemented. An existing implementation of the transformations can be found in (Pavlov et al., 2010). A formalism for the compilation, rendering, and transformation might be based on a graph transformation formalism. Two models (seen as two graphs) need to be homomorphically equivalent. However, this formalism is not described in detail in this thesis. The LPM design process in section 5.1 describes how the graphical process model is enhanced by execution information. The lower layer in Figure 6 covers the representation in BPEL that is a currently executable language. The model transformation from the LPML model into a BPEL model is covered by section 5.1.6.

The abstraction principle provides views on processes for
- Business users - Machine communication

The canonical format of the LPML as the ground layer is rather complex. The user only sees the abstract graphical model and gets guided through wizards to extend these models by information that is used to make the processes executable. This information is mainly given through semantic annotations that are described in the following section. Besides the elements for the graphical representation, the canonical layer comprises elements to process this information in terms of semantic annotations that is given by the user. In order to complete the abstract model with necessary execution details, components for composition, optimization, and execution have been specified. The user however, only gets in touch with them indirectly through a process editor in case additional information is needed. A composition component instantiates unbound activities and goals with patterns, templates, or services. The composer has additional optimization functionality with respect to both functional and constraints in terms of service quality. Finally, the process models represented in the LPML are executed by a specific execution engine. Since the abstract graphical process (upper layer in Figure 6) is just a representation format of the canonical layer, no model transformation is needed.

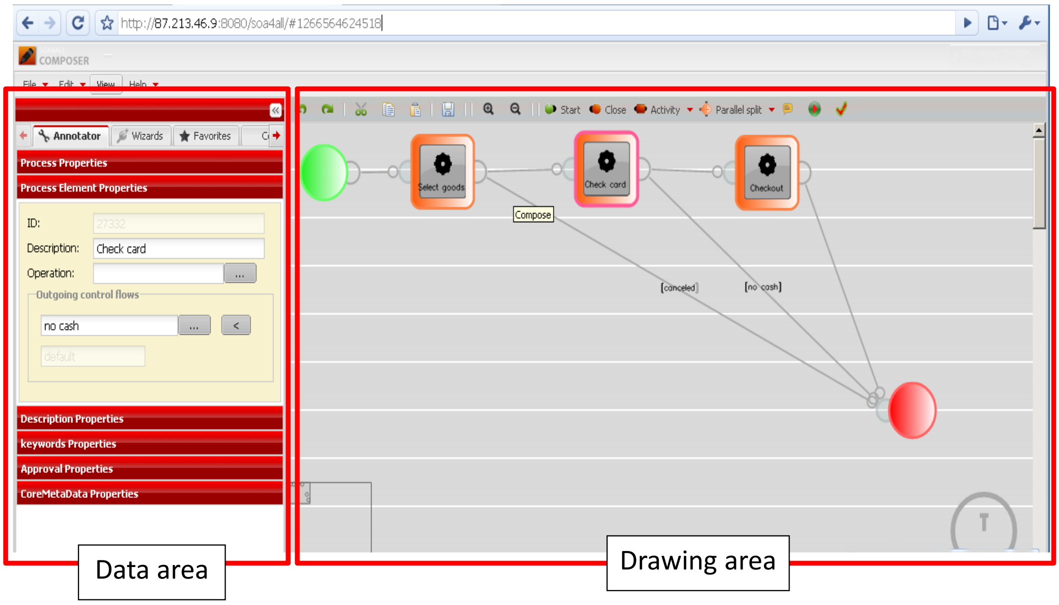

ID: 27332
Description: Check card
Operation:
Outgoing control flows
no cash
default

Figure 7: The abstraction principle implemented in the process editor

Figure 7 depicts how the abstraction principle is implemented in a process editor. On the right hand side, a standard drawing area is provided where users might create, modify, or view their process models. The drawing area represents the abstract graphical process model comprising simple symbols, such as a start element, an end element, activities, and flows to connect the elements. On the left hand side, the data area is provided serving as editor to specify elements and provide information necessary to make the processes executable. The data inserted directly instantiates the canonical format without any graphical representation. As aforementioned, the information to make processes executable is given in terms of semantic annotations. In the next section these semantic annotations are described.

4.2.2.1 Canonical Format

Figure 8 depicts the LPML metamodel of the canonical format. The elements of the metamodel are introduced in the following sections. The metamodel is represented by an UML class diagram. However, the metamodel might be represented as well in Java notation. The metamodel instances, the concrete process models, might be represented as well both in UML and Java notation. An example of a process model in Java notation can be found in section 9.1.1. A formalism to reveal that the LPML metamodel of the canonical format is based on the LPM metamodel might be shown by a graph transformation formalism. The two models (seen as two graphs) need to be homomorphically equivalent.

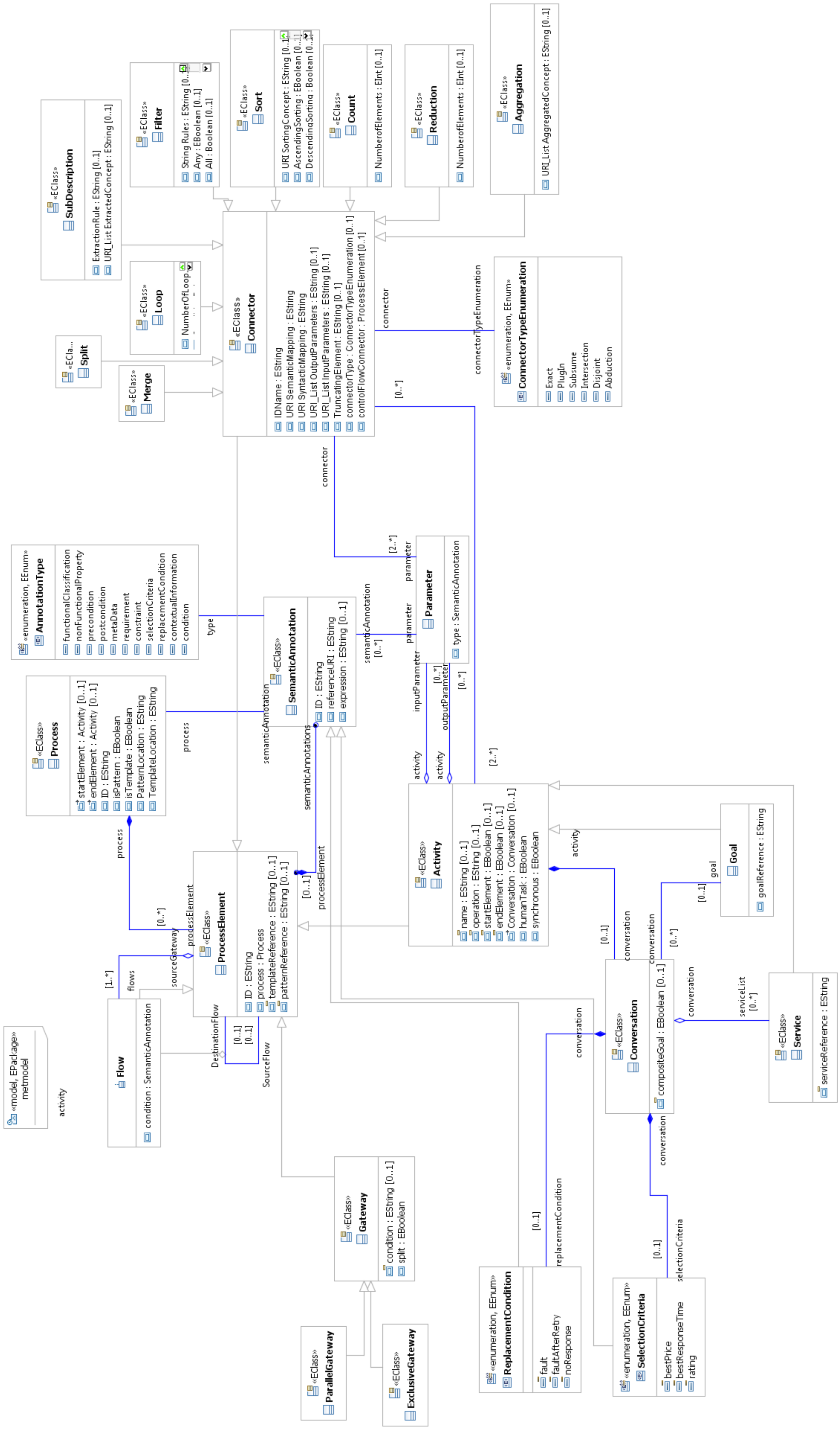


Figure 8: Complete LPML Metamodel

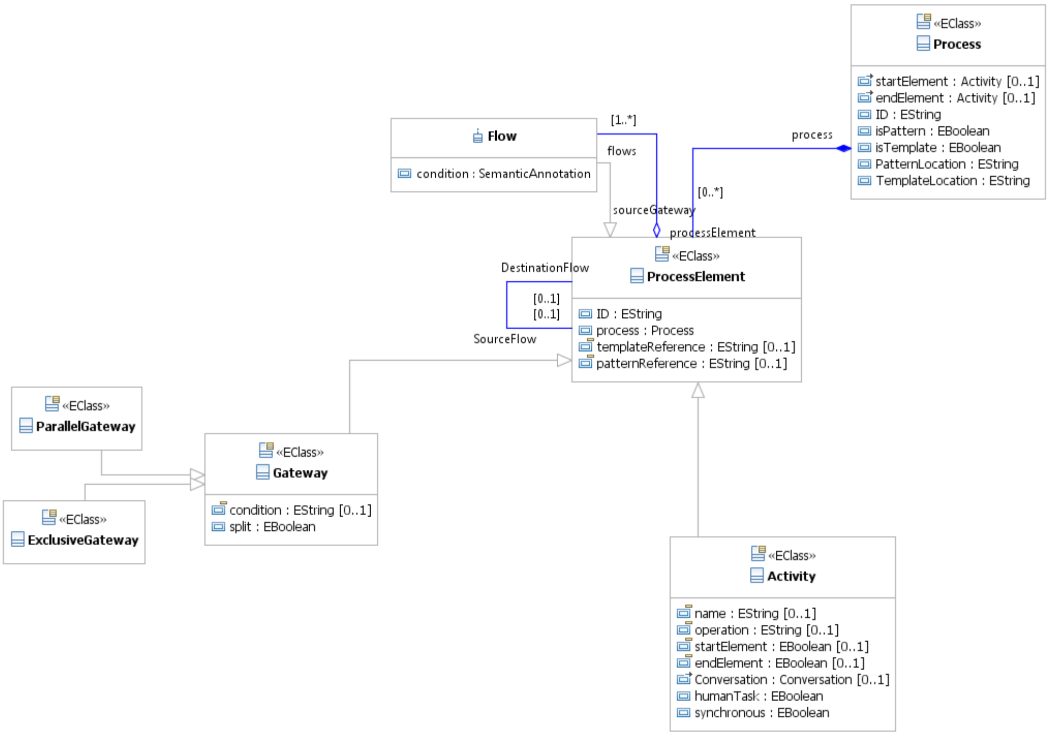


Figure 9: Parts of the LPML metamodel that are relevant for the graphical abstraction

The core elements of the LPML metamodel are those elements that are relevant for the graphical abstraction layer. Figure 9 depicts a view on those elements that are introduced in the following.

Process represents the container of process elements. Processes have a special association to exactly one start element. This start element represents the entry point into the process. Further, processes are associated to exactly one end element representing the entity that performs the callback in case the process has terminated. The start element is invoked by external callers and triggers the whole process. The process might be encapsulated and published as a service. It is described in detail in Table 18.

In addition the attributes *isPattern* and *isTemplate* are included. These two attributes define whether a process is a pattern or a template. In these two cases the process does not necessarily contain a start and an end element. The support for patterns and templates is described in detail in section 4.5.

Table 18 : Description of *Process*

Process	
Elements included	All elements might be included. A process necessarily contains one start and one end activity.
Attributes	ID, isPattern, isTemplate, patternLocation, templateLocation, startElement, endElement
Association	ProcessElement, SemanticAnnotation
Reference to context file	Yes
Semantic Annotation	functionalClassification, nonFunctionalProperty, precondition, postcondition, metadata, contextualInformation

ProcessElement is a generic construct for the abstraction of flows, gateways, and activities. Table 19 covers its description and most of its children. The children described here are the basic elements for structuring the process through the control flow. *ProcessElement* contains common attributes and is part of the process. Each *ProcessElement* is connected to another through a source and a destination association characterized by a *Flow*. In addition, *ProcessElement* and its children refer to the according context file.

Table 19: Description of *ProcessElement* and its children

Related Elements	**Attributes**	**Semantic annotations**
ProcessElement		
Part of: Process	ID, templateReference, patternReference	Not applicable
Children: Activity, Flow, Gateway, Connector		
Association: SemanticAnnotation		
Activity		
Parent: ProcessElement	name, operation, startElement, endElement, humanTask, synchronous	functionalClassification, nonFunctionalProperty, precondition, postcondition, metadata, requirement, constraint, contextualInformation, selectionCriteria, replacementCondition
Children: Goal, Service		
Association: Conversation, Connector, Parameter		

Flow		
Parent: ProcessElement	condition	Not applicable
Gateway		
Parent: Process Element	condition, split	Not applicable
Children: ExclusiveGateway, ParallelGateway		
Exclusive Gateway		
Parent: Gateway		Not applicable
Parallel Gateway		
Parent: Gateway		Not applicable

Flow is an association of two process elements, the source and the destination element, except the flow itself. An additional aggregation between *Flow* and *ProcessElement* describes the amount of incoming or outgoing flows a process element has. *Flow* might represent both control and data flow. Furthermore, a flow might comprise a condition determining when to follow a flow.

Gateway is a *ProcessElement* that represents a process split or merge according to a specific condition. It might be a *ParallelGateway* or an *EclusiveGateway*. An inclusive gateway element is not explicitly supported. This element might be replaced by a combination of exclusive and parallel gateways. Furthermore, an exclusive gateway comprises a default outgoing flow in case no condition is satisfied. The first outgoing flow drawn by the user is taken as default flow. It has no condition and is the last flow taken if none of the other flows are evaluated to "true". Concerning the condition satisfaction in an exclusive gateway, various options might be implemented. The easiest solution is to follow the first flow the condition of which is satisfied. The user gets displayed the order of the flows. Hence, it is transparent which condition is evaluated to true and which flow is thus followed. However, some scenarios require firstly evaluating the matching of all flows to the condition and then follow the flow that meets best the condition. This kind of condition might be seen as global condition and is represented by the condition attribute in the gateway.

Activity is a *ProcessElement* that specifies the execution of a unit of work. An activity serves as a placeholder for expressing requirements and constraints and is instantiated by an optional goal and a service. However, the user's point of view abstracts from the instantiation.

The graphical process models created by the user serve for documentation purposes. In order to make these abstract process models executable the design principles as described in this section 1 have to be applied and thus, the process elements have to be enhanced by additional information. Activities have to be instantiated by goals and services. Further, conditions for gateways and those flows connected to a gateway have to be defined. In the following, the LPML elements that are responsible for the automatic background enhancement of the user's process model elements are presented. The required information is semi-automatically provided by composition tools and by the user.

4.2.2.2 Graphical Representation

In order to be easily understandable for the business user the LPML needs to be simple, abstract, and hide technical aspects. Hence, on the abstract LPML level there are only a number of elements visible. The according view on the LPM metamodel is visualized in Figure 10. Except the *Process* and *ProcessElement*, all elements have an association to the *GraphicalRepresentation* element. The GraphicalRepresentation comprises layout information and is specific to the implementation of a Process Editor. An example of according layout information might be read in the documentation of the SOA4All Process Editor (Pavlov et al., 2010).

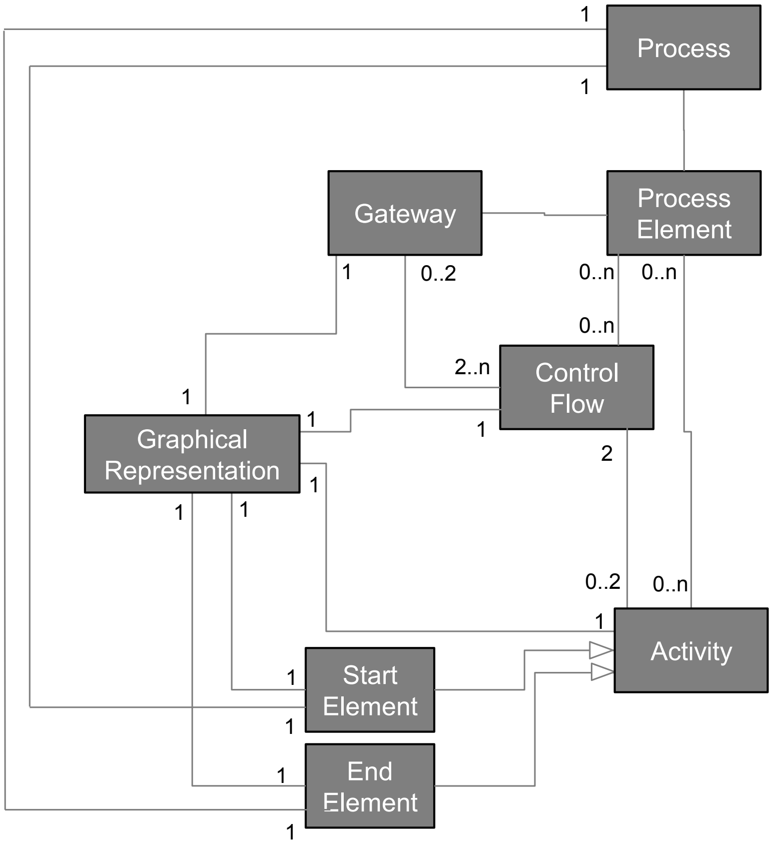


Figure 10: Metamodel of the graphical representation layer

The implementation of the graphical LPML layer in the Process Editor is depicted by Figure 11. The graphical symbols are given for the start element, the end element, activities, and the connections between those elements. The description of the symbols is addressed by Table 20. The user graphically creates his process by adding a start element, an end element, a couple of activities and by connecting these elements through flows. From a mathematical point of view, a process model is a directed graph.

For activities the user sets names and provides general information about requirements and constraints, often in natural language. In further steps, these process models and their elements are semi-automatically enhanced by information needed for execution purposes. However, these enhancements are not visualized in the process model view. Rather, a data area is provided supporting the user in entering the needed information. The data area is depicted by Figure 12.

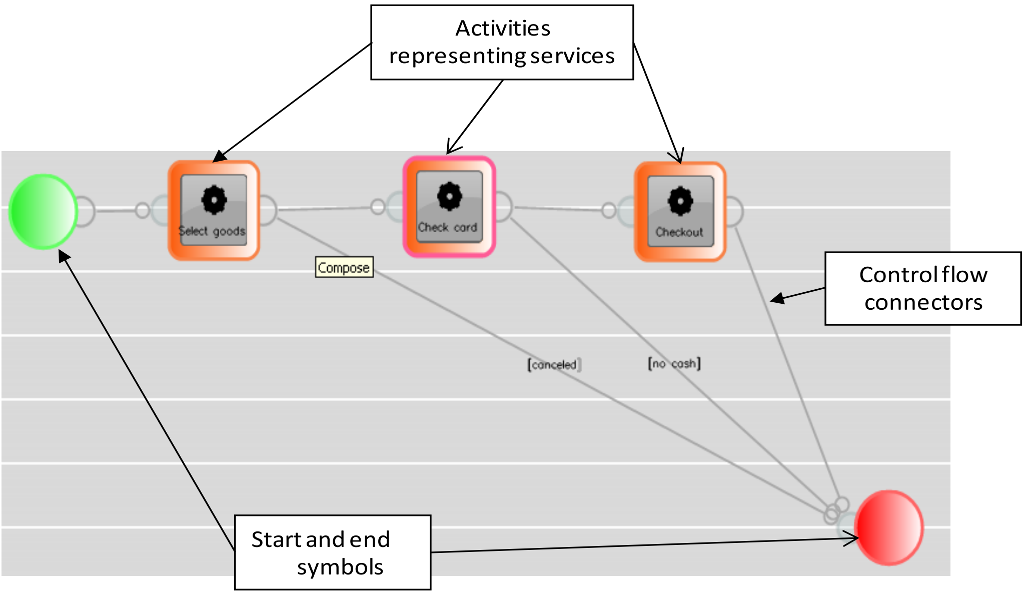
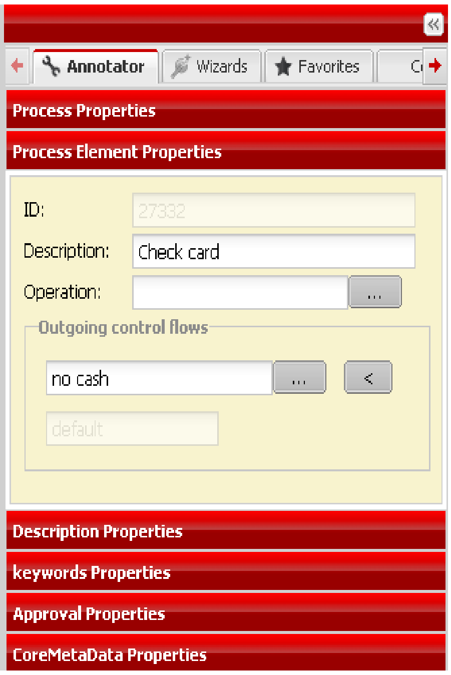


Figure 11: Graphical LPML representation



ID: 27332

Description: Check card

Operation:

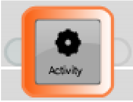
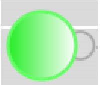
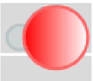
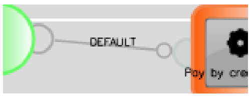
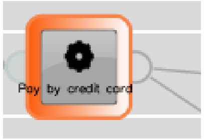
Outgoing control flows

no cash

default

Figure 12: Data area supporting the user in entering additional information

Table 20: Symbols of the graphical modelling layer

Symbol on the graphical layer	Graphical representation layer element	Creation, and hence LPM metamodel instantiation, of the following elements, the according attributes, the according semantic annotations, and the according associations
	Activity	- Process Element - Activity - Goal - Service - Parameter
	Start element	- Process - Process Element - Activity (Start element attribute is set to "True")
	End element	- Process Element - Activity (End element attribute is set to "True")
	Control flow	- Flow
	Gateway	- Gateway - Exclusive / Parallel Gateway

4.2.2.3 Design Rules for LPML Process Models

Design rules help to reduce any hurdles hindering the business user to create process models with the LPMS. It has to be made sure that the models the user creates might eventually be executed. Therefore, some general modelling design rules are presented the user has to stick to in order to provide sound process models. This is as well valuable to avoid deadlocks during process execution. Potential deadlocks should already be avoided at modelling time. The following rules apply to LPML models:

- A process model might contain multiple activities and gateways that are connected through the control flow
- Any control flow is connected to a source and a destination. The source and destination might be an activity, a gateway, a start element, or an end element

- An activity has exactly one incoming edge and exactly one outgoing edge.
- In the graphical abstraction of the LPML, an activity might contain an implicit gateway.
- For gateways, split and join gateways are differentiated in order to specify the incoming and outgoing edges.
- A split gateway has exactly one incoming edge and two or more outgoing edges.
- A join gateway has two or more incoming edges and exactly one outgoing edge.
- The LPML allows a gateway only to have incoming or outgoing connectors of the same kind. This means, a combination of AND, OR, and XOR in one gateway mustn't exist. Figure 13 depicts this design rule using a simple modelling sample.
- Any process model has exactly one start node and one end node.
- The control flow is a path from the start node to the end node. The process model has to make sure that all control flow branches are eventually merged before the end symbol.

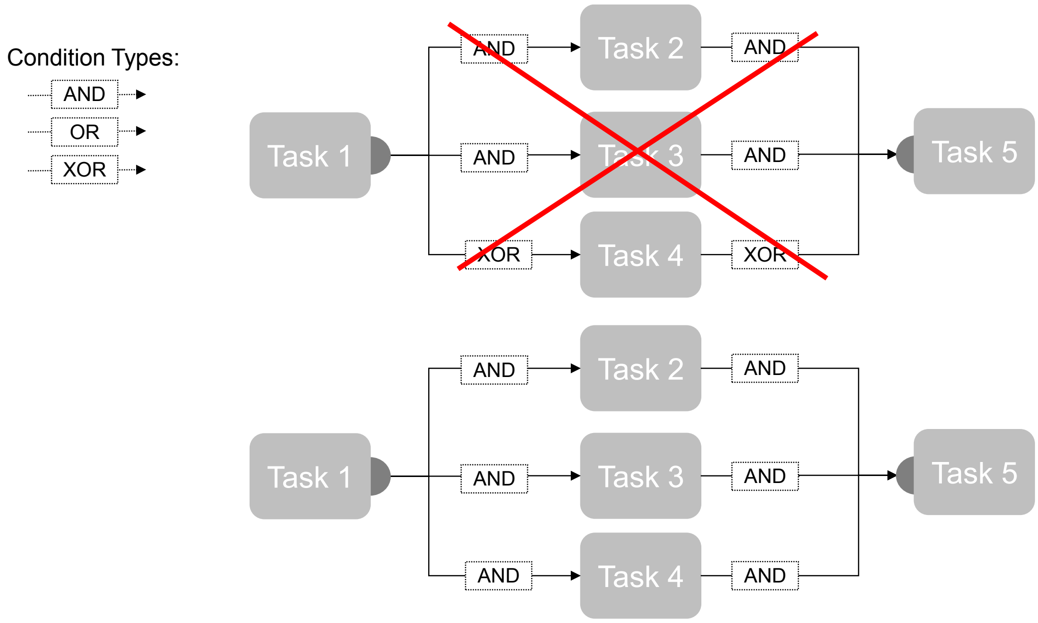


Figure 13: Gateway logic in the LPML

The mentioned design rules are needed in order to avoid deadlocks. Further, it has to be made sure that deadlocks aren't caused by missing elements. If the control flow is split, it will have to be made sure that it eventually is merged at a later point in the process.

4.3 Semantic Annotations

As described in the motivation section, the business users as target users of LPM often have difficulties in providing information to make processes automatically executable.

Further there is no common understanding of business process models and the terminology used (Blechar, 2007; Mendling & Recker, 2008).

4.3.1 Benefits of Semantic Annotations

In order to bridge the gap between business specification and execution details and to support the provisioning of execution-related information for processes and services, semantic annotations are attached to processes and its elements. These annotations mainly have to be provided by the user. They are more intuitively understandable for business users and allow for the definition of requirements rather than specifying concrete execution information. As well, properties and parameters might be modelled independently of objects through referencing semantic annotations representing these properties and parameters (Lillehagen & Krogstie, 2008). Reuse of processes and fragments might be supported through semantic tags (Hornung, Koschmider, & Lausen, 2008). The semantic annotations provide information about discovery, composition, and execution of processes and services.

Semantic annotations
- Provide a semantic meaning for processes and services - Reference a common knowledge model - Provide information about process and service discovery, composition, and execution

4.3.2 Structuring Semantic Annotations

LPM encourages enriching process models with semantic annotations. Semantic annotations complement the syntactic description of processes and their modelling elements by providing a semantic meaning and a reference to a common knowledge model, such as an ontology. Semantic annotations for software are expressions formulated in a common description language, such as RDF. The linkage of annotations to an ontology facilitates to automatically read and process the annotations. In addition, human users might use annotations to share descriptions and hence, foster a better understanding amongst users. Annotations support a sort of automation during both modelling and execution of processes. Annotations might also be used to check the fulfilment of requirements and constraints.

Madhusudan et al. (Madhusudan et al., 2004) propose an approach to add semantic annotations to processes – named cases in Madhusudan et al. – in order to support case-based reasoning. The authors define tags for the procedural structure, input, output, preconditions, postconditions, resource type, and ranking. In this thesis, a similar approach is followed. However, the LPMS not only needs declarative annotations for entire processes but as well for activities. Furthermore a broader set of predefined annotation types is needed.

4.3.3 Applying Semantic Annotations to LPM

The LPMS annotations are ontology-agnostic. Hence, they might reference concepts described by different ontologies, such as WSMO, WSMO-Lite, Micro-WSMO, or OWL-S. Reasonners properly interpret those annotations to instantiate goals by services, to discover, or to compose services. Business users might use annotations to describe the entire process and activities with the following aspects:

- *Requirements*: Might be used to specify a domain-specific scope for the process and other global requirements. For activities, requirements describe the functional classification of desired services.
- *Constraints*: Have restrictive or negative implications, limiting the scope or acceptable functionality for entire processes and activities. As requirements, constraints might be used to derive the functional classification of activities.
- *Non-functional properties*: May specialize the process and activity behaviour according to factors such as dependability, reliability, performance and ability for transactions. Other examples to characterize the activity instantiation with services are the location of services, the price, or security parameters.
- *Metadata*: Support additional information such as author, creation date, versions, and revisions. Metadata might be applied to entire processes as well as activities. Further, metadata supports the customization of the LPMS to a specific domain.
- *Logical expressions:* Gateways comprise logical expressions in order to select an outgoing flow. Furthermore flows might be annotated by logical expressions in order to specify when to follow this flow.
- *Functionality-based annotations:* Related to activities these annotations might reference functional classifications, preconditions and postconditions and inputs and outputs. The functional classifications comprise dimensions for the classifications and are categorized into three high-level classifications, the operational, designation, and extensible classifications. The latter is a general dimensional construct allowing for extending the set of dimensions by domain-specific dimensions.

The activities that are described by semantic annotations for the aforementioned categories form the basis for the discovery of services. A discovery engine might directly read these annotations provided in a semantic description language and match them to existing service descriptions in order to find fitting services. Another option is to first lower the semantic annotations to XML and then feed this XML description into a discovery engine.

4.3.4 Semantic Annotations in the LPML Metamodel

The semantic annotations might be provided in a user-friendly way in natural language or through keywords and in the preferred terminology. The annotations are easily entered through the annotator in the process editor as depicted in section 5.1.1 and section 5.2.2. These annotations can be in principle optionally attached to any process element including the process itself. However, for some process elements, the annotations are mandatory. The user has to provide annotations for the following process elements:

Process: annotations are used to describe the functional classification, non-functional properties, preconditions, postconditions, and metadata. These annotations are valid in a global scope and have precedence over annotations of the process modelling elements.

Activity: Annotations are mainly used to describe the following aspects. These annotations are used to map goals and services to activities and to compose services.

- Functionality and category, e.g. create invoice or send invoice
- Input and output data
- Non-functional properties, such as price for sending an invoice or availability of a send service. Further non-functional properties support optimization of processes and self-healing features at runtime.
- Preconditions, e.g. a valid order has to exist before an invoice can be created
- Postconditions, e.g. an acknowledgement has to exist after an invoice has been sent
- Metadata serving as additional information such as process author or privacy.
- Gateway/Flow: Annotations are used to describe conditions in flows or in exclusive gateways. The evaluation of the conditions determines the selection of the appropriate flow.
- Parameter: Activities have placeholders for input and output parameters. Both are semantic annotations, the input parameter is of type precondition, the output parameter of type postcondition.

In case the user has specified requirements and constraints to describe the process or activities, these might be used to derive semantic annotations. The requirements and constraints are translated into ontology-based semantic service and goal annotations, e.g. WSMO annotations for goals, WSMO-Lite annotations for Web Services, or

Micro-WSMO annotations for REST services. The translation is semi-automatically performed by the composition component. The degree of automation hereby depends on the IT-related quality of the users' requirement and constraint descriptions. In the following, the metamodel elements of the canonical LPML format for handling the semantic annotations are described. In the graphical LPML layer, the semantic annotations are not explicitly handled through symbols.

SemanticAnnotation is the class for all types of annotations for process elements. It contains the reference to the annotation file in case of an existing ontological annotation. This is represented by the attribute *referenceURI*. In case the annotation is newly created it is represented by the attribute *expression*. Any annotation is of a certain type *AnnotationType*. For the element descriptions provided by the user the types of *requirement* and *constraint* are used.

AnnotationType enumerates the potential annotation types. It is limited to annotations for functional classification, non-functional properties, preconditions, postconditions, metadata, conditions, requirements, constraints, selection criteria, replacement condition, and contextual information.

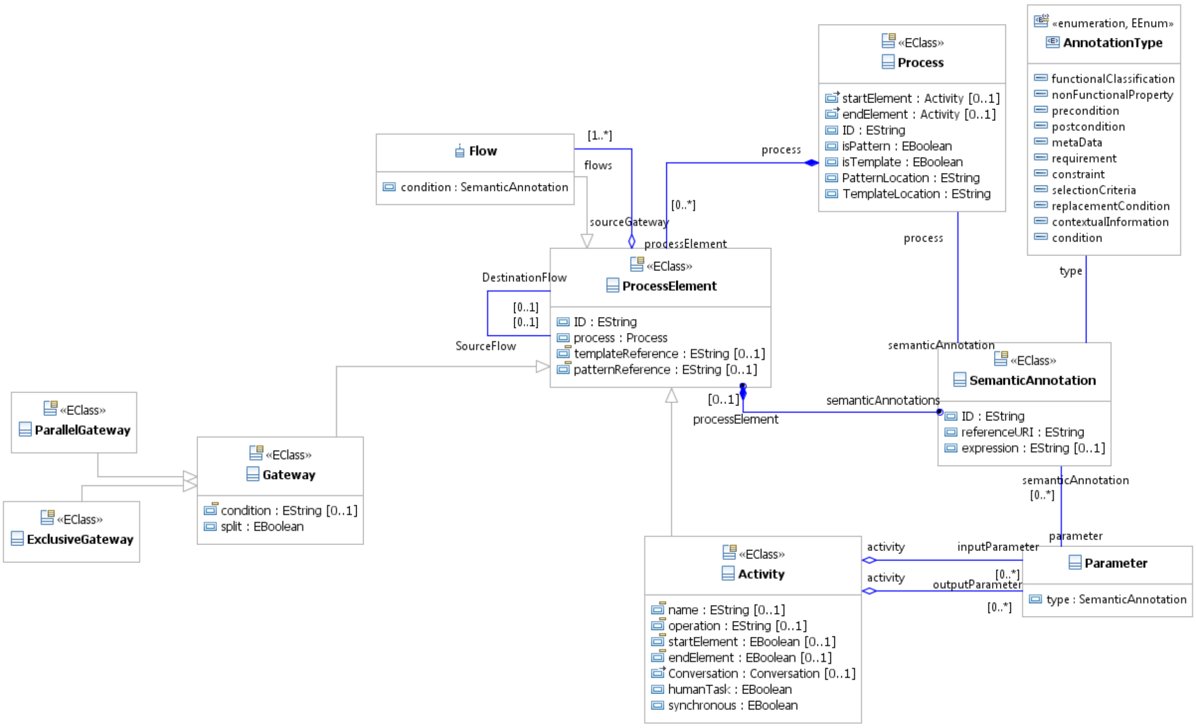


Figure 14: View on semantic annotations in the metamodel of the canonical LPML format

Parameter: The relation of *inputParameter* and *outputParameter* of an activity is specified by the Parameter element.

Figure 14 depicts the metamodel elements that are relevant for the concept of semantic annotations. Table 21 provides a detailed description of the elements needed in order to attach semantic annotations to the process elements.

Table 21: Elements for semantic annotations

Element	Related elements	Attributes/Literals
Semantic Annotation	Association: Process, ProcessElement, AnnotationType, Parameter	ID, referenceURI, expression
	Children: ReplacementCondition, SelectionCriteria	
Annotation Type	Association: SemanticAnnotation	functionalClassification, nonFunctionalProperty, precondition, postcondition, metadata, condition, requirement, constraint, selectionCriteria, replacementCondition, contextualInformation
Parameter	Association: SemanticAnnotation, Connector, Activity	type

Resolving semantic annotations

The semantic annotations are resolved at different stages of the process modelling and execution. How semantic annotations are resolved for discovering services is described in section 5.1 and in section 5.2.3. Further, the annotations are used at runtime to execute the processes. If a semantic annotation has to be resolved, the process will call a service representing a conceptual layer. This layer comprises a reasonner that is able to transform the semantic annotations into queries, to resolve the queries, and to discover a suitable service fulfilling the needs. Afterwards, the conceptual layer calls this service. The conditions in gateways and flows are mostly evaluated at execution time. A condition references a query, such as a SPARQL query, or other formal condition expressions. At runtime, the query is evaluated by a reasonner. According to the reasonner response, the decision is made which flow to follow.

Languages for semantic annotations

Potential languages for semantic annotations are RDFS and WSML axioms. RDFS is suitable for process fragment descriptions, context, and domain-specific knowledge while WSML is appropriate for axioms and logical expressions that can’t be expressed with RDFS.

4.4 CONTEXT-AWARENESS

4.4.1 Benefits of Context-Awareness

Applications for service composition and business process modelling have to react dynamically and flexibly to changes in the environment (Maamar, Benslimane, & Narendra, 2006; Michael Rosemann & Recker, 2006). Currently, the environment of the user and the systems is often not explicitly considered, for example the location, language, legal issues, or financial regulations (Pedrinaci et al., 2010; Michael Rosemann, Recker, & Flender, 2008; zur Muehlen, 2004). This leads to a limitation of the system capabilities.

For example, web services are a widespread option for the implementation of global service compositions through a simple request-response pattern. However, some situations require flexibility or autonomy in order to dynamically select operations or to dynamically participate in composite services or processes. Hence, services are required that are able to assess their current capabilities, commitments, and environment before participating in any composition (Maamar et al., 2006). Furthermore, processes have to be flexible and adaptable and the adaptation time has to be decreased (Michael Rosemann & Recker, 2006). Any changes in the environment might require the integration of filter mechanisms that allow for the dynamic adaptation of processes. A cause-effect relationship between the need for process flexibility and the impact on the process has to be increasingly considered (Michael Rosemann & Recker, 2006; Michael Rosemann et al., 2008).

A design principle of LPM is to overcome this limitation and take explicitly into account information about the user, the application, and their environment, the so called context. Context is further described as an application capability of discovering and responding to changes in the environment (Maamar et al., 2006). In this thesis, the following definition of context is used according to Dey (Dey, 2001):

“Context is any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction

between a user and an application, including the user and the application themselves."

In order to adjust the services and processes to the environment, the contextual information has to be gathered, e.g. through heuristic classification, and the relevant information has to be identified. Various context-aware applications have been developed, overviews can be found in literature (Lillehagen & Krogstie, 2008; Pedrinaci et al., 2010).

Context-Awareness
- Ensures the consideration of the process and service environment - Allows for the adaptation of processes and services to specific environments

However, according to Pedrinaci et al., most of the applications have been built for specific environments and lack a broad applicability in various domains. Therefore, frameworks have to be built that systematically support the adaptation of services and processes to contexts. In more detail, the framework has to support the modelling of contextual information and a solution for context adaptation. Maamar et al. (Maamar et al., 2006) discuss the suitability of context-awareness for a semantic mapping of web services. Often, an issue arises if different syntactic types do not match. This is important for LPM since the modelled processes are orchestrations of services. Three major issues are subject to the discussion: The deployment of context-aware web services, the contextualisation of web service compositions, and the conciliation of web service contexts. Through the use of context, the semantic meaning and the value of the syntactic types might be determined in order to achieve an automated, correct mapping. In the following, these issues are addressed in terms of service orchestrations in processes.

4.4.2 Structuring Context

It is assumed that context is a kind of information that is available or might be derived or extracted from data. The context might be represented as an object in a space. This space comprises dimensions or parameters representing a domain of values allowed for a specific context attribute, such as "time" or "location" (Lenat, 1998; Padovitz, Loke, & Zaslavsky, 2004; Saidani & Nurcan, 2007; Zaslavsky, 2002). In order to gather context knowledge out of context information, the search space has to be reduced (Pedrinaci et al., 2010). In the context of BPM, Rosemann et al. (Michael Rosemann et al., 2008) have built a metamodel of context perspectives based on the work performed by zur Muehlen (zur Muehlen, 2004). This metamodel integrates context into the traditional workflow perspectives introduced by Jablonski and Bussler (Jablonski & Bussler, 1996).

Ontologies might be used to handle context information for various purposes. One purpose is to express and structure relevant context information. Another purpose is to create contextual knowledge out of context information (Pedrinaci et al., 2010).

Bouquet et al. (Bouquet, Fausto, Van Harmelen, Serafini, & Stuckenschmidt, 2003) structure the context based on ontologies. According to the authors, an ontology is contextualised in case the contents are kept local and not shared with other ontologies. The relation to other ontologies is described in explicit mappings. Bouquet et al. have further constructed OWL-C as specification language for context. This thesis slightly modifies this proposed distinction between ontologies and context. The term *context* is used in a broader sense. In the sense proposed by Bouquet et al., context is similar to a domain ontology of this thesis.

Pedrinaci et al. (Pedrinaci et al., 2010) propose an umbrella ontology and a set of dimension ontologies as core part to structure context information. A dimension has structural and logical aspects. It describes aspects, such as quantities, units, and the conversion between units. Structural aspects describe the characteristics of the values included. In a dimension, the included values might be ordered or grouped. In contrast, the logical aspects cover the relative complexity and the relative convertibility. By relative complexity values are referred to that are obtained through an operation on values of other dimensions, such as a conjunction of two values, building a new value. Convertibility applies to the combination of values given in different units.

Besides ontologies, a context model might use the Core Components Technical Specification (CCTS) (UN-CEFACT, 2003) as a context base. The CCTS focuses on context for data modelling. The purpose of the CCTS is to provide a common semantic base and structure for data. Context-awareness is implemented by the concepts *Business Contexts* and *Business Information Entities*. Business contexts are grouped into eight context categories: Business Process, Product Classification, Industry Classification, Geopolitical, Official Constraints, Business Process Role, Supporting Role, and System Capabilities.

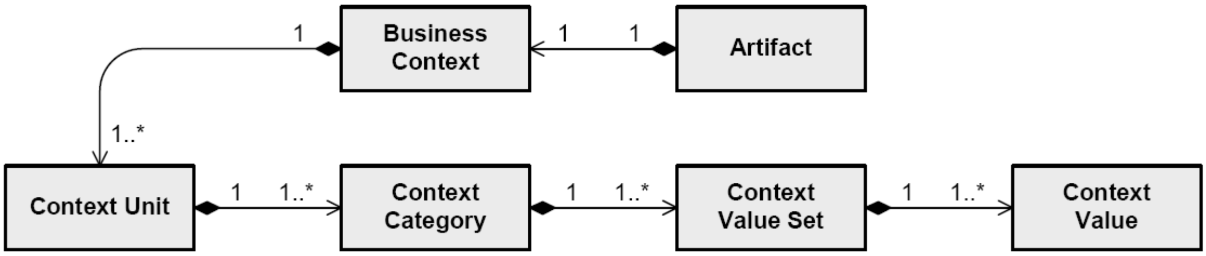


Figure 15: Simplified Meta Model of the Context Driver Principle according to (Stuhec & Crawford, 2007)

Figure 15 depicts an example of a context model based on the simplified meta model of a context-driver approach (Stuhec & Crawford, 2007). The model is drawn in a UML class diagram notation.

Any process artefact is associated to one business context that comprises various *context units*. The context units are related to context categories that are adopted from CCTS.

4.4.3 Applying Context to LPM

Depending on the application scenario and domain, different contextual information might be relevant for LPM. Saidani and Nurcan (Saidani & Nurcan, 2007) propose a context model organised according to the categories location, time, resource, and organisation. Further relevant context categories might comprise information about gender (Male/Female), country, language, season, weather, family, hobbies, company, user-ID, name, or affiliations. Rosemann et al. (Michael Rosemann et al., 2008) propose an onion model for context classification and typing that structures context into an immediate, internal, external, and environmental layer. Tan (Tan, 2007) introduces a context framework for B2B collaborations. According context dimensions are the user, company, time, and location. In their work, context is used to match suitable services to realize the cross-organisational business processes. Lechner et al. (Lechner, Schmid, Schubert, & Zimmermann, 1998) define various profiles that characterize a participant in a community, such as user profile, content profile, community profile, system profile, session profile, transaction profile, and case-based profile. Since LPM builds the foundation for community-driven process modelling and execution, these profiles might be used as context as well. A tool using context information has to define for any of them where the information comes from, e.g. from the user or from other sources.

An overview of context-adaptation in the BPM area can be found in (Ko et al., 2009). The application to web service composition is manifold. For example, Rosemann et al.

(Michael Rosemann et al., 2008) propose to align the contextualisation of processes with the overall goal of the process. The context-awareness allows for late binding, processes might be adapted based on context information, or process structures or the naming of elements might be proposed based on the context. Further usage scenarios concern the filtering of artefacts or the proposition of reference processes from other business domains (Schnabel, Born et al., 2009). Context-awareness for the LPMS means that any artefact of the process model is linked to the context. An artefact is only valid in that context. For the entire process and each element the context has to be identified and relevant information derived from the contextual information. According to (Saidani & Nurcan, 2007) this might be seen as definition of the context value range. The implementation is then chosen that fits best this value range. Further, it has to be defined how a process or an activity reacts on contextual information.

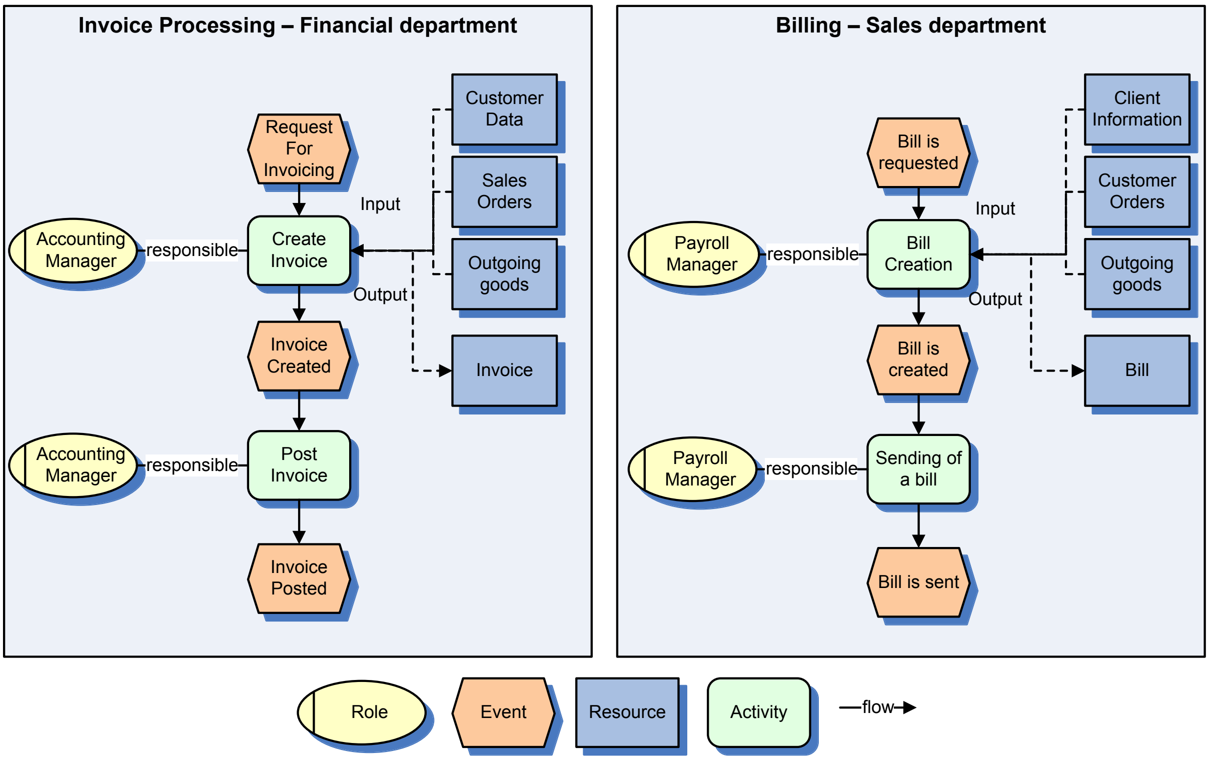


Figure 16: Example of an invoice processing process at business level (Schnabel, Born et al., 2009)

The design time and runtime for process modelling and execution in the context of LPM cannot be clearly differentiated. Hence, context information might influence process modelling and execution at design time and runtime. This differentiates the context-awareness of LPM from other existing approaches. In the following, the motivation for context-awareness and the application to LPM is presented.

Process models might be seen as a container for processes of the same nature (Rolland, 1998). Hence, the processes are instantiations of the process model. These instantiations might occur in various settings and contexts. Processes in the same context might build an additional context container that is part of the process model (Schnabel, Born et al., 2009). Figure 16 defined in the thesis of Matthias Born and depicted in Schnabel et al. (Schnabel, Born et al., 2009) visualizes an example of an invoice process model in two different contexts. The models use the notation of Event-Driven Process Chains (EPC).

On the left hand side in Figure 16, a typical invoice process of a financial department is depicted while on the right hand side an invoice process of the sales department is presented. Both departments are using their own terminology to describe the process artefacts. Often, the labelling of artefacts doesn't follow any naming conventions (Mendling & Recker, 2008). In order to achieve a better understanding of process models and hence increase the level of reuse, a common terminology is helpful. Hereby, reuse might increase in two dimensions. Reuse across process models means reusing parts of that model, e.g. activities. By applying common naming conventions the meaning of the activities gets unambiguous and it might be used in other models as well. Reuse of structural dependencies means using a set of connected process elements, e.g. a set of activities or a role that is linked to an activity. In Figure 16, in the financial department the linked role is called "Accounting Manager" while in the sales department the same role is called "Payroll Manager".

The introduction of context-awareness of process models and its elements allows for the creation of standardized, consistent, and understandable labels and descriptions of process model artefacts. This increases the reuse of these process artefacts. The labels and descriptions are based on common ontologies or dictionaries and linked to a specific context. Hence, reusing insufficient business artefacts and terminology might be avoided (Schnabel, Born et al., 2009). An insufficient terminology might be words having the same spelling and pronunciation but different meanings (homonyms) or words having the same spelling but different pronunciation and different meanings (heteronyms) (Schnabel, Born et al., 2009).

Another aspect is to adjust the sequence of activities according to the context. An example on how to adapt a process model structure to the context is visualized in Figure 17. The example is taken from Schnabel et al. (Schnabel, Born et al., 2009) and the dissertation thesis of Matthias Born. The upper process model is valid in the context C1, the lower model in C2 differentiating from C1. The process in C2 comprises an additional activity A4. According to the principle of context-awareness the process model is only stored once. For any activity the valid context is specified. Furthermore, the artefacts responsible for the process structure, such as source or destination references, connectors, or pre- and postconditions are related to specific contexts. Hence, the process structure might be adjusted to the context.

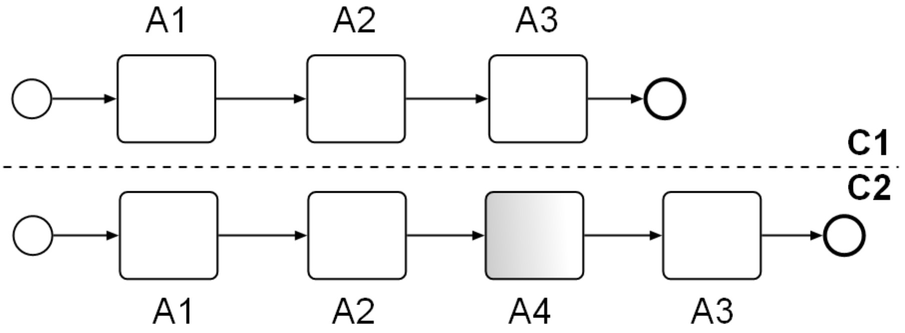


Figure 17: Context-aware Business Process Model

In a tooling setting for process modelling, the user might either specify the context or the context is set automatically by using information, such as the user profile or the tooling environment.

As aforementioned and shown in the example above, the context-awareness might be applied to various scenarios in BPM. LPM focuses on three of them. Context is used to dynamically discover, select, bind, and compos services, to label process artefacts according to a common understanding, and to adjust the structure of processes.

4.4.4 Context-Awareness in the LPML Metamodel

The context defines the environment a process artefact is used in. The LPMS uses context information in three ways to support the user. First usage is to support the discovery, selection, binding, and composition of services in order to instantiate the abstract activities modelled by the user. Further, the naming of activities and the process structure might be adjusted based on the context.

Instantiating activities based on context information

In order to support the activity instantiation, the context-awareness is implemented by the LPMS similar to an approach proposed by Ardissano et al. (Ardissano, Furnari, Goy, Petrone, & Segnan, 2007). Ardissano et al. propose a *Context-aware Workflow Execution* (CAW) comprising the following artefacts. For each artefact the relation to the LPMS is described.

- An abstract representation of the workflow that might include *abstract activities* hiding context-dependent properties.

Through the abstraction principle and the according separation of activity descriptions and their instantiation with services, the LPMS relies on the same foundational principles. A process model is created using the graphical elements of the LPML. Some properties of the process elements don't necessarily have to be specified by the user. These properties are filled by context information.

- An assignment of abstract activities with its implementations. The implementations differ according to the context.

In contrast to the CAW, the LPMS does not necessarily have predefined implementations for an activity. The context information rather supports the discovery, selection, binding, and composition of services in order to instantiate activities.

- An explicit declaration of conditions determining the selection of the context-dependent process part at execution time.

This is in line with the LPM tools – the process editor, the discovery engine, the composition component, and the execution engine – that have implemented rules to react on specific context values.

To process the context information the proceeding of Ardissano et al. (Ardissano et al., 2007) is used as a basis and modified for the purposes of the LPMS. As aforementioned, in the work of Ardissano et al., the process adaptation according to the context is only possible at runtime. The execution engine therefore, interacts with other components to retrieve the context-dependent implementations from a knowledge base and read the applicability conditions for the implementations. Then, a specific implementation is bound to the abstract activity. The LPMS however, uses context information both at design time and runtime. Pedrinaci et al. (Pedrinaci et al., 2010) describe how the context information is processed to invoke services. This thesis adjusts this approach to process modelling. After identifying the dimensions attached to the process elements, a context reduction gains context knowledge out of the context information. This content knowledge might be used for a service classification framework. The classification framework might then be used for discovering, selecting, binding, executing, replacing, or adapting services in a process.

Process elements reference business entities to align their labels

The second approach to use context information is to support the naming of activities as described in (Schnabel, Born et al., 2009). Every process element is linked to a business entity that has an unambiguous semantic meaning and is stored in a specific repository. As well, every business entity is linked to a specific business context. The business entity comprises various terms that are based on a dictionary or an ontology and make the business entities understandable for human users. In the context of the LPMS, a process element might be named using one of the terms of the related business entity. An example is the naming of an activity sending a purchase order. The business entity might be named "Send Order". The according terms could be "send purchase order" or "Send PO". In order to base the terms on a common understanding, existing dictionaries are used, such as Wordnet.

In the sample process depicted in Figure 16 in section 4.4, a potential invoice processing in the financial department and in the sales department is shown. In the example, the creation of an invoice represented by an activity is named differently in the two departments. The names are "Create Invoice" in the financial department and "Bill Creation" in the sales department. However, both activities have the same meaning. Hence, both activities refer to the same business entity in the associated repository. In addition, the terms of the business entity might be restricted to a specific context. For example, the term "bill" might be restricted to the sales context.

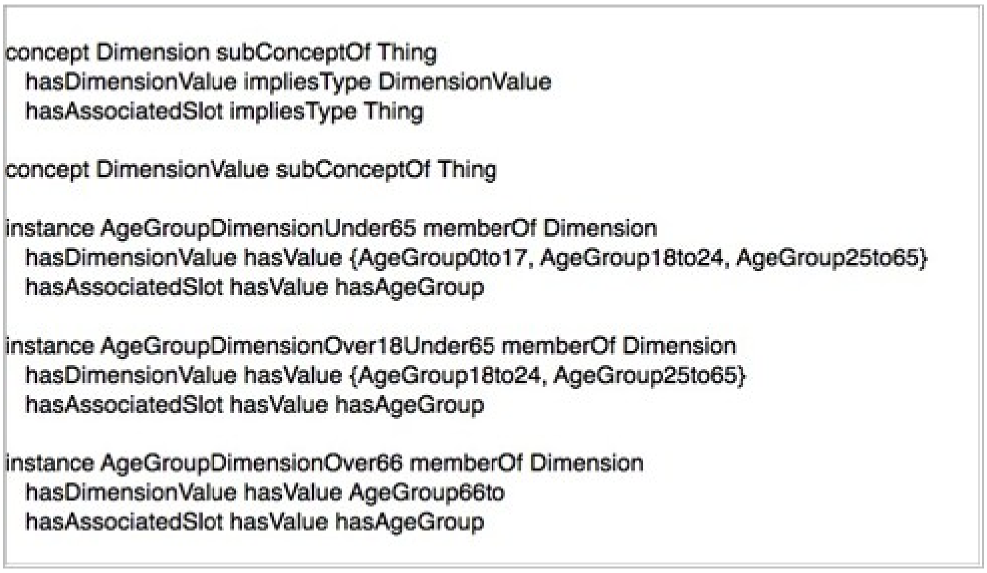
Another option for the alignment of activity names is to reference to ontological descriptions. These descriptions provide relations of concepts that might be used for relating activity names in different contexts.

Adjusting the process structure to the context

The third approach of context-awareness is the structuring of processes. This approach is as well described in Schnabel et al. (Schnabel, Born et al., 2009). Through the attachment of context information to a process element it might be indicated whether an element is valid in a given context. The context file hence contains a validity entity. This mechanism allows for the dynamic and flexible adaptation of process models.

Attach context reference to process elements

In this thesis, any process model is linked to a context file. Therefore in the metamodel of the canonical LPML format, a semantic annotation of the type *ContextualInformation* referencing the location of the context file is attached to the process element. In the graphical LPML layer, contextual information is not explicitly depicted, since this layer should be kept simple. The context file comprises a set of context dimensions the service, process, or process element is sensitive to. In addition, the context file identifies the business entity used for adapting the naming of activities.



```
concept Dimension subConceptOf Thing
  hasDimensionValue impliesType DimensionValue
  hasAssociatedSlot impliesType Thing

concept DimensionValue subConceptOf Thing

instance AgeGroupDimensionUnder65 memberOf Dimension
  hasDimensionValue hasValue {AgeGroup0to17, AgeGroup18to24, AgeGroup25to65}
  hasAssociatedSlot hasValue hasAgeGroup

instance AgeGroupDimensionOver18Under65 memberOf Dimension
  hasDimensionValue hasValue {AgeGroup18to24, AgeGroup25to65}
  hasAssociatedSlot hasValue hasAgeGroup

instance AgeGroupDimensionOver66 memberOf Dimension
  hasDimensionValue hasValue AgeGroup66to
  hasAssociatedSlot hasValue hasAgeGroup
```

Figure 18: Sample of an ontological dimension[15]

Representation of context information

The representation of context information is independent of the usage through naming or process structure adaptation or through support for the instantiation of activities.

In a formal way, the context might be represented in various formats. A simple approach is the representation as mathematical set. A predefined set comprises various values, e.g. a set of industries comprises the values *finance*, *automotive*, or *telecommunications*. The set attached to a process element comprises those values the process element is valid in. However, the sets and its values have to be predefined. A more flexible approach is based on ontologies. By using an ontology, the context information might be modelled as dimension. Any ontological goal or service description is then linked to the user's context information dimension, such as location, or language. In order to further structure the context information a hierarchy of ontologies is developed comprising one umbrella ontology and dimension ontologies. The dimension ontologies are based on this umbrella ontology. Pedrinaci et al. (Pedrinaci et al., 2010) assign general aspects of web services and temporal aspects

[15] See www.soa4all.eu for more information

to the umbrella ontology, whilst aspects, such as quantity, unit, or the conversion, are described by a dimension ontology. Figure 18 depicts a sample of an ontological dimension. The sample file is written as WSMO-based ontology in WSML notation. Currently, WSMO is extended by elements for context information by the SOA4All project team.

4.5 Reuse through Patterns and Templates

4.5.1 Benefits of Patterns and Templates

A pattern is „*the abstraction from a concrete form which keeps recurring in specific nonarbitrary contexts*" (Riehle & Zuellighoven, 1996). This definition has been created in the context of software development. Patterns provide independence from any implementation technology and from domain-specific settings (van der Aalst, ter Hofstede, Kiepuszewski et al., 2003). To apply patterns to the context of BPM, van der Aalst et al. is referred to (van der Aalst, ter Hofstede, Kiepuszewski et al., 2003). The authors state, that patterns address business requirements in terms of workflow constructs but are independent of specific workflow languages.

Patterns might accelerate the designing of process models, reduce their modelling time, foster reuse of processes and its parts, and improve the modelling quality by simply being instantiated or customized (Gschwind, Koehler, & Wong, 2008; N. Russell, Arthur, van der Aalst, & Mulyar, 2006; van der Aalst, ter Hofstede, Kiepuszewski et al., 2003). An early approach for reusing processes and parts based on case-based reasoning is described in (Kim, Suh, & Lee, 2002) and (Madhusudan et al., 2004).

Patterns allow for a more effective communication within communities and provide better conciseness leaving less space for ambiguities (Buschmann, Henney, & Schmidt, 2007; Gschwind et al., 2008; Medicke & McDavid, 2004; Tran, Coulette, & Dong, 2007). Patterns might range from very simple to very complex and cover the behaviour that might be captured within most business process models. LPM encourages the reuse of known process patterns in order to support the modelling of the control flow perspective on different business domains. The crucial point in applying patterns to LPM is that the targeted business users often don't understand the patterns. The patterns have to be applied in the background, hence, the user doesn't necessarily have to be aware of this application. Anyway, the pattern application should support the users in modelling their processes.

Patterns and templates
- Allow for reusing processes and parts of it - Ensure the soundness of process parts

4.5.2 Structuring Patterns and Templates

The workflow patterns as defined by van der Aalst et al. (van der Aalst, ter Hofstede, Kiepuszewski et al., 2003) are used as process patterns in order to support modelling the control flow perspective. Besides the control flow patterns, patterns for the data flow, resources, and exception handling exist. In this thesis, the control flow patterns are focused. The proposed proceeding of applying these patterns to support modelling might be transferred to other kinds of patterns as well.

The workflow patterns from van der Aalst et al. might be complemented by more coarse-grained sets of process elements. Further, context or business information might enrich these sets. Therefore, *workflow templates* combine workflow patterns, cover a certain business functionality and might be valid in a specific context. Table 22 specifies the distinction between patterns and templates valid in this thesis.

Table 22: Distinction between patterns and templates

In this thesis the following distinction between patterns and templates is defined:

Patterns:

- Comprise a set of process elements describing a certain behaviour
- Are in line with the workflow and data flow patterns defined in (van der Aalst, ter Hofstede, Kiepuszewski et al., 2003)

Templates:

- Might include one or more patterns
- Might be enriched with context information
- Might be enriched by business information

Workflow templates might cover almost complete processes, however, the templates do not contain a start and an end element. The processes composed by one or more workflow templates are guaranteed to be functionally and syntactically sound. A definition of soundness of Gschwind et al. (Gschwind et al., 2008) is considered: *“By soundness of a process model, we mean the absence of deadlocks and lack of synchronization.”* Even for unstructured processes, the application of patterns according to predefined rules guarantees soundness of processes (Gschwind et al., 2008). Certain workflow templates might be enriched with specific information in order to be applicable to different business domains. In order to identify a pattern or a template in a process model any process element is annotated in order to indicate the assignment to a pattern or a template.

4.5.3 Applying Patterns and Templates to LPM

In this thesis, three different scenarios are considered to apply process patterns and templates. In the first scenario the user might browse a repository to retrieve patterns and templates. This supports the user in avoiding modelling from scratch. The second scenario is the refinement of activities with patterns or templates, the third scenario is the completion of processes according to patterns and templates. In all scenarios, the users benefit from the support to model sound processes.

Reuse existing patterns and templates

The workflow templates are defined by users and stored in a template repository. Once business processes are modelled through the LPML, workflow templates and process patterns might be incorporated as first class modelling elements. As workflow templates are sound process parts, it is also possible to model novel processes by simply instantiating and customizing a template. A collection of sample process patterns can be found in (Malone, Crowston, & Herman, 2003).

A process editor for LPM has to implement a functionality that makes the patterns and templates visible in a process. Furthermore, the patterns have to be explained to the user. Semantic annotations are provided explaining the functionality of the pattern and the consequences of applying the pattern.

Madhusudan et al. (Madhusudan et al., 2004) have proposed an approach to reuse patterns through retrieval based on case-based reasoning. The authors state that standards for template representations and associated ontologies, formal guidelines for reuse of these templates, rules for their instantiation or modification, and procedures for their composition into complex workflows are currently non-existent. This thesis defines semantic annotations for the use, integration, and instantiation of patterns and templates.

Refine activities through patterns and templates

Another application scenario of patterns and templates is the refinement of process elements. Gschwind et al. (Gschwind et al., 2008) describe an approach of applying patterns during the process modelling to edges. The authors differentiate three scenarios on how to insert patterns into an existing workflow, the application of a pattern to a single edge, to two edges, and to a set of edges. For any scenario the authors define how to support the user in applying patterns correctly.

In this thesis this approach is slightly modified. The refinement is not applied to edges but to activities. Hence, unbound activities of the abstract LPML layer might be resolved by tools, such as the composition tool, and be replaced by suitable patterns or templates. This resolution is achieved by matching the semantic annotations of the activity against the semantic annotations that describe available workflow templates in a template repository. Hereby semantic annotations about functionality, non-functional properties, input and output data, and preconditions and postconditions are analysed. Through the refinement mechanism the user is able to express an abstract activity instead of a detailed subprocess.

The recommendation of process patterns and templates to refine activities is implemented as a two-step proceeding. First, by analysing the information in the already existing activities, patterns are checked that might potentially be recommended. In the second step, the similarity of the found patterns to the existing activity description is checked.

In this second step, the recommender service calls another service to measure the similarity of the found patterns. This similarity measurement service returns similarity values for each pattern. Based on these values, the recommender service selects the pattern with the highest similarity value and returns the recommendations. A recommendation is given for every pattern retrieved in the first step. The recommendation comprises the original process model, the modified process model based on the recommendation, a numeric value indicating the quality of the recommendation, information about the applied pattern, and the recommendation type.

A process editor for LPM has to implement a wizard in order to guide the user through the correct application of the pattern. This wizard guides the user in applying the pattern in a step-by-step manner and explains the consequences of the pattern integration.

Completion of processes with patterns and templates

The third application scenario covers the completion of processes with patterns and templates. This is the most complicated option of applying patterns. Since the business user is not expected to be able to select fitting patterns, this step has to be performed automatically. Hereby, tools have to understand the semantics of the modelled process as well as the trade-off of applying the pattern (Kircher, 2007). In this thesis, the patterns are not automatically applied but proposed through a recommendation mechanism in order to support the user in modelling sound processes. The recommendation mechanism proposes the next modelling activities stepwise. It is important for users to provide support in small steps the user might understand. It might rather confuse the user if a complete pattern would be automatically applied.

As aforementioned, Madhusudan et al. (Madhusudan et al., 2004) have proposed an approach to reuse process models through case-based reasoning. In their work, the authors describe the tagging of cases, the storage, and the retrieval using a similarity algorithm. This proceeding builds the basis for the proposed retrieval of process patterns and templates described in this thesis.

As described above, for the activity refinement the recommendation to complete processes is as well implemented in a two-step proceeding. After analysing the existing part of the process model patterns are retrieved that potentially might complete the process. In the second step as well, the similarity of the found patterns is compared to the existing process part.

The first step of retrieving potentially fitting patterns is controlled by a recommender service. This service queries the pattern repository service to retrieve available process patterns. The retrieved process patterns already roughly match the requirements formulated in the query. This limits the set of retrieved patterns in order to reduce the following effort to measure the similarity to the modelled process. The second step is performed identically to the proceeding for the activity refinement.

Similarity measurement

The similarity measurement of process patterns to activities and to process parts is an important step for the quality of the recommendations. Ehrig et al. (Ehrig, Koschmider, & Oberweis, 2007) describe an approach to semi-automatically detect synonyms and homonyms of process element names. A prerequisite for their work is that the element names are based on ontologies. Following the approach of Ehrig et al., this thesis measures similarity on a syntactic level based on the edit distance, on a linguistic level based on the senses of terms, and on a structural level evaluating the context of a term or an element. However, the similarity measurement is not a core part of this thesis. Hence, the work performed by Ehrig et al. is referred for a more detailed analysis.

4.5.4 Patterns and Templates in the LPML Metamodel

In the context of the LPMS, process patterns and templates are not executable since they miss a start and an end element. Hence, there’s no syntactic invocation description defined. This makes the traditional discovery based on the syntactic description impossible. Like processes, the patterns and templates are described semantically in order to make them retrievable. In addition, annotations of type *metadata* are provided describing the use, integration, and instantiation for users searching manually for patterns and templates in a repository.

The activity refinement through patterns and templates and the process completion recommendation makes as well use of the semantic annotations. The annotations of activities and existing process parts are compared to the annotations of patterns and templates in order to select the appropriate one. The comparison might be based on all annotation types, such as functional categorization, non-functional properties, input and output data, or preconditions and postconditions.

In order to identify a pattern or a template in a process model, any process element is annotated in order to indicate the assignment to a pattern or a template. Templates and patterns are similar to processes as their descriptions are stored in a common repository and might be referenced by a URI. Patterns and templates are indicated in the metamodel of the canonical LPML format. There's no explicit symbolization in the graphical LPML layer. In order to distinguish between processes, templates, and patterns, two annotations are added to the process element. The flag *isTemplate* is set to true, if a template is described. The flag *isPattern* is set to true, if a process pattern is described. Furthermore, a reference *patternLocation* and *templateLocation* is added as attribute indicating the URI of the pattern respectively template. In order to indicate whether a process element belongs to a pattern or a template the attributes *templateReference* and *patternReference* are included in the *ProcessElement*.

The Process Editor implements a functionality that makes the patterns and templates visible in a process. This functionality uses the flags *isPattern* and *isTemplate*. The semantic annotations explaining the functionality of the pattern and the consequences of applying the pattern to the user rely on the annotation type *metadata*.

4.6 Goals as Abstract Service Descriptions

4.6.1 Benefits of Goals

Process activities are often directly bound to services or operations at design time. For the business users targeted by LPM, the service functionality and binding might be difficult to understand. Further, it might be difficult to decide what kind of services to bind, e.g. a WSDL or a REST service. Rather, a construct is needed that states the users' perspective, abstracts from the implementation, specifies a kind of service category, and expresses the user's needs in terms of desired outputs or environment changes. This construct is called a goal. The definition of goals in this thesis is as follows: *Goals are unbound activities, provide requirements to services, and represent categories of services*. Goals are bound to a particular service either later at design time or at runtime by a discovery engine and composition tools. This keeps the process models more flexible and agile. Goals support users in modelling the control flow perspective. The goal definition of this thesis differs from the traditional definition in the literature, where goals are defined for the entire process (Kueng & Kawalek, 1997; Neiger & Churilov, 2004a, 2004b).

Goals
- Formulate process and service requirements, rather than concrete technical specifications - Are more intuitively understandable for business users

4.6.2 Structuring Goals

Goals describe from a client's perspective a set of web services that would potentially fulfil the user's needs. In terms of the WSMO specification, a goal has the same characteristics as a service description. These characteristics describe importing ontologies, mediators used, non-functional properties, capabilities, and interfaces. The capabilities comprise among others preconditions, assumptions, postconditions, and effects. Hereby, assumptions and effects describe the state of the world before and after the execution of a service. The preconditions and postconditions describe the information space of the service before and after the execution. However, in a goal, the non-functional properties, capabilities, and interfaces are formulated as requests rather than concrete properties (Roman et al., 2006).

In order to formulate the WSMO goal and service specification in a more lightweight way, WSMO-Lite has been introduced dedicated to build an ontology base for WSDL and REST services. A potential goal definition has to be in line with the WSMO-Lite service definition for service discovery reasons. The goal description is mapped to descriptions of existing services in order to find fitting services. Keller et al. (U. Keller et al., 2004) are referred for a specification of discovering services for goals based on WSMO descriptions. Hence, for LPM, in order to be in line with WSMO-Lite service descriptions, another goal definition is needed. However, WSMO-Lite currently doesn't provide a goal specification.

4.6.3 Applying Goals to LPM

For LPM, a new kind of goal is defined, the LPM goal. This LPM goal references the WSMO-Lite ontology. The main difference of the LPM goal to the WSMO goal is the demission of assumptions and effects and the introduction of inputs and outputs. Inputs and outputs represent sets of variable names, such as strings. In contrast, preconditions and postconditions are logic expressions on an abstract level. These logic expressions might be formulated as axioms in semantic description languages, such as RDF or WSML. The axioms might include the input and output variables. Hence, the input and output variable names are subsets of those variables defined in terms of axioms of the preconditions and postconditions. The use of input and output variables seems appropriate for LPM. Preconditions and postconditions might be seen as refinement of the names of inputs and outputs and might be set as optional properties thus.

A LPM goal comprises the following characteristics:

- *Inputs*: The set of input variable names
- *Outputs*: The set of output variable names
- *Preconditions*: The information space before a service execution, formulated as logical expressions
- *Postconditions*: The information space after a service execution, formulated as logical expressions
- *Functional Classification*: Assigns the required functionality to a set of functionality descriptions. The classification might comprise a hierarchy of classes.
- *Preferences* over non-functional Properties

The non-functional properties might be used for ranking the goals according to the users' preferences. A LPM goal is a class and might be understood as an abstract classifier for fitting SWS. Hierarchies of goals might be built comprising subgoals as refinements of goals. An instance of the goal class represents a concrete goal providing concrete instance references to some of its optional properties. They might be provided by the modeller, but they could also be derived from domain specific contextual information and tools. In particular, business users benefit from the goal approach. They might browse and inspect available goals stored within a goal registry. The goals are more intuitively understandable than mere service descriptions.

Binding a service to a goal

Goals represent abstract classifiers of services. In order to execute processes the goals have to be instantiated with services. An important aspect is thus the discovery of appropriate services. LPM allows for discovering and binding services at various stages to achieve flexibility in the service selection. Goals serve as a kind of placeholder for the late binding of services. To instantiate the goals, a discovery engine reads the goal descriptions, matches it to existing service descriptions, and returns fitting service descriptions. A reasonner might support these tasks.

In the context of LPM, the goal descriptions are transformed into queries that might be processed by a discovery engine. The preconditions and postconditions of the LPM goal specification are transformed into ontology-based preconditions and postconditions in order to be able to map the service descriptions. The complete goal description is then transformed into a query, such as a SPARQL (W3C, 2008) query. Depending on the scenario and for performance reasons, a query generation based on partial information might be considered. The query is processed by the discovery engine in order to find appropriate services or service orchestrations, such as patterns and templates.

Discovering services based on goal descriptions might be achieved differently. The functional classification might be used or the capabilities. Table 23 presents the various options to discover services based on goals[16]. Furthermore, the non-functional properties might be used which is however not considered in the table below.

Table 23: Overview of service discovery types based on goals[16]

Type of Discovery	Discovery Means	Example
Functional Classification	One class	E.g. Travel Booking
	Intersection of several classes	E.g. Travel Booking and Security
	Union of several classes	Train Booking or Flight Booking
	Arbitrary expression	Unions, Intersections, Negation

[16] The discovery types are described according to the SOA4All goal specification.

Capabilities	Preconditions, postconditions	
Inputs/Outputs	Input/output variable names	

Besides various discovery types, the matchmaking strategy from goal to service descriptions might determine the resulting set of appropriate services. Four strategies might be differentiated taken out of Pedrinaci and Krummenacher (Pedrinaci & Krummenacher, 2009): Strict exact match, relaxed exact match, subcategory match, super-category match. Table 24 explains the matching strategies according to Pedrinaci and Krummenacher.

Table 24: Strategies of matching services to goals

Matching strategy	**Explanation**
Strict exact	The service is associated with exactly the same set of categories as the goal
Relaxed exact match	The service is associated with all the categories of the goal and some others
Subcategory match	The service is more specific than the goal
Supercategory match	The service is more general than the goal

4.6.4 Abstract Service Descriptions in the LPML Metamodel

In the following, the elements for the instantiation of an activity in the process model with a goal and with a service are described. Hereby, the elements are only defined in the metamodel of the canonical LPML format. In order to separate the activity description and its instantiation, the *Conversation* element has been created that associates goals or services to an activity. This approach is similar to the separation of activities (invoke, receive etc.) and their partner link in WS-BPEL. This mechanism allows for late binding, as described in the literature (Adams, Ter, Edmond, & van der Aalst, 2006; Hagemeyer, Hermann, Just, & Rüdiger, 1997; Han, 1997; Sadiq, Sadiq, & Orlowska, 2005; Weber, Reichert, & Rinderle-Ma, 2008). While the *Goal* element references existing goals, the *Service* element provides a list of services. The *Conversation* might have attached a goal and a list of potential services. It references a concrete service that is selected by analysing the semantic annotations, the referenced goal, and the *SelectionCriteria*. The *SelectionCriteria* class is of type enumeration and defines the ranking of services in the service list. In case a selected service is not available, the *ReplacementCondition* as an enumeration type defines when to replace that service. Service selection and replacement at runtime is performed assuming that all implementations of an abstract service have different interfaces or adopt different communication protocols. These mismatches are solved exploiting the semantic annotations of service descriptions as described in (Cavallaro, Ripa, & Zuccala, 2009). The relation of *inputParameter* and *outputParameter* of an activity is specified by the *Parameter* element. Table 25 gives an overview of the *Conversation* element and its attached services and goals. The *Conversation*, *SelectionCriteria*, and *ReplacementCondition* elements reference a context file.

Table 25: Service and goal description

Related elements	**Attributes / Literals**	**Semantic annotations**
Conversation		
Association: Activity, Goal, Service, ReplacementCondition, SelectionCriteria	compositeGoal	Not applicable
Service		
Association: Conversation	serviceReference	Not applicable
Parent: Activity		
Goal		
Association: Conversation	goalReference	functionalClassification, nonFunctionalProperty, precondition, postcondition
Parent: Activity		
Selection Criteria		
Parent: SemanticAnnotation	bestPrice, bestResponseTime, rating	Not applicable
Association: Conversation		
Replacement Condition		
Parent: SemanticAnnotation	Fault, faultAfterRetry, noResponse	Not applicable
Association: Conversation		

In the following pseudo java code extract of a sample process the activity *handlingPayment* is instantiated by the goal *handlingPaymentGoal* and the service *creditCardPaymentService*.

```
// A payment handling activity
Activity handlingPayment = new ActivityImpl();
handlingPayment.setName("Handling Payment");
handlingPayment.setOperation("charge");
// will be the operation of the bound service
handlingPayment.setSynchronous(true);
handlingPayment.setConversation
(handlingPaymentConversation);
p.addProcessElement(handlingPayment);

// a payment handling goal
Goal handlingPaymentGoal = new GoalImpl();
handlingPaymentGoal.setName("Payment Handler");
handlingPaymentGoal.setGoalReference(payment_url +
"handlingPaymentGoal.wsmo");

// a credit card payment service instantiating the payment
handling goal
Service creditCardPaymentService= new ServiceImpl();
creditCardPaymentService.setName
("CreditCardPaymentService");
creditCardPaymentService.setServiceReference(payment_url +
"services/CreditCardPaymentService.wsdl");
```

In order to attach appropriate services to a goal, the goal description has to be used for search queries. Therefore, the semantic annotations of the goals are transformed into queries. The LPMS might use RDFS to represent the semantic annotations. Further, it is assumed, that the service descriptions might be represented in RDFS as well. Hence, the goal annotations are transformed into SPARQL queries. However, the LPMS is not restricted to a specific description language.

4.7 Data Flow Support

For process modelling, various perspectives might be distinguished. As introduced above, the control-flow perspective captures aspects related to timely dependencies between various activities, such as parallelism, concurrency, or synchronization. In addition, the data perspective describes a means of communicating data elements. This covers aspects, such as passing information between process elements or scoping of variables. As per defining patterns and templates for the control flow, patterns and templates for the data flow might be defined. However, the data flow patterns are not main work of this thesis.

4.7.1 Benefits of Data Flow Support

Supporting the user in passing data from one process element to another is a main issue for LPM. Currently, most of the solutions for data flow modelling and execution are implemented for closed environments. For example, SAP has defined the SAP Global Data Types harmonizing syntactic and semantic characteristics of data. Further examples of data harmonization frameworks are ebXML(OASIS, 2001) for electronic business or RosettaNet (RosettaNet, 2010) for an industrial consortium. However, the LPMS aims at connecting heterogeneous web services of different providers and contexts. Often the syntactical and semantic characteristics of data are not aligned. In order to transfer data from one service to another, mapping and transformation mechanisms are needed that are mostly performed by expensive IT experts. The goal of LPM is to achieve a higher degree of automating the data flow and to support the business user to perform a data mapping and transformation. Hereby, the mapping and transformation have to consider both the semantic and syntactic level.

Data flow support
- Promises to overcome syntactic and semantic mismatches of data - Automates data mappings through reasoning on semantic annotations

4.7.2 Structuring Data Flow Support

In case the user has to interact to model the data flow, LPM supports the user in understanding data types and in performing operations on data. In the following is presented how this support is implemented.

Russell et al. (N. Russell, ter Hofstede, Edmond, & van der Aalst, 2005; N. Russell, ter Hofstede, Edmond, & Aalst, 2004) structure the data perspective according to the following characteristics.

- *Data visibility* covering how process elements might view data elements
- *Data interaction* addressing the communication of data between process elements
- *Data transfer* describing the manner of passing data
- *Data-based routing* focussing on the interaction of data flow and other perspectives, such as control flow

In this thesis, the data flow structuring focuses on data interaction and data transfer. Data visibility depends on the parameter type. Any service parameters are only visible by the according service, globally visible parameters are represented by context information or might be referenced by semantic annotations. The data flow implementation for LPM is designed based on the following options as described in (N. Russell et al., 2005) and (N. Russell et al., 2004).

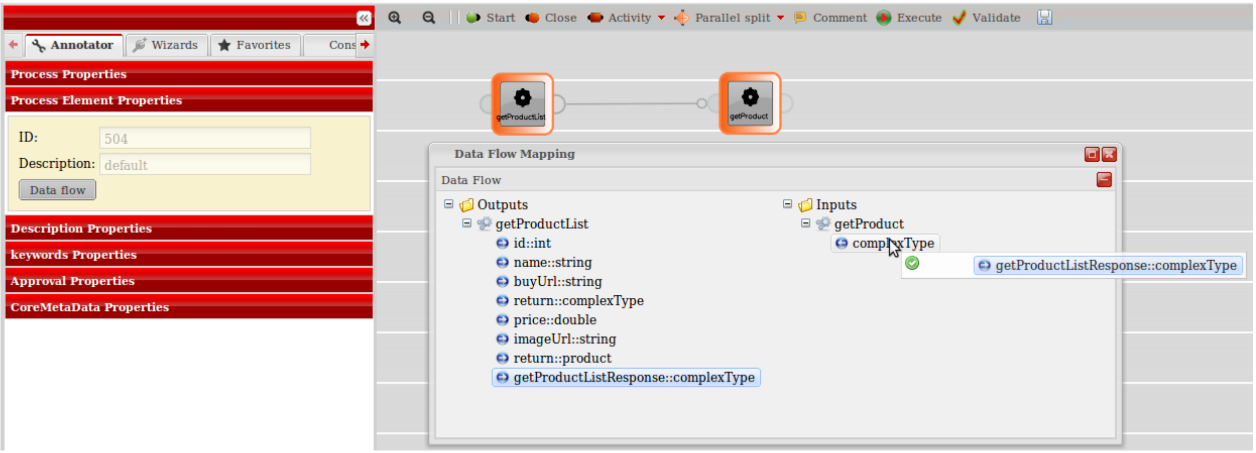
- *Distinction of control flow and data flow vs. integrated flow vs. no data passing*: LPM differentiates between control flow and data flow. The user has to model the control flow in any case. In addition, the data flow might be optionally specified, if required. The data elements are passed between activities, no explicit global shared data is provided for a process.
- *Data interaction*: The data interaction might be simple in case of an output parameter becoming an input parameter of the following activity. More sophisticated data interaction is managed by connectors that manipulate data.
- *Data transfer*: The data mediation is easy for a couple of primitive data types, e.g. transforming *Char* to *String* or vice versa, *Integer* to *Flow* or vice versa, etc. However, for other primitive data types the transformation is not that easy. Further, the transformation of complex types has to be specified individually for any case. The data might be passed by value or by reference. Mapping between activities have to exist, in case of an equal data type (syntactic mapping) and meaning (semantic mapping) the value might be passed, otherwise a transfer by reference is required. Another option is to copy values into the spaces of other services. In case the data of two connected services is not coherent, a data transformation has to be implemented by the process execution engine.
- *Data-based routing*: LPM implements data-based routing through preconditions and postconditions of services. Further, patterns to specify the process routing are implemented through connectors.

4.7.3 Data Flow Support for LPM

As aforementioned, the data flow modelling covers both a semantic and syntactic mapping. The following proceeding describes how the data flow modelling is performed for LPM. This proceeding is adjusted from Lecue et al. (Lecue, Salibi, Bron, & Moreau, 2008). A data flow from process element A to process element B is assumed. The goal is to dynamically generate mappings between service parameters based on semantic connections.

- First, the input parameters of process element B are read. A file of the semantic descriptions of the input parameters is created.
- In the second step, a file of the semantic descriptions of the output parameters of process element A is created.

- Afterwards, the semantic descriptions are mapped. This could be performed for all parameters or according to predefined selection criteria. The mapping is performed through a similarity measurement of parameters based on an ontology. In addition, the semantic consistency of data is checked through reasoning in order to figure out the semantic feasibility. In case the mapping is semantically not feasible, the user has to interact in order to manipulate the data leading to a semi-automated mapping.
- For all identified matchings the syntactic mapping is checked in the next step. The syntactic mapping is performed by the execution engine since reasonners performing the semantic mapping do not support any transformations. Hereby, the information provided by the semantic mapping is used. For example, an Extensible Stylesheet Language (XSL) transformation file (W3C, 2007b) might be created dynamically through listing the XSD (W3C, 2004e) mappings. Afterwards, the XSDs and the data types might be transformed, if needed. In the optional final step, the values are transformed.



ID: 504
Description: default

Figure 19: LPM tool support for data mapping

The data mapping is performed automatically in LPM in order to keep the user free from mapping details. In case the automatic mapping is not feasible, a tool supports the user in performing the mapping manually. Figure 19 shows a snapshot of the mapping tool.

Besides purely control flow oriented constructs, LPM provides a couple of data flow oriented constructs. These constructs support service composition and sophisticated data operations, such as mashups do. To this end, a list of operators, the so called connectors in the context of LPM, is introduced to model and execute data manipulation. The user might easily integrate the connectors in the process model in order to specify a service mapping. By doing so, this approach differentiates from existing approaches that handle mediation by dynamically defining mediators (Joerg Nitzsche & Norton, 2009). Such an approach would require deep knowledge in ontologies that the typical users of LPM don't have. The connectors have to consider both semantic and syntactic mapping of the activities. Furthermore, the connectors allow for the specification of the kind of connection between the services. In case the user requires more sophisticated data manipulations between services, he might use external built-in services provided by the process editor. A detailed listing and description of the connectors is given in section 4.7.4.

4.7.4 Data Flow Connectors in the LPM Metamodel

As introduced in section 4.7.2 and described in (Schnabel, Xu et al., 2009) the data flow handling requires several steps. In this section is described how these steps are implemented by the LPMS for an automatic handling. If an automatic handling is not feasible, the user is supported to provide missing information for completing the data flow.

The elements for the data flow handling are only represented in the metamodel of the canonical LPML format. In order to keep the graphical LPML layer simple, the data flow is not explicitly depicted.

Semantic matching

The first steps for data flow handling cover the semantic matching of parameters. Thus, files are created comprising the semantic service and parameter descriptions of the services to be connected. In the context of the LPMS, these files are lists of the semantic annotations of process elements. For example, in terms of the LPML activities, these annotations describe the input and output parameters. The semantic annotations might be represented by RDFS triples. The RDFS statements are based on ontologies, such as WSMO, WSMO-Lite, or domain ontologies. The LPMS is not restricted to a specific ontology in order to be applicable to various scenarios.

In the following steps, the service parameters have to be mapped on the semantic level. Therefore, the RDFS triples representing the parameters of the two service lists are compared. The composition component in cooperation with a reasonner performs the comparison by measuring the similarity. A standard similarity measurement algorithm is applied to compare the RDFS triples. In case the service parameter descriptions refer to different ontologies, in addition, ontology mapping information has to be considered.

Syntactic matching

In case a parameter matching on the semantic level has been identified, the matching on the syntactic level has to be ensured. The syntactic similarity check compares the syntactic service description files (e.g. the XSDs and WSDLs) comprising the data types of the service parameters. In case of different data types a transformation mechanism is triggered. This transformation mechanism uses the information provided by the semantic matching. Besides the data type mapping, the values and the value meanings have to be in line. For example, the country *Germany* might be coded as "Germany" or as "GER" depending on the context. Hence, the transformation mechanism has to consider a value mapping as well, again using the information provided by the semantic matching.

The execution engine performs the syntactic mapping, e.g. through the copy/assign mechanism in WS-BPEL combined with XPath (W3C, 1999) and XQuery (W3C, 2010) processes. Hereby, the XML data of one service is passed to the other service. If the automatic mapping is not possible the process editor provides a tool for the manual mapping. Another option is to dynamically create XSLT scripts for the transformation. The work described by Lecue et al. (Lecue et al., 2008) is referred for further details of this approach.

Connectors supporting more complex data manipulations

The LPMS and hence, the LPML provides a list of operators supporting more complex data manipulations. In the following, only input and output based data manipulations are considered. A SPARQL-based mechanism to manipulate both RDF data types and values required (as inputs) and provided (as outputs) by services is provided. To this end, the *Connector* element is responsible for the data mapping between services. The following specifications are provided by the latter element:

- *Merge* takes an arbitrary number of input parameters and composes them to an output parameter.

- The *Split* receives an input parameter and produces two or more identical output parameters.

- The *Count* operator counts the number of inputs and outputs

- The *Filter* operator might be used to include or exclude items from an input. Therefore some SPARQL rules might be created on top of the language to compare inputs to values the end-user specifies.

- The *Reduction* operator returns a specified number of items from the top of the input and hence limits the number of items in the output. Also a random item might be selected.

- The *Sort* operator sorts an input by any item element, such as title or description. Items might be sorted in either ascending or descending order.

- The *Loop* operator introduces the idea of sub-data processing. Any other operators could be inserted inside the *Loop* operator. Any input is provided to the *Loop* operator, the sub-data processing is run once for each item in the input.

- In case the data required by the end-user is deeply placed in the description, such as nested inputs, the *Sub-Description* operator might be used to extract and select some sub-descriptions from the input.

- The *Sub-Description* operator is the reverse of the *Aggregation* operator that aggregates descriptions of an input into a more general input.

Each connection (operator, service) has the following optional attributes regarding the data passing: Rounding-up, rounding-down, truncating. These attributes are required for cases when the data provided and consumed is not of the same data type. Hence, a process of rounding-up, rounding-down, or truncating might be required in some cases.

Connector generation and integration

All described operators are considered as semantic and syntactic mapping elements in the LPML. The data they manipulate are propagated through the user and the tools for enhancing the abstract process model.

Ideally, the generation of the connectors is automatically performed by the composition component of the LPMS. Based on the information of the semantic annotations of the parameters and the semantic mapping, the connector type might be determined. In case the connector cannot completely be generated automatically, the user interacts through a wizard implemented in the process editor of the LPMS to support the connector generation.

More experienced users might select and integrate the connectors themselves. Thus, from the user perspective, the aforementioned list of connectors is available through a toolbox in the process editor (such as in Yahoo!Pipes[17]). Besides simply drawing connections from outputs of services to inputs of subsequent services, the user might optionally specify the kind of connection between the services through the connectors. This proceeding is in line with other Mashup editors, stating that the output is sent to this input. In that way, the user could easily drag and drop the appropriate connectors and then link them to the services to be included in the final composition. All these connections are stored through the LPML description of the composition.

Alternatively, in case the user requires more specific or advanced data manipulation within the process, the process editor provides a functionality for dragging and dropping an appropriate, external built-in service in order to achieve the required specific data manipulation. The built-in services are included as *Activities*. For any connector a semantic data mapping component checks the semantic consistency of data connections through reasoning based on the data type ontology. In case the mapping is either semantically or syntactically not feasible, the process editor is called by the composition component. The process editor then interacts with the user in order to fix the mapping issues.

The composition component supports 1-to-1 connectors, namely exact and plug-in matching. Further, subsumption and intersection matching are supported that imply merging or aggregating connectors. These merging strategies however, are not part of this thesis.

The connectors created by the composition component might optionally be visualized in the graphical process model. If the connector cannot be created automatically, the human modellers will be able to interact with the process editor to complete the connector specification.

[17] http://pipes.yahoo.com/pipes/

Table 26 summarizes the elements for the data flow handling. All elements for the data flow handling refer to a context file.

Table 26: Elements for the data flow handling

Element	Related elements	Attributes / Literals
Connector	Parent: ProcessElement	IDName, URISemanticMapping, URISyntacticMapping, URIListInputParameters, URIListOutputParameters, TruncatingElement, connectorType, controlFlowConnector
	Association: Activity, Parameter, ConnectorTypeEnumeration	
	Chlidren: Merge, Split, Loop, Filter, SubDescription, Sort, Count, Reduction, Aggregation	
ConnectorTypeEnumeration	Association: Connector	Exact, PlugIn, Subsume, Intersection, Disjoint, Abduction
Merge	Parent: Connector	
Split	Parent: Connector	
Loop	Parent: Connector	NumberOfLoop
Sub Description	Parent: Connector	ExtractionRule, URIListExtractedConcept
Filter	Parent: Connector	Rules, Any, All
Sort	Parent: Connector	URISortingConcept, AscendingSorting, DescendingSorting
Count	Parent: Connector	NumberOfElements
Reduction	Parent: Connector	NumberOfElements
Aggregation	Parent: Connector	URIListAggregatedConcept

4.8 Conclusion

In this section, the design principles for LPM, the LPM metamodel, and the representation of the LPM metamodel in two LPML formats have been presented. The principles aim at supporting the business user in modelling and executing processes. In the following, the main aspects of the principles and their impact on the LPM metamodel are summarized:

- Abstraction: The abstraction concept aims at keeping the business user free from execution details. In the context of LPM, two abstraction layers for the LPML are defined. While the abstract graphical layer is dedicated to the business user, the canonical format targets at the machine-communication and serves as ground layer. Both layers use the same metamodel of the LPML.
- Semantic annotations: In order to bridge the gap between business specifications and execution details and to support the provisioning of execution-related information for processes and services, semantic annotations are attached to processes and its elements. These semantic annotations are more intuitively understandable for business users and allow for the definition of requirements rather than specifying concrete execution information. In the context of LPM, the semantic annotations will provide information about discovery, composition, and execution of processes and services.
- Context-awareness: In order to flexibly and dynamically adjust processes to the environment, context information is integrated into process models. The context-awareness allows for late binding, processes might be adapted based on context information, or process structures or the naming of elements might be proposed based on the context. The Context-awareness for LPM means that any artefact of the process model is linked to a specific context.
- Reuse through patterns and templates: Patterns and templates might accelerate the designing of process models, reduce their modelling time, foster reuse of processes and its parts, and improve the modelling quality by simply being instantiated or customized. For LPM, patterns and templates are used to avoid modelling from scratch by browsing a repository to retrieve patterns and templates, to refine activities with patterns or templates, and to complete processes according to patterns and templates.
- Goals as abstract service definitions: Instead of implementation-oriented service descriptions, goals state the users' perspective, abstract from the implementation, specify a kind of service category, and express the user's needs in terms of desired outputs or environment changes. For LPM, goals are used as unbound activities that are bound to a particular service either later at design time or at runtime by a discovery engine and composition tools.
- Data flow support: Besides the support for modelling the control-flow, LPM supports the modelling of passing information between activities, the data flow. To support the data flow modelling, a list of connectors is introduced to model and execute data manipulation. The user might easily integrate the connectors in the process model in order to specify a service mapping. The connectors have to consider both semantic and syntactic mapping of the data.

In order to keep LPM light, event handling is not explicitly supported. For LPM, events are subsumed to activities in the LPML.

The LPML implements the LPM design principles as follows:

- Abstraction layer: The abstraction layer comprises the elements Process, ProcessElement, Activity, Flow, and Gateway. These elements are directly manipulated by the user and might be visualized through graphical symbols.
- Semantic annotations: Semantic annotations support the automated discovery, selection, binding, composition, and execution of services. They are based on an ontology and might be attached to any process element including the process itself. A reasoner resolves the semantic annotations. For some process elements, such as Process, Activity, Gateway, Flow and Parameter, the annotations are mandatory. The user has to provide these annotations comprising information about functional classification, non-functional properties, preconditions, postconditions, metadata, conditions, requirements, constraints, selection criteria, replacement condition, and contextual information.
- Context-awareness: The context defines the environment a process artefact is used in. The LPMS uses context information to discover, select, bind, and compose services through the automated fulfilment of properties through context information, to align and adjust the naming of activities through references to ontology-based business entities, and to adjust the process structure through the context references of the process elements.
- Patterns and templates: The support for patterns and templates is achieved through indicating the affiliation of process elements to patterns and templates.
- Activity instantiation: The activities in the abstract process model have to be instantiated by a goal and by a service for the process execution. Through the separation of the activity description and the instantiation, binding might be done at various stages. The LPML further provides elements for defining service selection criteria and replacement conditions.
- Data flow connectors: The data flow connectors perform a data mapping both on syntactic and semantic level. The more complex data manipulations are covered by special connectors, such as merge, split, filter, or count connectors.

Design rules for the modelling of graphical LPML processes have further been presented.

The LPML does not cover elements for specific execution aspects, such as error or compensation handling. These aspects are not relevant for the design of the language.

However, it has to be ensured that the execution engine implements those aspects based on the LPML metamodel.

According to the design science, the design of an artefact might be specified iteratively. Following this approach, the presented design principles might as well be implemented iteratively in LPM. Starting point should be to implement the handling of semantic annotations for goals and the activity instantiation. The next step for the implementation of LPM should be to implement the support for data flow. Following, semantic annotations for context information should be defined. Afterwards, the handling of patterns and templates should be implemented. The benefit of applying patterns and templates to process models increases with the amount of available patterns and templates in a repository.

After the definition of the design principles, the LPM metamodel, and the two LPML layers, the following section describes how an executable process model is generated through interactions of the user and supporting tools. Therefore, a design process is specified that describes the proceeding to model and enhance executable processes. This design process implements a program to generate executable processes. Further, tooling functionalities that support the user in modelling executable processes are described in detail.

5 THE LIGHTWEIGHT PROCESS MODELLING SOLUTION

LPM aims at enabling business users to model and execute processes. In the previous section, the design principles for LPM, the LPM metamodel, and the implementation of the LPM metamodel in two LPML layers have been introduced. The design principles comprise abstraction, semantic annotations, context-awareness, reuse through patterns and templates, goals, and data flow support.

In this section, the LPMS is presented as a prototypical implementation architecture for LPM. Hence, it is described, how the user is supported in providing necessary information and how the tools for LPM implement the modelling functionality and enhancement steps. This is performed in order to create an executable process specification out of the graphical process model and to eventually execute the process with respect to flexibility and adaptability. The various enhancement steps follow a well-defined design process. This section starts with the definition of the design process in section 5.1. The design process is a program to generate executable processes differentiating actions the user has to perform and actions that are automatically performed by the tools.

Further, the tools for LPM are covered in section 5.2. As aforementioned, both the language and the tools comprise elements and functionalities to implement the LPM metamodel as discussed in the previous section. Section 5.2.1 addresses how the LPML is made available to the tools through an API. The following sections introduce the special LPM functionalities in the process editor (section 5.2.2), the composition component (section 5.2.3), and the execution engine (section 5.2.4).

In the context of the design science, this section describes the second part of the design of an artefact, the development.

5.1 DESIGN PROCESS FOR CREATING EXECUTABLE PROCESS MODELS

In this section, the design process for creating an executable process model out of the graphical LPML model is introduced. This process specifies a program to transform the graphical process model into the canonical format, to enhance the canonical format by execution details, and transform this canonical model into an extended form of BPEL that is a currently executable language. A special focus is on the support for the users to provide the necessary information.

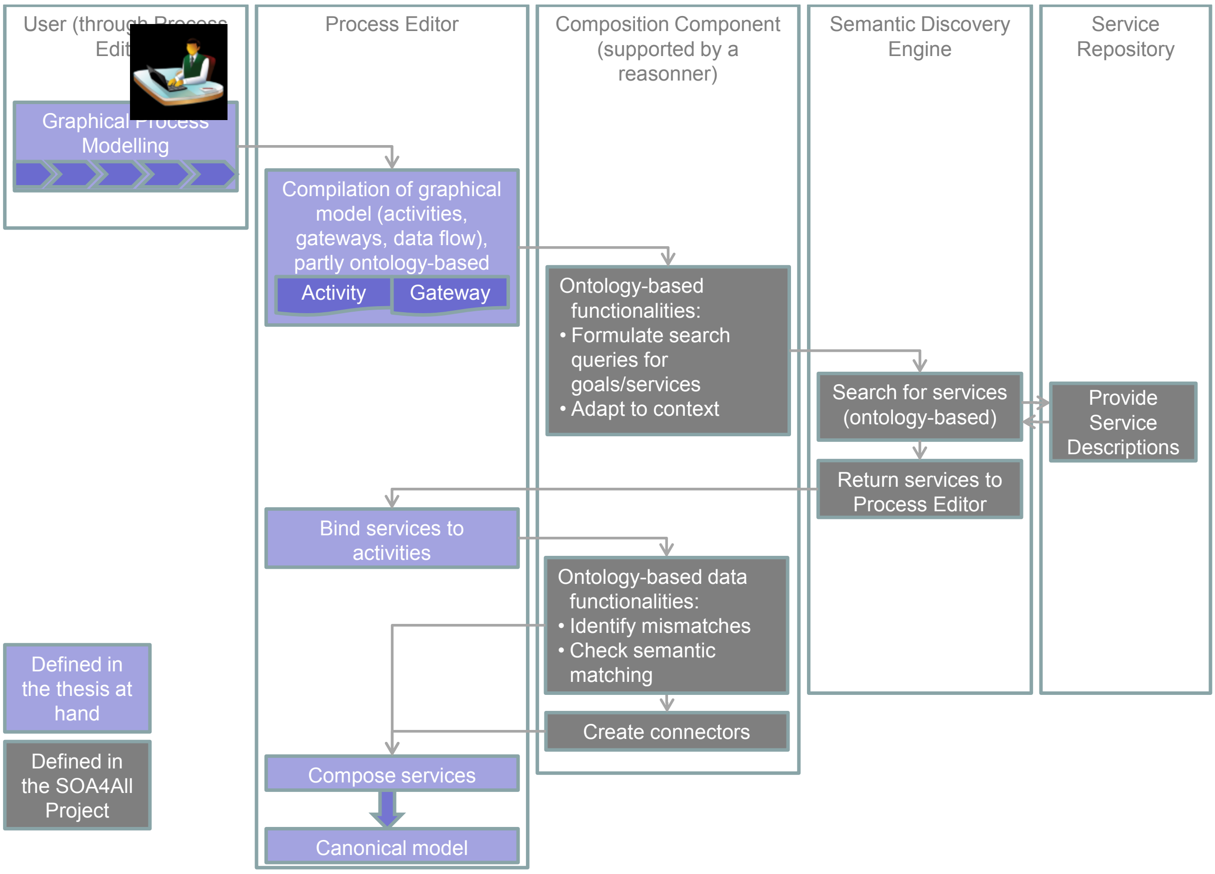
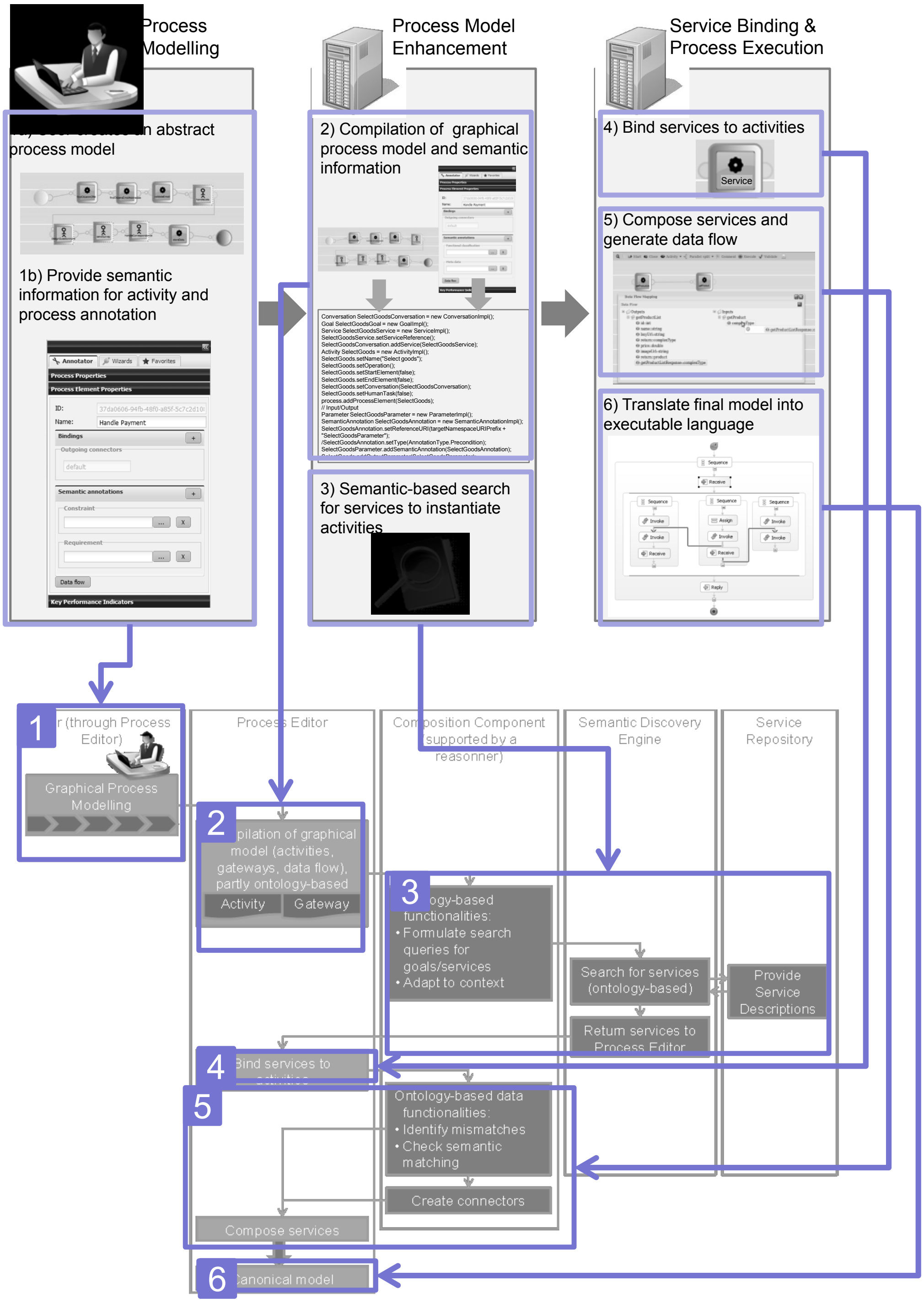

User (through Process Editor)
Process Editor
Composition Component (supported by a reasonner)
Semantic Discovery Engine
Service Repository
Graphical Process Modelling
Compilation of graphical model (activities, gateways, data flow), partly ontology-based
Activity
Gateway
Ontology-based functionalities:
• Formulate search queries for goals/services
• Adapt to context
Search for services (ontology-based)
Provide Service Descriptions
Return services to Process Editor
Bind services to activities
Ontology-based data functionalities:
• Identify mismatches
• Check semantic matching
Create connectors
Compose services
Canonical model
Defined in the thesis at hand
Defined in the SOA4All Project

Figure 20: Process modelling and compilation

Figure 20 depicts this design process that is described in the following subsections 5.1.1 to 5.1.6. In the figure, the activities are assigned to roles that can be software components or the user. Further, the activities are differentiated according to their definition in this thesis or in the context of the SOA4All project. The blue boxes symbolize tasks defined in this thesis. The grey boxes reference tasks performed by technologies and components defined in the SOA4All project. Figure 21 visualizes how the design process is mapped to the approach described in Figure 4 in section 3.3.


Process Modelling
n abstract
process model
1b) Provide semantic information for activity and process annotation
Annotator
Wizards
Favorites
Process Properties
Process Element Properties
ID:
Name:
Handle Payment
Bindings
Outgoing connectors
default
Semantic annotations
Constraint
Requirement
Data flow
Key Performance Indicators
Process Model Enhancement
2) Compilation of graphical process model and semantic information
3) Semantic-based search for services to instantiate activities
Service Binding & Process Execution
4) Bind services to activities
Service
5) Compose services and generate data flow
6) Translate final model into executable language
Sequence
Receive
Invoke
Assign
Reply
1
r (through Process Editor)
Graphical Process Modelling
Process Editor
2
pilation of graphical model (activities, gateways, data flow), partly ontology-based
Activity
Gateway
Composition Component supported by a reasonner)
3
ogy-based functionalities:
• Formulate search queries for goals/services
• Adapt to context
Semantic Discovery Engine
Search for services (ontology-based)
Return services to Process Editor
Service Repository
Provide Service Descriptions
4
Bind services to activities
5
Ontology-based data functionalities:
• Identify mismatches
• Check semantic matching
Create connectors
Compose services
6
anonical model

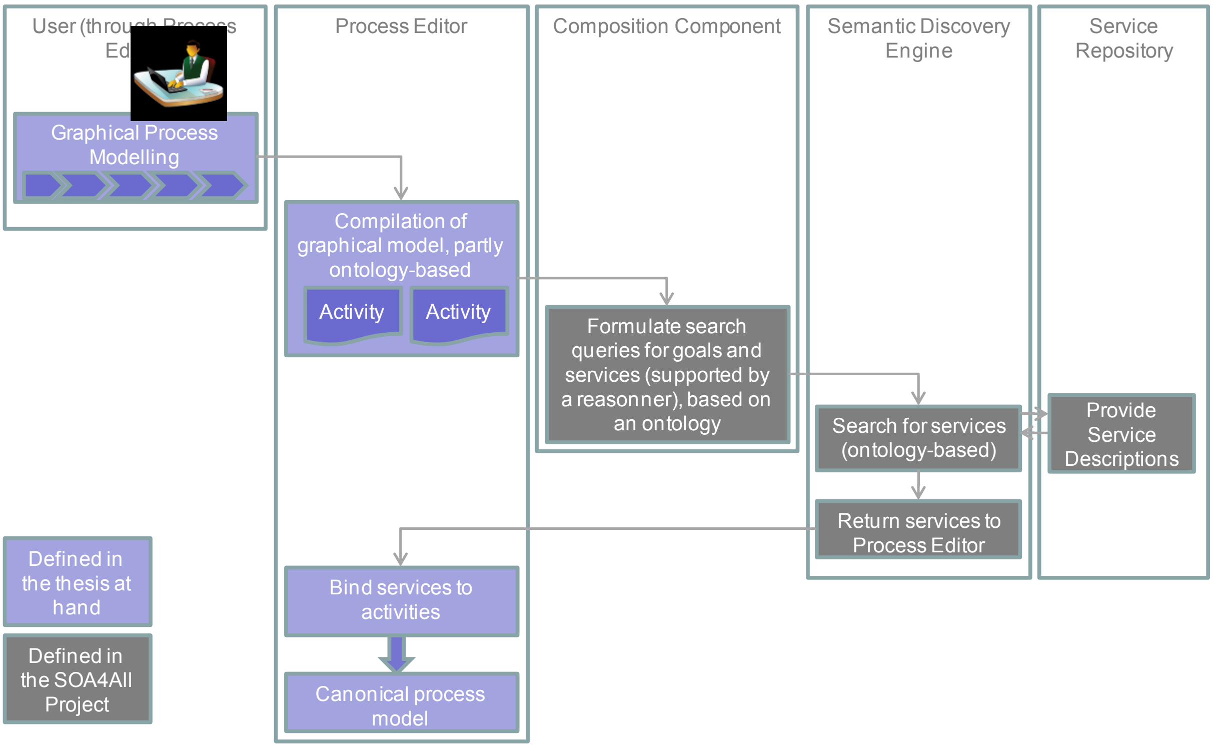
```
Conversation SelectGoodsConversation = new ConversationImpl();
Goal SelectGoodsGoal = new GoalImpl();
Service SelectGoodsService = new ServiceImpl();
SelectGoodsService.setServiceReference();
SelectGoodsConversation.addService(SelectGoodsService);
Activity SelectGoods = new ActivityImpl();
SelectGoods.setName("Select goods");
SelectGoods.setOperation();
SelectGoods.setStartElement(false);
SelectGoods.setEndElement(false);
SelectGoods.setConversation(SelectGoodsConversation);
SelectGoods.setHumanTask(false);
process.addProcessElement(SelectGoods);
// Input/Output
Parameter SelectGoodsParameter = new ParameterImpl();
SemanticAnnotation SelectGoodsAnnotation = new SemanticAnnotationImpl();
SelectGoodsAnnotation.setReferenceURI(targetNamespaceURIPrefix +
"SelectGoodsParameter");
/SelectGoodsAnnotation.setType(AnnotationType.Precondition);
SelectGoodsParameter.addSemanticAnnotation(SelectGoodsAnnotation);
```

Figure 21: How the design process is derived from the approach as presented in Figure 4

The design process is in general performed in six steps that are described in the following.

5.1.1 Step 1 – Process Modelling

The first step is performed by the user. Firstly, he specifies with graphical elements in a process editor an abstract process model. Afterwards, the semantic information is provided for the process, activities, gateways, and the data flow. The tasks to be performed by the user are depicted in Figure 23 and explained in the following.

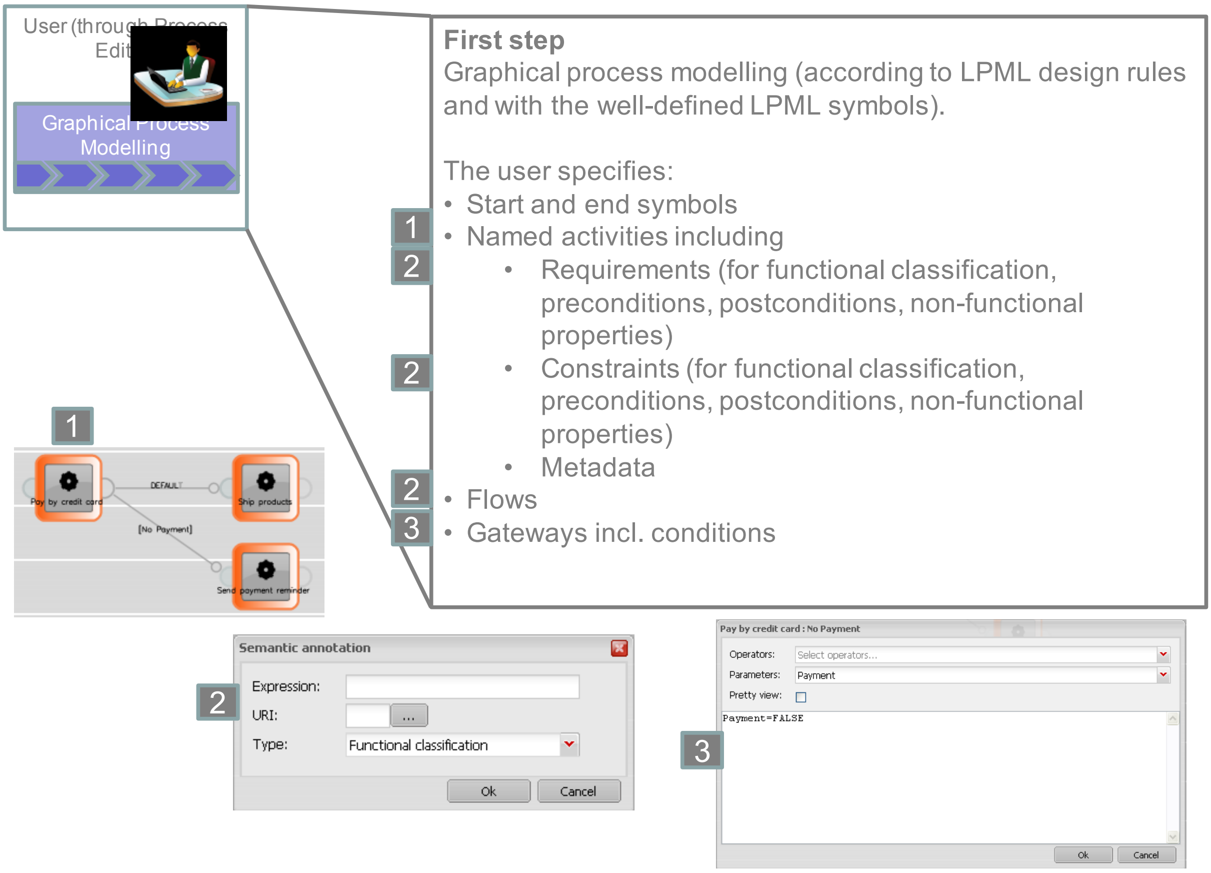


User (through Process Ed
Process Editor
Composition Component
Semantic Discovery Engine
Service Repository
Graphical Process Modelling
Compilation of graphical model, partly ontology-based
Activity
Activity
Formulate search queries for goals and services (supported by a reasonner), based on an ontology
Search for services (ontology-based)
Provide Service Descriptions
Return services to Process Editor
Bind services to activities
Canonical process model
Defined in the thesis at hand
Defined in the SOA4All Project

Figure 22: Activity modelling and service binding

An important part of designing a process model in the context of LPM and hence, creating an executable process model, is the specification of the activities. Figure 22 visualizes an activity-focused view on the LPM design process. Those steps are extracted that are necessary to model activities and to bind services to those activities. The specification of the data flow to compose those services is described in section 5.1.5.

The process model comprises a start element, an end element, activities, and the connections between those elements. The graphical LPM representation, as described in section 4.2.2.2, covers the information that is provided by the user's process model in the process editor (see step 1 in Figure 21). Further, the activities contain a name and semantic annotations about the functionality. These semantic annotations are formulating requirements and constraints, and hence comprise information, such as the functional classification, requirements, constraints, non-functional properties, or metadata (see again step 1 in Figure 21). A wizard supports the user in providing and structuring the needed semantic descriptions. Ideally, the tools then automatically enhance and execute the process model without any more user interaction.



First step
Graphical process modelling (according to LPML design rules and with the well-defined LPML symbols).

The user specifies:

- Start and end symbols
- Named activities including
  - Requirements (for functional classification, preconditions, postconditions, non-functional properties)
  - Constraints (for functional classification, preconditions, postconditions, non-functional properties)
  - Metadata
- Flows
- Gateways incl. conditions

Semantic annotation

Expression:
URI: ...
Type: Functional classification

Ok Cancel

Pay by credit card : No Payment

Operators: Select operators...
Parameters: Payment
Pretty view: □

```
Payment=FALSE
```

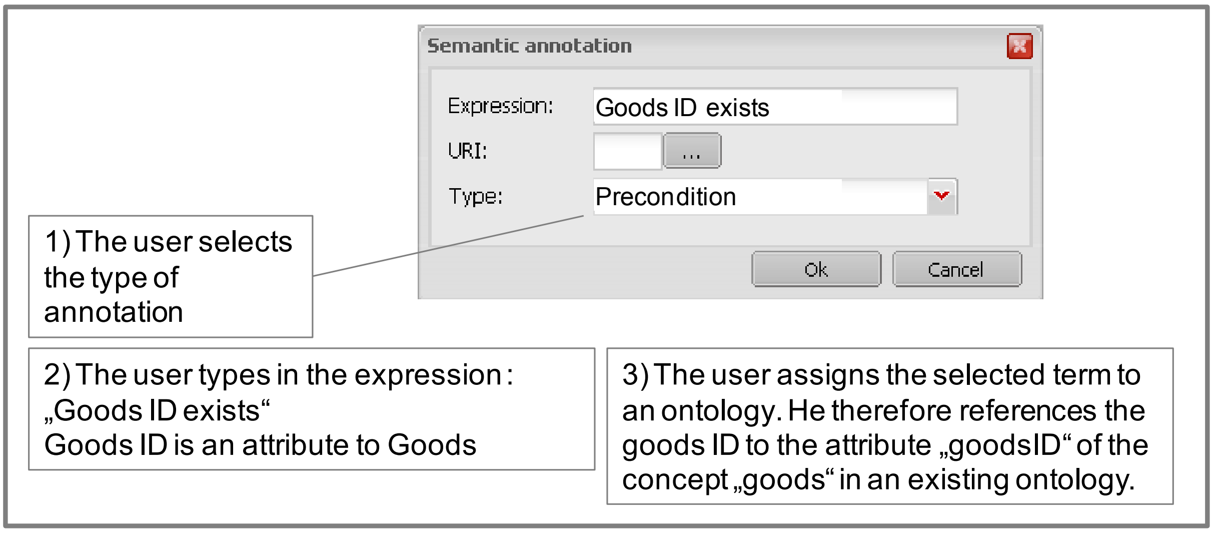
Ok Cancel

Figure 23: Process modelling performed by the user

In the Process Editor, the user is guided to specify as well the semantic annotations. He therefore adds a new semantic annotation specifying the following aspects:

- The annotation type by selecting the type from a predefined list (e.g. functional classification, precondition etc.)
- The annotation expression, such as
  - Select by goods ID (for functional classification)
  - Select goods by barcode (for functional classification)
  - Specify goods amount (for functional classification)
  - A goods ID or a barcode has to exist
  - The desired amount has to be known
- Assignment of the used expression terms to predefined ontological concepts

Figure 24 depicts the proceeding to specify the semantic annotations. In case the used terms cannot be assigned to existing ontological concepts, the user has to request an extension of the ontology or select another ontology.



Expression: Goods ID exists

Type: Precondition

1) The user selects the type of annotation

2) The user types in the expression: „Goods ID exists“
Goods ID is an attribute to Goods

3) The user assigns the selected term to an ontology. He therefore references the goods ID to the attribute „goodsID“ of the concept „goods“ in an existing ontology.

Figure 24: Proceeding to specify semantic annotations in the Process Editor

Figure 25 depicts the gateway-focused view on the design process. Main tasks for the gateway modelling and enhancement are the specification and compilation of the conditions. The specification of conditions is similar to the specification of semantic annotations for activities. The user is supported in defining the relevant variables and the conditions for those variables. For example, the user might specify the relations between variables or values, such as equal, bigger than, less than, string comparisons, etc.

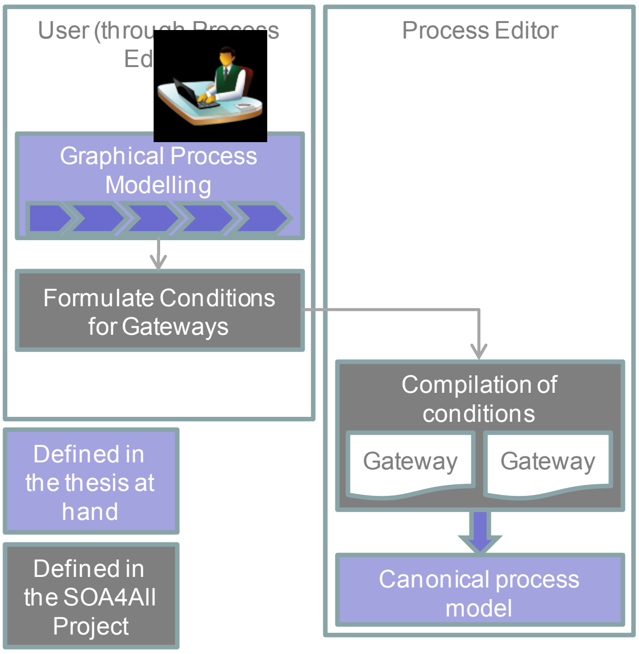

User (through Process Editor)
Process Editor
Graphical Process Modelling
Formulate Conditions for Gateways
Compilation of conditions
Gateway
Gateway
Canonical process model
Defined in the thesis at hand
Defined in the SOA4All Project

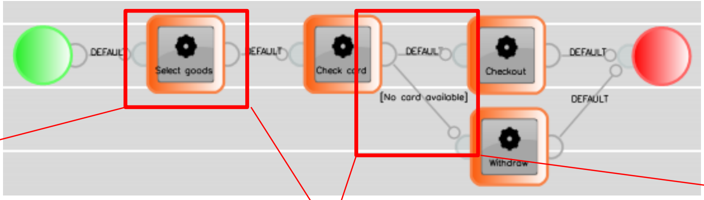
Figure 25: Gateway modelling in the LPM design process

5.1.2 Step 2 – Process Model Compilation and Semantic Annotations

The second step covers the compilation of the graphical process model and the semantic annotations that are in line with service and goal annotations out of the annotations provided by the user. The process editor implements a functionality to perform this step (see step 2 Figure 21). The semantic annotations are generated for the functional classification, non-functional properties, preconditions, postconditions, conditions, selection criteria, replacement conditions, and metadata.

Furthermore, if the semantic annotations are given in XML notation, they will be lifted to RDFS notation in order to be in line with ontology languages. If this step 2 cannot be performed completely automatically, the user will be requested to provide missing information through a wizard.

Figure 26 depicts the code of the canonical model in Java notation that is automatically generated out of the graphical process model.



According code of the canonical process model in Java notation:

```
// ---- Select goods: Activity bound to a conversation
Conversation SelectGoodsConversation = new ConversationImpl();
Goal SelectGoodsGoal = new GoalImpl();
Service SelectGoodsService = new ServiceImpl();
SelectGoodsService.setServiceReference();
SelectGoodsConversation.addService(SelectGoodsService);
Activity SelectGoods = new ActivityImpl();
SelectGoods.setName("Select goods");
SelectGoods.setOperation();
SelectGoods.setStartElement(false);
SelectGoods.setEndElement(false);
SelectGoods.setSynchronous(true);
SelectGoods.setConversation(SelectGoodsConversation);
SelectGoods.setHumanTask(false);
process.addProcessElement(SelectGoods);

// Input/Output
Parameter SelectGoodsParameter = new ParameterImpl();
SemanticAnnotation SelectGoodsAnnotation = new
SemanticAnnotationImpl();
SelectGoodsAnnotation.setReferenceURI(targetNamespaceURIPrefi
x + "SelectGoodsParameter"); // ReferenceURI links the place of the
semantic annotation
SelectGoodsAnnotation.setType(AnnotationType.Precondition);
SelectGoodsParameter.addSemanticAnnotation(SelectGoodsAnnota
tion);
SelectGoods.addOutputParameter(SelectGoodsParameter);
```

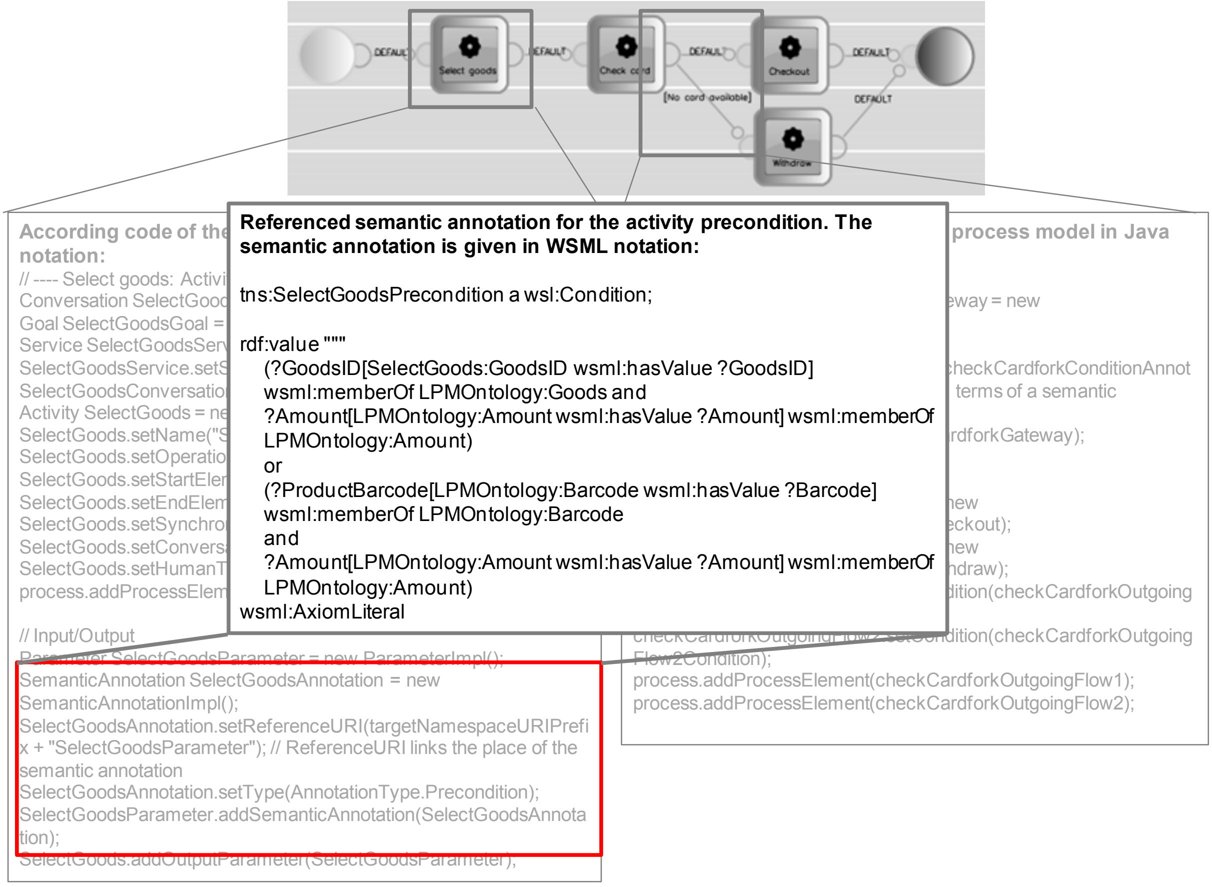
According code of the canonical process model in Java notation:

```
// Check card exclusive fork gateway
ExclusiveGateway checkCardforkGateway = new
ExclusiveGatewayImpl();
checkCardforkGateway.setSplit(true);
checkCardforkGateway.setCondition(checkCardforkConditionAnnot
ation); // This condition is formulated in terms of a semantic
annotation
process.addProcessElement(checkCardforkGateway);

// Check card fork gateway flows
Flow checkCardforkOutgoingFlow1 = new
FlowImpl(checkCardforkGateway, Checkout);
Flow checkCardforkOutgoingFlow2 = new
FlowImpl(checkCardforkGateway, Withdraw);
checkCardforkOutgoingFlow1.setCondition(checkCardforkOutgoing
Flow1Condition);
checkCardforkOutgoingFlow2.setCondition(checkCardforkOutgoing
Flow2Condition);
process.addProcessElement(checkCardforkOutgoingFlow1);
process.addProcessElement(checkCardforkOutgoingFlow2);
```

Figure 26: Automatically generated according code of the canonical process model for the graphical process model

Figure 27 visualizes the code in WSML notation that is generated for the semantic annotation, stored in a separate file, and referenced by the canonical process model.



According code of the
notation:

```
// ---- Select goods: Activi
Conversation SelectGood
Goal SelectGoodsGoal =
Service SelectGoodsServ
SelectGoodsService.setS
SelectGoodsConversatio
Activity SelectGoods = ne
SelectGoods.setName("S
SelectGoods.setOperatio
SelectGoods.setStartElen
SelectGoods.setEndElem
SelectGoods.setSynchro
SelectGoods.setConvers
SelectGoods.setHumanT
process.addProcessElem

// Input/Output
Parameter SelectGoodsParameter = new ParameterImpl();
SemanticAnnotation SelectGoodsAnnotation = new
SemanticAnnotationImpl();
SelectGoodsAnnotation.setReferenceURI(targetNamespaceURIPrefi
x + "SelectGoodsParameter"); // ReferenceURI links the place of the
semantic annotation
SelectGoodsAnnotation.setType(AnnotationType.Precondition);
SelectGoodsParameter.addSemanticAnnotation(SelectGoodsAnnota
tion);
SelectGoods.addOutputParameter(SelectGoodsParameter);
```

Referenced semantic annotation for the activity precondition. The semantic annotation is given in WSML notation:

```
tns:SelectGoodsPrecondition a wsl:Condition;

rdf:value """
   (?GoodsID[SelectGoods:GoodsID wsml:hasValue ?GoodsID]
   wsml:memberOf LPMOntology:Goods and
   ?Amount[LPMOntology:Amount wsml:hasValue ?Amount] wsml:memberOf
   LPMOntology:Amount)
   or
   (?ProductBarcode[LPMOntology:Barcode wsml:hasValue ?Barcode]
   wsml:memberOf LPMOntology:Barcode
   and
   ?Amount[LPMOntology:Amount wsml:hasValue ?Amount] wsml:memberOf
   LPMOntology:Amount)
wsml:AxiomLiteral
```

process model in Java

```
eway = new

checkCardforkConditionAnnot
 terms of a semantic

rdforkGateway);

new
eckout);
new
ndraw);
dition(checkCardforkOutgoing

checkCardforkOutgoingFlow2.setCondition(checkCardforkOutgoing
Flow2Condition);
process.addProcessElement(checkCardforkOutgoingFlow1);
process.addProcessElement(checkCardforkOutgoingFlow2);
```

Figure 27: Automatically generated according code for the semantic annotation

5.1.3 Step 3 – Semantic-based Service Search

The third step is dedicated to find a set of appropriate services to instantiate the activities. The discovery of services might be based on syntactic information, on semantic information, or on both. The discovery on semantic level uses the description of the attached goal found in a separate previous step or the semantic activity annotations in case no goal has been attached.

In order to map a goal to an activity, this goal has to be discovered that meets best the semantic annotations provided for the activity. Therefore, the semantic activity annotations are transformed into queries, such as SPARQL queries, that might be executed by a discovery engine. Since a goal is an artefact on semantic annotation level, the queries have to be formulated as well for the semantic level. Based on these queries, the goal discovery engine automatically searches for an existing goal. Hereby, the queries are mapped to the semantic goal annotations. The goal instantiation is referred to as step 3 in Figure 21.

To retrieve appropriate services, again, queries have to be created. Either the query created for the goal retrieval is used or the semantic goal description is translated into a search query. If the goal description is represented in RDFS, it will be translated into a SPARQL search query. A service discovery engine executes the queries and returns a set of fitting services. These services fulfil the basic requirements concerning functional classification, preconditions, postconditions, and hence input and output data.

Besides the service discovery on semantic level, the syntactic information – if existing - might be used. Therefore, the information about the syntactic service description is transformed into syntactic query statements, such as XQuery statements. These statements are used by the discovery engine to find fitting syntactic service descriptions, such as WSDL or REST interface descriptions. For documentation purposes, the syntactic descriptions might be lifted on a semantic description level.

For each found service the *Service* element is instantiated that contains the reference of the service URI. This is represented by step 4 in Figure 21.

5.1.4 Step 4 - Service Binding

The fifth step is now performed through an interaction of the composition component and the execution engine. These two components select at runtime the best-fitting service out of the created service list. The selection is mostly based on preferences described in the non-functional properties. The execution engine might finally compose the process with respect to adaptations in reaction to various kinds of changes. Hereby, the execution engine applies the *SelectionCriteria* and the *ReplacementCondition* defined at design time. The service selection is represented by step 5 in Figure 21.

5.1.5 Step 5 – Service Composition and Data Flow Generation

As previously described, a prerequisite for service composition in the context of the LPMS is that these services are semantically annotated. These semantic annotations have to be checked in order to figure out the compatibility of two services that are connected through the data flow in a process.

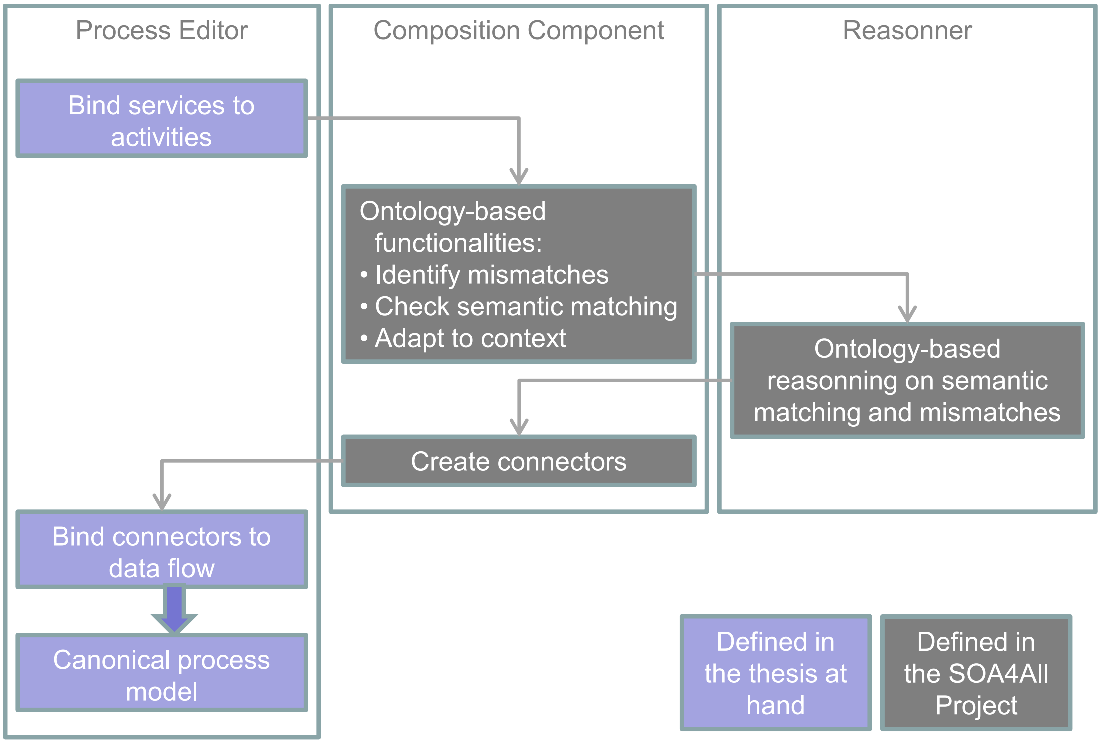

Process Editor
Composition Component
Reasonner
Bind services to activities
Ontology-based functionalities:
• Identify mismatches
• Check semantic matching
• Adapt to context
Ontology-based reasonning on semantic matching and mismatches
Create connectors
Bind connectors to data flow
Canonical process model
Defined in the thesis at hand
Defined in the SOA4All Project

Figure 28: Data flow modelling

Figure 28 visualizes the metamodel view on the data flow modelling procedure. The view starts with the binding of services to activities. Afterwards, the matching has to be checked, data flow connectors have to be generated if required, and the connectors have to be bound to the data flow. A reasoner (as described in section 3.4.2) supports the data matching.

A data flow matching has to be ensured for the following artefacts:

- Parameters: The output variables of the preceding activity have to match the input variables of the following activity in case of a data flow connection.
- Preconditions and postconditions: The Postconditions of the preceding activity mustn't contradict the preconditions of the following activity.
- Non-functional properties: Since the non-functional properties impact the selection criteria for services and the replacement conditions, they mustn't contradict each other.

The compatibility of these characteristics is checked by a reasoner. The reasoner creates queries, again for example SPARQL queries, out of the annotations of the connected services. The queries are used to check the characteristics of each connected service. Then, the results of these queries are analysed and a matching value is returned. If one of the characteristics doesn't match, the further proceeding will be different. If the parameters do not match, a transformation mechanism will be triggered as described in detail in the data flow section 4.7. If the preconditions, postconditions, or non-functional parameters do not match, it will be figured out whether other services fit better. Therefore, another service out of the service set discovered by step 3 is selected and again, the matching checked. The service composition is depicted as step 6 in Figure 21.

In addition, the execution engine encapsulates the executable process model in a service interface that is published in the process repository.

An instantiation of parts of the design process with concrete technologies is presented in Krummenacher et al. (Krummenacher, Domingue, Pedrinaci, & Simperl, 2010). The authors introduce the Minimal Service Model (MSM) in order to have a common service model for WSDL-described services and REST services. This MSM serves as a common model for service descriptions. Another approach is the so called Service Template (ST) according to (Lampert & Pedrinaci, 2010). Like goals, the STs represent the user's perspective, express requirements rather than concrete specifications, and form the basis for semantic discovery.

In their work, Krummenacher et al. propose to map the activity or goal annotations of the LPML process model to the properties of the service templates. Afterwards, SPARQL queries are derived from the RDF-based service templates representing the user's perspective. The SPARQL queries are then executed against semantic service descriptions in a service repository. Since these descriptions represent the service perspective, they are based on the MSM.

5.1.6 Step 6 - Transformation into a currently executable language

The eventual execution of the processes represented through the LPML is performed by an execution engine. Either an engine might be created executing directly the LPML models or the LPML models might be transformed into a standard executable process language, such as BPEL, and executed on an existing engine. In order not to reinvent the wheel concerning workflow execution, the LPMS relies on existing workflow execution engines and modifies these according to its purposes. Hence, the

LPML models have to be transformed into the language the execution engine is capable to execute.

In this thesis, the transformation into an executable language is not described in detail. As aforementioned, a wide-spread common language for executable process models is BPEL. Hence, in order to use an existing process execution engine, the LPML is transformed into BPEL. Within the SOA4All project, an approach has been developed using the Intermediate Model (IM) project[18] that is part of the Eclipse SOA Tools Platform Project[19]. A transformation from the IM to BPEL had already been defined that has been reused and adjusted for the LPMS. A detailed description of the transformation from the LPML to the IM can be found in Ripa et al. (Ripa, De Giorgio, Gorronogoitia, Mos, & Ravoajanahary, 2010). A description of the specific LPM features of an execution engine is described in section 5.2.4.

5.2 Modelling and Executing LPML Processes

In this section is described, how the tools of the LPMS implement the modelling functionality and enhancement steps. This is performed in order to create an executable process specification out of the abstract process model and to eventually execute the process with respect to flexibility and adaptability. Thus, the section 5.2.1 addresses how the LPML is made available to the tools through an API. The following sections introduce the special LPMS functionalities in the process editor (section 5.2.2), the composition component (section 5.2.3), and the execution engine (section 5.2.4).

5.2.1 Tooling Interface

On the programmatic perspective, the LPMS including the LPML needs to be managed by an Application Programming Interface (API) that abstracts and hides the complexity of the LPML elements and their concrete serialization formats to the

[18] See www.eclipse.org/stp/im

[19] See www.eclipse.org/stp

programmer. This API is used by the LPMS components whenever LPML code needs to be exchanged. The API supports the LPML code serialization into an extended form of WS-BPEL that is usable by the execution engine but may also be used by third party tools as it is backwards compliant to RDF. RDF might provide a full semantic representation of the LPML. While the WS-BPEL serialization is obviously limited in terms of LPML-specific features, the RDF version covers all details and contains semantic annotations. The library might also be used to convert the various serializations by simply loading an LPML model and serializing it again. The API is kept generic in order to allow for the implementation of additional serialization approaches.

5.2.2 Process Editor

In this section, the process editor as part of the LPMS is described. Those features are focused that are specific to the implementation of the user support and to the LPML. Since the LPML is a new language, an according new process editor is needed for the LPMS.

The concept of LPM for the tooling perspective means easy deployment, implementation, and execution of processes. The user interface doesn't require high installation effort. This is based on the fact that users are looking for models that enrich existing internet applications. These models should allow for combining the media-rich power of traditional desktop applications with the deployment and content-rich nature of web applications (Allaire, 2002). Hence, the LPMS relies on the principles of a Rich Internet Application (RIA) in order to be easily usable. An overview of RIAs addressing BPM can be found in Le Clair and Teubner (Le Clair & Teubner, 2007). Allaire (Allaire, 2002) describes the characteristics of RIAs. The business value of RIAs is covered by Geelan (Geelan, 2008) and Driver and Rogowski (Driver & Rogowski, 2007). Potential implementations of RIAs are described by Adobe (Adobe, 2008) and by Schmelzer (Schmelzer, 2006).

According to the RIA principles, the process editor comprises two components, one on the client side and one on the server side. On the client side, the process editor is available through a standard internet browser. This requires a minimal installation effort for the user. The client part comprises all major process editing functionality. It allows for creating process models in an interactive way.

The server side is implemented as a set of services to be invoked by the client. Hereby, additional information is requested from the server.

In the following, the new process editor functionalities related to LPM and hence, the LPMS are listed.

- A new process editor for the new LPML: Figure 29 depicts the drawing area on the right hand side, where LPML models can be created.
- Implementation of abstraction layers: The process editor visualizes the graphical abstraction layer to the user. In the backend, the process model is stored in the canonical format as described in section 4.2.2.1. Thus, the editor is based on its own data model that is converted to and from the LPML. The process editor data model comprises additional information about the graphical layout of the process model, such as the position of a process element.
- Drag-and-drop functionality: Various artefacts, such as favourite activities or other modelling elements, might be added to the process model through drag-and-drop functionality. This increases usability of the process editor.
- Support for gateway handling: On the graphical abstraction layer (as described in section 4.2.2.2) no symbols for gateways exist. Rather, a start element might have multiple outgoing flows, an end element multiple incoming flows, and an activity multiple incoming and outgoing flows. If the user draws multiple outgoing or incoming flows, a wizard will pop up supporting the user in specifying the gateway type and potential conditions. Through this wizard the user might select a function from a predefined set in order to create conditions. These functions represent requirements or constraints on properties defining the value or value range for input or output parameters provided by the services. Furthermore, the wizard supports the user in creating valid conditional expressions that might be evaluated at runtime by the execution engine. Valid conditional expressions comprise valid parameters besides the valid functions.
- Context-awareness: As described in section 4.4, the process structure might depend on the context. Furthermore, activity naming and service binding might as well depend on the context. The process editor supports gathering, applying, and visualizing contextual information.
- Data area: Besides the drawing area, a data area supports the user in providing semantic annotations and context information. Figure 29 visualizes the data area on the left hand side.
- Wizards: By default, the process editor implements wizards to recommend patterns and templates, to provide information for semantic annotations, for the gateway specification, and for the modelling of sound process models. For example, if the user splits the flow this wizard will make sure that a corresponding merge is added.

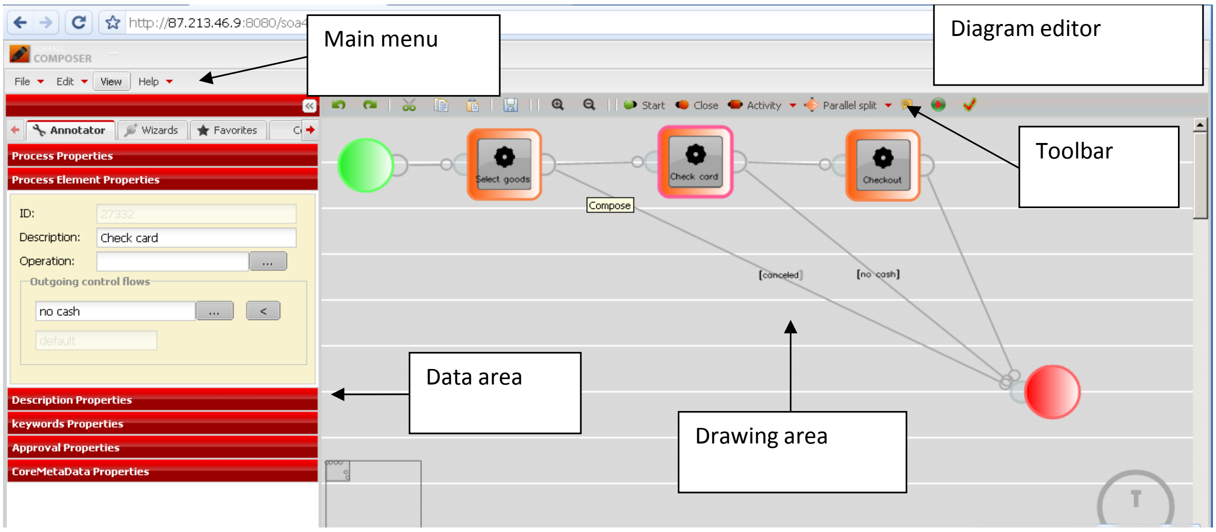
Furthermore, a wizard-creation mechanism is implemented supporting users in creating customized wizards.

- Different modelling modes depending on the user expertise: The process editor implements different support levels. For non-experienced users a guided modelling mode might be selected. This modelling mode comprises wizards for every modelling step. More experienced users might select a free modelling mode having the wizards disabled.

- Favourite lists: In order to support the user in reusing artefacts, favourite lists for services, activities, patterns, and templates are implemented.

- Support for data flow: A special support is dedicated to model the data flow. By default, the data flow is not visualized on the graphical abstraction layer. However, the user might set control flow connections as data flow connections and add connectors through a drag-and-drop functionality. The connectors are implemented by the composition component that uses the input and output data of the subsequent services. Furthermore, a wizard is implemented supporting the user in specifying the connectors in case these cannot be created automatically. In both cases, the connectors are visualized in the process editor. The process editor interacts with the execution engine in order to perform a manual data mapping in case the automatic mapping is not feasible.

- Pattern and template recommendation: The process editor implements a pattern and template recommender service that calculates at design time the similarity of the modelled process to patterns and templates stored in a repository. As described in section 4.5, the recommendations are given for activity refinements and for process completion. Besides the interaction with the process editor, the recommender service interacts with the composition component. The components exchange data about the actually modelled process, the process after modification through the recommender service, the calculated recommendation quality, information about the applied pattern or template, and the recommendation type. In addition, a recommendation viewer within the process editor visualizes the potential changes of the process model performed by the recommender service.



ID: 27332

Description: Check card

Operation:

no cash

default

Figure 29: Data area and drawing area of the SOA4All process editor

An implementation of a prototype of the process editor is performed in the context of the SOA4All project. Detailed information can be found in the documentation of Task 2.6 of SOA4All[20] (Pavlov et al., 2010). Figure 29 is as well a snapshot of this SOA4All process editor.

5.2.3 Composition Component

The composition component is a central part of the LPMS in order to enhance the abstract process models by information that makes the processes executable. The purpose of this thesis is to introduce the main functionality that enables LPM and hence, supports the business user through backend functionality.

Main features

The composition component enables the flexible and dynamic creation, instantiation, and adaptation of processes at design time. Thus, the composition component supports the entire life-cycle of service orchestration from high-level activity specification in terms of process models to discovery and binding of services. In the context of the LPMS, this means binding goals and services to activities, binding services to goals,

[20] See www.SOA4All.eu

resolving activities to process patterns and templates, checking and supporting the matching of activities and services on semantic and syntactic layer, and creating data flow connectors.

The composition component tightly interacts with the process editor, a reasonner resolving semantic annotations, a discovery engine, and the execution engine. It serves as a mediator between these components.

Resolution of semantic annotations through interaction with a reasoner

As introduced in section 3.4.2, the main feature of the interaction with the reasoner is the resolution of the semantic annotations. The resolution of the semantic annotations is needed for binding goals and services to activities and for composing services in terms of data flow mappings. For the LPMS, RDFS representations are proposed for process and activity descriptions, process fragment descriptions, and context information. WSML-based annotations are proposed for axioms and logical expressions that can't be expressed through RDFS, such as conditions.

Besides the resolution of semantic annotations, support for querying is needed. Again, various query formats might be supported, such as WSML or SPARQL queries.

The resolution of semantic annotations, and hence the service binding and the data flow mapping, is implemented as a parametric design process. The parametric design process is described in the work of Mittal and Frayman (Mittal & Frayman, 1989), MacCallum and Yu (MacCallum & Yu, 1996), and Wielinga et al. (B. J. Wielinga, Akkermans, & Schreiber, 1995). Parametric design is a refinement of configuration design (Motta, 1999; B. Wielinga & Schreiber, 1997). In the configuration design, a set of predefined components is selected and combined until a set of requirements and constraints is satisfied. Parametric design assumes a set of functional solution templates and parameters. Hence, the selection and composition is guided and the space of possible designs is restricted.

Basically, parametric design comprises two phases, namely analysis and synthesis. The analysis phase gathers the needed information for the later synthesis and represents this information in a formal way. An ontology of structuring the design might be found in the work of Motta (Motta, 1999). The analysis is conducted iteratively refining the design stepwise. In the context of the LPMS, the information is already well structured in terms of process models, the according elements, and the semantic annotations.

The synthesis phase might be seen as a search in a space of potential solutions (Chandrasekaran, 1990). Starting from an initializing design, multiple intermediate designs are navigated through in various ways in order to reach a final solution. The transitions between the intermediate designs are triggered by applying design operators or transition agents.

The composition component transforms the input data of the LPML process models into a parametric design problem. As aforementioned, Motta describes an approach to structure a parametric design problem.

In order to make the abstract LPML models executable, various tasks have to be performed by the composition component. In the following Table 27, these enhancement tasks, the according parameters, and the value ranges are listed. For all tasks, the non-functional properties and context information influence the preferences, constraints, and requirements as described in the theoretic parametric design approach by Motta (Motta, 1999).

Table 27: Enhancement tasks for the LPMS as parametric design

Enhancement task	Parameter	Values, Value Range
Binding goals and services to activities	Goal binding, service binding	Activity name, functional classification, preconditions, postconditions, input and output parameters, selection criteria, contextual information
Binding services to goals (according to (Pedrinaci et al., 2010))	Service binding	Goal in terms of semantic annotations: functional classification, preconditions, postconditions, input and output parameter, selection criteria, contextual information
Resolving activities to process patterns and templates	Activities as process fragment	Activity name, functional classification, preconditions, postconditions, input and output parameter, contextual information
Match services on syntactic and semantic layer	Service matching	Service descriptions: preconditions, postconditions, input and output parameters
Dynamically create data flow connectors	Connector type, connector instantiation, transformation script,	Service descriptions: Input and output parameters

	mapping file	

In this thesis, the focus is on the design principles of LPM and the according LPMS. Thus, a detailed specification of the composition component is not provided. The work performed in the SOA4All project is referred for a detailed description of the composition component (González-Cabero, Lecue, & Villa, 2008; Gorronogoitia, Radzimski, Lecue, Villa, & Di Matteo, 2010).

5.2.4 Execution Engine

Main features

This thesis describes only those aspects of the execution engine that are specific to LPM. Although these features are not new as such, the new thing of the LPMS is the support through semantic annotations. A detailed specification of the execution engine can be found in the documentation of the SOA4All project in Ripa et al. (Ripa et al., 2010; Ripa, Zuccala, & Mos, 2009).

- *Handling of semantic information for process execution*: Basically, an execution engine for the LPMS requires the representation of the control logic, such as an LPML model, and means for service selection and adaptation at runtime. As aforementioned, the control logic might be represented through LPML or through other executable process representations, such as BPEL. The information required for service selection and adaptation at runtime is given in terms of semantic annotations. Hence, the LPMS execution engine comprises a component for syntactic process execution and a component for reading and applying the semantic information for service selection and adaptation.
- *Runtime adaptation and dynamic binding*. Traditional process execution systems suppose that service interfaces are known at design time. However, in order to support dynamic selection and binding and due to a lack of standardization of service-oriented systems, systems should be able to handle services the interfaces of which are only

known at runtime (see (Cavallaro & Nitto, 2008) and the European Integrated Project SeCSE[21]). Hence, services have to be composed dynamically and heterogeneous interfaces have to be mapped at runtime, e.g. interfaces of WSDL and RESTful services. Semantic annotations of the service interfaces support this dynamic mapping. These semantic annotations might be used to automatically create mappings and according transformation scripts during execution of the process at runtime. In order to create the scripts, a complete understanding of the semantics of the involved service parameters and operations is needed. Thus, the semantic annotations have to use a common knowledge base. Furthermore, a lifting or lowering mapping schema might be needed. This applies to cases where syntactic descriptions have to be lifted to semantic information in order to create a mapping on the semantic level. As well, a lowering schema might be required in order to perform a mapping on the syntactic layer based on semantic information.

- *Dynamic replacement of services*: To support self-healing functionality, the LPML has introduced the element *ReplacementCondition*. Besides the dynamic selection and mapping of services, the LPMS supports as well the dynamic replacement of services. The *ReplacementCondition* element indicates the conditions and preferences to replace a service.
- *Handle user preferences and context*: The LPMS considers context information and supports the user in expressing preferences for the selection of appropriate services. While the semantic annotation *ContextualInformation* states the context information, the *SelectionCriteria* element indicates these preferences stated as non-functional properties. Hence, the execution engine considers the expressions of *ContextualInformation* and *SelectionCriteria* to dynamically select and bind services.

Current technologies for process execution

As aforementioned, the execution engine has to handle various kinds of services and according descriptions and compose them seamlessly. Currently, REST and

[21] See www.secse-project.eu

WSDL/SOAP services are wide-spread standards for services. The interfaces of REST services are described by hRESTS, the WSDL/SOAP services through WSDL. In order to integrate semantic annotations in hRESTS descriptions, MicroWSMO has been defined (Kopecky et al., 2009). MicroWSMO defines annotations for hRESTS as SAWSDL does for WSDL.

A detailed description of a prototypical execution engine for the LPMS can be found in the documentation of the project SOA4All (Ripa et al., 2010; Ripa et al., 2009).

5.3 Conclusion

In this section, the LPM design process and the prototypical tools supporting LPM have been presented. These artefacts form the LPMS as an implementation architecture for LPM.

- Design Process in section 5.1: This design process starts with the process modelling performed by the business user. The further steps comprise the generation of semantic annotations, the mapping of goals to activities, the service discovery, the service selection, and the service composition. The result of this design process is the executable process.
- API in section 5.2.1: The API manages the exchange of LPML code. This API abstracts and hides the complexities of the LPML elements and their concrete serialization formats to the programmer. In the context of the LPMS, the API supports serialization into an extended form of BPEL but may also be used by 3rd party tools.
- Process Editor in section 5.2.2: The Process Editor is the user interface allowing for the creation, manipulation, and execution of LPML models. It is implemented according to the principles of RIAs. In order to implement the abstraction principle of the LPMS, a drawing area and a data area are provided. The drawing area is dedicated to the abstract, graphical LPML layer and implements modelling support functionality, such as hiding gateways. The process editor supports gathering, applying, and visualizing contextual information. Further, favourite lists and wizards are provided supporting the user in modelling and providing information step by step. Depending on the user expertise, different modelling support modes might be selected. The data flow modelling is facilitated through support on the specification of data flow connections and according connectors. Finally, a pattern and template recommendation functionality is implemented.
- Composition Component in section 5.2.3: The composition component enables the flexible and dynamic creation, instantiation, and adaptation of processes at design

time. In the context of the LPMS, this means binding goals and services to activities, binding services to goals, resolving activities to process patterns and templates, checking and supporting the matching of activities and services on semantic and syntactic layer, and creating data flow connectors.

- Execution Engine in section 5.2.4: To execute the LPML processes a special execution engine is needed. The LPMS leaves open whether to create an engine executing directly the LPML models or to transform the LPML models into a standard executable process language, such as BPEL, and to be executed on an existing engine. The new functionalities for usage in the context of the LPMS comprise the handling of execution-relevant, semantic information, the runtime adaptation and dynamic binding, the dynamic replacement of services, and the handling of user preferences and context.

In the context of the design science, this section 1 describes the second part of the design of an artefact, the development. The iterative design approach proposed in section 4.8 might be further refined according to the proposed artefacts of this section. Again, the starting point should be to implement the handling of semantic annotations for goals and the activity instantiation. Afterwards, the support for data flow should be implemented. The semantic annotations for context information and the handling of patterns and templates might be implemented with lower priority.

This section finalised the design of the artefact, the LPMS. The next section covers the application of the LPMS in a use case. Hence, it is presented, how the LPMS might add value to the public sector as a potential target industry.

6 LPM IN THE PUBLIC SECTOR

After the design and development phases, the design science defines the demonstration of the use of the artefact to solve a problem (Peffers et al., 2008). In this thesis, the demonstration phase involves a use case in the public sector. A process in the context of the EU Services Directive has been selected in order to reveal the application of the LPMS.

6.1 INTRODUCTION

Typical public administrations interact with citizens, businesses, and other administrations in a plethora of administrative procedures (PICTURE, 2007). Existing IT infrastructures however, often don't support these procedures in an efficient and effective way. Island solutions exist and services are still executed manually.

To adjust to changes in mandates and laws is a key success factor for administrations while budgets get smaller. Modern IT technologies and approaches, such as SOA, SWS, Web 2.0, and BPM as well as end-user empowerment, promise to support administrations to cope with these challenges. In this thesis, sample scenarios and specifics of the public sector are picked up. Thus, the thesis reveals, how the LPMS based on these technologies might be used in the public sector.

6.1.1 The Public Sector as a Target Market for LPM

Any functioning society is dependent on services provided by public authorities. Services in the area of public security, healthcare, education, or defence are provided by the governmental organizations. Hence, a large number of customers and significant IT budgets make the public sector very attractive for new software solutions. However, like in the private sector, in the public sector as well, the need for effective and efficient IT support increases. Costs have to be reduced.

The public sector comprises a huge amount of users that have to fulfil administrative tasks in an efficient way. In Germany, for instance, the public sector is divided into three main decision areas, namely the state, regions (federal states), and communities (cities, towns, and municipalities). In the community area, about 25.000 public administrations exist in 12.000 communal authorities (Habbel, 2008). The according workforce of these administrations counts a number of 4.5 million people (Destatis, 2010). The customers of the public sector in Germany are 82 million citizens and 3.5 million enterprises (Habbel, 2008).

The public sector is characterized

- By a huge amount of users
- By the wide-spread existence of island solution
- By a majority of civil servants being typical business users

6.1.2 Current Situation in Typical Public Sector Organisations

Traditional applications in the public sector are based on heterogeneous technology stacks, are often custom-built, and keep core processes or master data redundantly (Vogel et al., 2009). Any information exchange between applications is often done manually, has to overcome media breaks, or requires high effort to hard-wire any connections. Due to the inflexible IT infrastructure and applications, the quality of services provided to citizens is often low. Furthermore, services are often produced at high costs. Difficulties in internally developing new or adjusting existing IT-based services leads often to high costs and limited flexibility, especially in case of requesting service development from external companies or consultants. As well, small public administrations often do not have resources and know-how to develop services. Moreover, projects exceeding a budget threshold have to apply a bidding mechanism, again limiting the flexibility of software development and changes.

Improving business processes is ranked as the primary business objective for governments in 2010 according to a Gartner CIO survey (Meehan, 2010) and a Gartner Report (Bittinger & Di Maio, 2010; Meehan, 2010). Further, Bittinger and Di Maio claim a shift to more collaborative work for process engineering and a shift from function-centric to process-centric thinking.

Although the adoption of SOA in governmental organisations is commonly recognized for a couple of years (Leganza, 2006) and seems to be on a good level, the progress is slow. Common issues concern low reuse, shortcomings in enterprise architecture, business processes, and SOA governance (Bittinger & Di Maio, 2010).

The application of the LPMS might significantly improve the adoption of BPM and SOA and hence make governmental organisations more flexible and reduce IT costs. Furthermore, through the empowerment of constituents, costly software sourcing from external providers or consultants might be reduced.

The majority of civil servants have limited IT skills and no interest in becoming an IT expert making the public sector a typical application area for the LPMS.

6.1.3 Potential Users in the Public Sector

In order to further motivate the application of the LPMS to the public sector, the potential users are investigated in more detail. A study about a user-friendly tool for process modelling has been conducted in the PICTURE project (PICTURE, 2007). The respondents had been business users with limited IT knowledge. One of the core results has been that 80% of the respondents evaluate the simplicity of a modelling language as "important" or "very important" on a six point scale ranging from "absolutely unimportant" to "very important". Further findings of the study had been the importance of an intuitive user interface of a modelling tool and the empowerment of all users to model processes.

6.1.4 Need for Action

As aforementioned, the public sector use case serves as a demonstrator according to the design science research methodology. The applicability of the LPMS in a real setting is revealed by this use case. Therefore, user roles for the public sector are defined. A concrete public sector process has been selected to reveal the application of the LPMS. Further, specific extensions required in the public sector and their implementation by the LPMS are presented.

6.2 THE LPMS IN THE PUBLIC SECTOR

In this section, a typical scenario including a sample process and according services of the public sector is shown. The scenario is based on the EU Services Directive. In the following, the scenario is introduced, the user roles are described, the typical context information for the public sector is provided, and a typical process including services and annotations are presented.

6.2.1 The EU Services Directive as public sector scenario

The EU Services Directive is an important initiative aiming at facilitating cross-border settlement for service companies (Commission, 2006). For all member states of the European Union the administrative procedures shall be harmonized and supported by appropriate IT. According to the directive, *"service providers should be able to complete electronically and at a distance all procedures and formalities necessary to provide a given service"* (Commission, 2007). The implementation of these objectives is performed through a so called single point of contact that handles all procedures to fulfil the constituent's needs. Hence, all kind of document and information exchange should be handled electronically.

The LPMS might support the specification and implementation of new business tasks and processes. Due to the implementation of the service directive in multiple countries and in multiple scenarios with various variants and deviations, a flexible IT solution is required, such as the LPMS. Furthermore, the administrations don't have the time and budget to source the new IT solutions from expensive external software providers. A solution is required empowering the employees of the administrations that are mainly business users to implement the tasks and processes themselves.

6.2.2 User Roles

The main target users of this public sector use case are civil servants with business knowledge but without any programming skills. According to their job role, they are not interested in gaining IT knowledge. The goal of the LPMS is to empower constituents (citizens and businesses) to issue and monitor requests for public services in a web portal. Further, civil servants should be enabled to search for, compose, annotate, execute, monitor, and adjust public services and processes in order to reply accordingly to the constituents' requests.

Table 28: Roles in the public sector

Role	Description
Process Modeller	Produce processes
Process Expert	Edit and refine existing processes
Process User	Use and hence, execute processes
Legal Expert	Check legal compliance
Reviewer	Review processes, e.g. for compliance or security reasons
Approver	Approve processes as a whole and activities

Table 28 lists the different roles in the public sector the users might take when interacting with the LPMS. According to the target user definition in section 1.3, these users have IT knowledge to an extent that allows for dealing with standard office, desktop, and web applications. However, they lack specific programming or process composition skills. An employee might take several roles according to his job tasks and responsibilities.

6.2.3 Context Information

The context information that is important in the public sector might be differentiated into profile and organisational context as follows:

- Profile context: User-ID, Name, Country, Region, City, Language, and preferred method of payment
- Organizational context: Organization, organizational unit / department, user role / position / rights, and user skills / competencies

6.2.4 Process Modelling

From the EU Services Directive scenario, the process “Registration of a business” has been selected as a typical process in order to instantiate the LPMS. This process is based on the process implemented by the city of Duesseldorf in Germany (Duesseldorf, 2009). However, such a process might be found in multiple public administrations. It describes the situation in which a service provider plans to open a business in another country of the EU that he is not citizen in. Hence, he registers a new business in the selected country.

Figure 30 shows this sample process in a BPMN notation. According to Smith (Smith, 2010), scenarios for end-user programming have typical characteristics, such as being a human-centric application, having minimal system integration, having limited organizational scope, being geographically localized to minimize network security requirements, and not handling any critical data. The selection of the process “Registration of a business” has considered these characteristics.

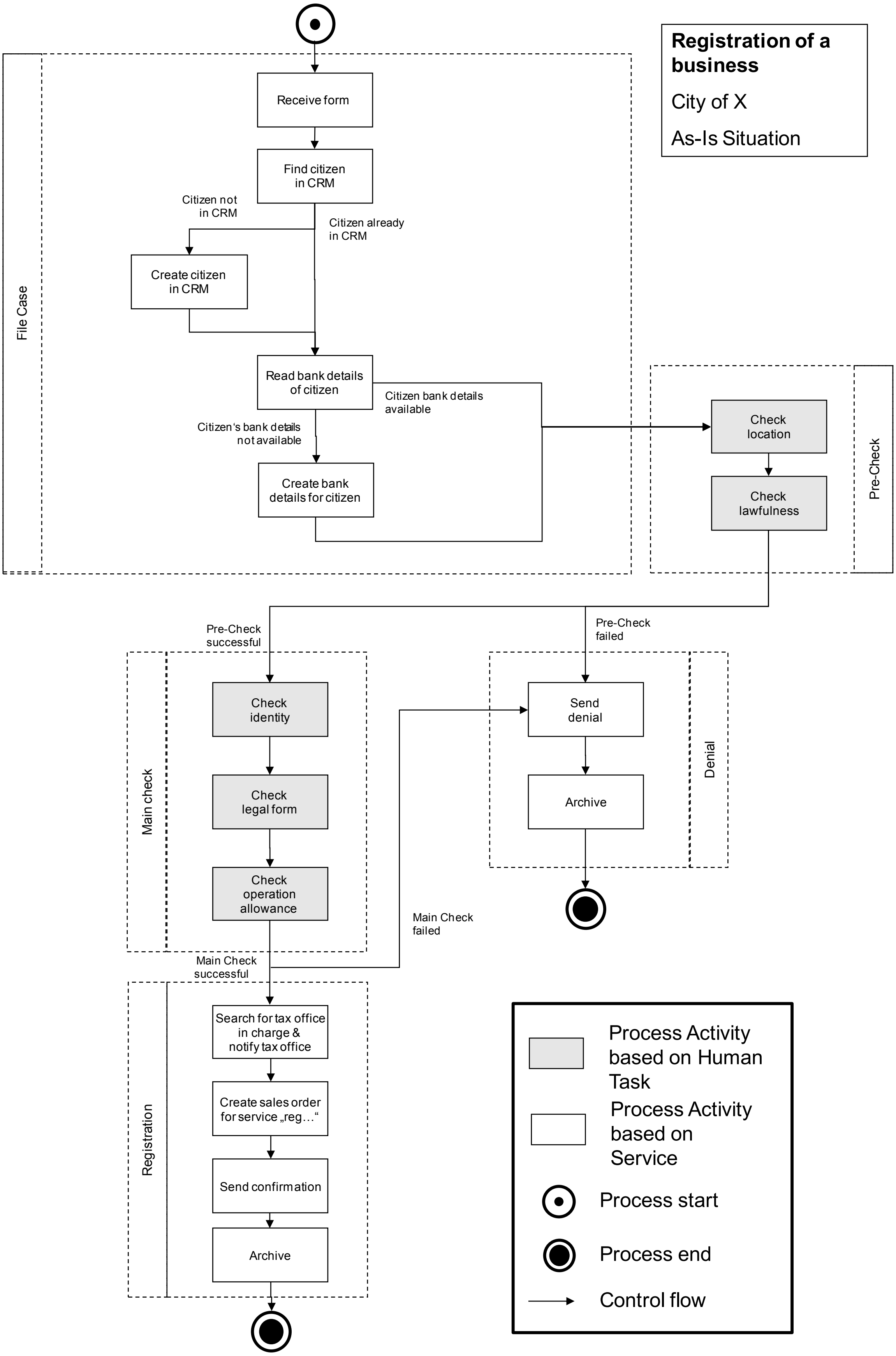


Figure 30: Sample process "register a new business"

Figure 31 depicts a sample part of the process "Registration of a business" in the graphical abstraction notation of the LPML. Parts of the canonical LPML model of the process "Registration of a business" are presented in the following. These parts are based on the work performed in work package 7 of the SOA4All project. I have been as well the author of the process model described in the following. The complete process model can be found in Java notation in section 9.1.1. Table 29 depicts a sample activity as described in section 4.2.2.1 in Java notation. The activity "Find citizen in CRM" from the process "Registration of a business" has been selected. This activity is performed by the administration officer after receiving a request to register a new business. The purpose of the activity is to check whether the requesting citizen is already registered in the administration's CRM system.

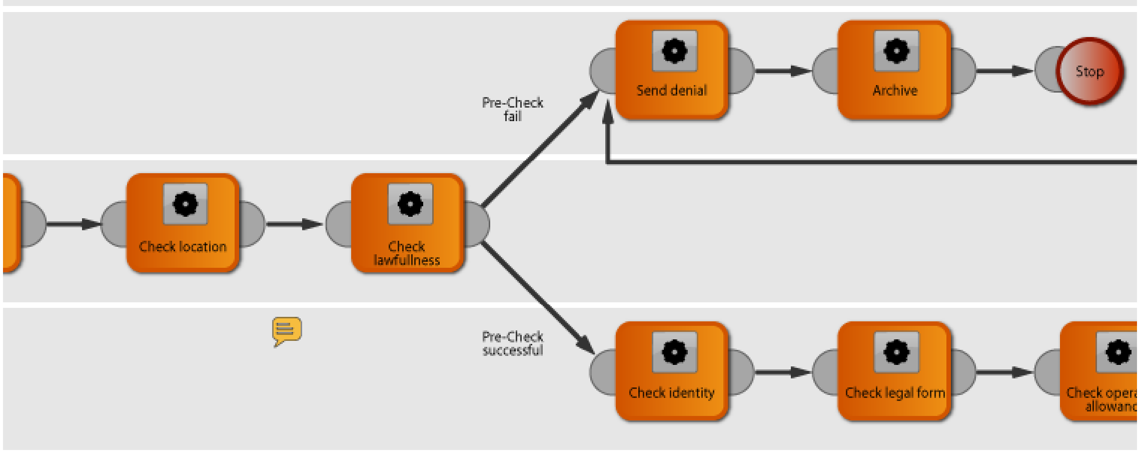


Figure 31 : Sample part of the process "Registration of a business"

The sample file shows the instantiation of the activity, the conversation element preparing the binding of a goal and a service, the goal element, and the service element. The activity attributes are set as well as the input and output parameter. An important characteristic of the LPMS is the section about semantic annotations. Besides the URI reference for any semantic annotation, the type is specified. The type specification is performed semi-automatically. The user is supported in differentiating functional and non-functional properties. For example, for the public sector, the process editor might implement a wizard that provides a predefined set of functional classes and non-functional requirements and constraints. The user might then easily select a functional class and adjust it, if required. Furthermore, the non-functional properties might easily be selected and specified. For example, the wizard for non-functional properties provides the property "price". In addition, the user might specify a threshold price and whether he intends to have smaller or bigger prices. Like for the functional classification, another wizard supports the user in structuring preconditions and postconditions.

Table 29: Activity "Find citizen in CRM"

```
Activity findCitizenInCRM = new ActivityImpl();

Conversation findCitizenInCRMConversation = new
ConversationImpl();
Goal findCitizenInCRMGoal = new GoalImpl();
Service findCitizenInCRMService = new ServiceImpl();

findCitizenInCRM.setName("Find Citizen in CRM");
findCitizenInCRM.setOperation();
findCitizenInCRM.setStartElement(false);
findCitizenInCRM.setEndElement(false);
findCitizenInCRM.setSynchronous(true);
findCitizenInCRM.setConversation(findCitizenInCRMConversat
ion);
findCitizenInCRM.setHumanTask(false);

process.addProcessElement(findCitizenInCRM);
```

Table 30 shows an example of the process gateway "Citizen in CRM exclusive fork gateway" and Table 31 visualizes the sample flow element "Find citizen in CRM flow". Both examples are as well taken out of the process "Registration of a business" and depicted in Java notation. The gateway describes a process split, hence the *split* attribute is set to "true". Furthermore, the gateway references the semantic annotation for its condition. The condition is a logical expression and references a SPARQL query that is evaluated at runtime.

Table 30: Citizen in CRM exclusive fork gateway

```
ExclusiveGateway citizenInCRMforkGateway = new
ExclusiveGatewayImpl();

citizenInCRMforkGateway.setSplit(true);
citizenInCRMforkGateway.setCondition(citizenInCRMforkCondi
tionAnnotation);

process.addProcessElement(citizenInCRMforkGateway);
```

Table 31 describes three flows. The first example describes a simple flow without any condition. The second example covers a complex sample flow with conditions. The gateway in the example has two outgoing flows. Each of the outgoing flow references a condition. These conditions have to be evaluated in the context of the gateway condition in order to decide which flow to follow.

Table 31: Find citizen in CRM flow

```
//Example for a simple sample flow
process.addProcessElement(new FlowImpl(findCitizenInCRM,
citizenInCRMforkGateway));

//Example for a complex sample flow

Flow citizenInCRMforkOutgoingFlow1 = new
FlowImpl(citizenInCRMforkGateway, citizenInCRMMerge);
Flow citizenInCRMforkOutgoingFlow2 = new
FlowImpl(citizenInCRMforkGateway, createCitizenInCRM);

citizenInCRMforkOutgoingFlow1.setCondition(citizenInCRMfor
kOutgoingFlow1Condition);
citizenInCRMforkOutgoingFlow2.setCondition(citizenInCRMfor
kOutgoingFlow2Condition);

process.addProcessElement(citizenInCRMforkOutgoingFlow1);
process.addProcessElement(citizenInCRMforkOutgoingFlow2);
```

6.2.5 Services and Annotations

A prerequisite for the service composition in the context of the LPMS is that potential services are semantically described. Further, these annotations have to be based on a common ontology or on ontologies that are related through a mapping specification. Various ontologies have to be considered for the sample process "Registration of a business", such as a service ontology and a public sector ontology. These ontologies might be further structured into a hierarchy of ontologies. The ontology specifications and relations are described in detail in Vogel et al. (Vogel et al., 2010). An extract of one of those ontologies is presented in the Annex in section 9.1.2. As aforementioned, a complete specification of the ontologies related to the public sector can be found in Vogel et al. (Vogel et al., 2010).

As an example of how an activity is annotated to find an appropriate service, the activity "Find citizen in CRM" is selected. Besides the name for the activity, the user has to specify requirements and constraints for the activity. These requirements and constraints build the base for the ontological annotations. For the activity "Find citizen in CRM" the following requirements and constraints could be typically provided:

- Requirement: Search for citizen (R1)

- Requirement: Search by Name/First name, Address (R2)
- Requirement: Search by ID (R3)
- Requirement: Name and first name or address exists (R4)
- Requirement: Return ID that is assigned to Name/First name (R5)
- Constraint: Search only in CRM system (C1)
- Constraint: Search only in public sector entries (C2)
- Constraint: No service fees (C3)
- Constraint: Only trusted services might be selected (C4)
- Constraint: Prefer service searching by name or address if existent (C5)

In the next step, these requirements and constraints have to be transformed into semantic annotations for functional classification, non-functional properties, preconditions, postconditions, and metadata. Ideally, this step is performed completely automatically. In case the annotations cannot be generated automatically, the user is supported in categorizing the requirements and constraints through a wizard. Table 32 reveals how the semantic annotations are derived from the requirements and constraints given by the user.

Table 32: Deriving semantic annotations from requirements and constraints

Semantic annotation classification	Requirement/Constraint
Functional classification	R1, R2, R3, C1, C2
Non-functional properties	C3, C4
Preconditions	R4
Postconditions	R5
Selection Criteria	R3, C5
Replacement Condition	R3
Metadata	None

Table 33 shows the description of the precondition of the activity "Find citizen in CRM" in N3[22] notation. This description is based on the work performed in Vogel et al. (Vogel et al., 2010). The expressions are generated by the process editor, based on WSMO-Lite and the WSML namespace, and reference an ontology named *LPMOntology*. The precondition requests the existence of a name and a first name, an address, or an ID. A couple of further existing ontologies had been presented in section 3.4.2.

For the postconditions the declaration code looks similar to the code of the preconditions.

Table 33: Description of the activity precondition in RDF based on WSMO-Lite

```
tns:ActivityFindCitizenInCRMPrecondition a wsl:Condition;

rdf:value """
  (?Name[FindCitizenInCRM:FirstLineName wsml:hasValue
  ?firstLineName] wsml:memberOf LPMOntology:Name and
  ?Name[LPMOntology:SecondLineName wsml:hasValue
  ?secondLineName] wsml:memberOf LPMOntology:Name)
  or
  (?PhysicalAddress[LPMOntology:CountryCode wsml:hasValue
  ?countryCode] wsml:memberOf LPMOntology:PhysicalAddress
  or
  ?PhysicalAddress[LPMOntology:RegionCode wsml:hasValue
  ?regionCode] wsml:memberOf LPMOntology:PhysicalAddress
  or
  ?PhysicalAddress[LPMOntology:StreetPostalCode
  wsml:hasValue ?streetPostalCode] wsml:memberOf
  LPMOntology:PhysicalAddress or
  ?PhysicalAddress[LPMOntology:CityName wsml:hasValue
  ?cityName] wsml:memberOf LPMOntology:PhysicalAddress or
  ?PhysicalAddress[LPMOntology:POBoxID wsml:hasValue
  ?pOBoxID] wsml:memberOf LPMOntology:PhysicalAddress or
  ?PhysicalAddress[LPMOntology:StreetName wsml:hasValue
```

[22] N3 is a RDF syntax. For details see http://www.w3.org/DesignIssues/Notation3.html

```
  ?streetName] wsml:memberOf LPMOntology:PhysicalAddress
  or
  ?PhysicalAddress[LPMOntology:HouseID wsml:hasValue
  ?houseID] wsml:memberOf LPMOntology:PhysicalAddress)
  or
  (?ID [FindCitizenInCRM:PersonID wsml:hasValue
  ?PersonID] wsml:memberOf LPMOntology:ID)
wsml:AxiomLiteral
```

The functional classification might be described ontologically – as well in N3 notation - as depicted in Table 34. Again, this example is based on the work described in Vogel et al. (Vogel et al., 2010). Here, the namespace “lpmfc” stands for “lightweight process modelling functional classification”. The functional classifications are categorized into three high-level classifications, the operational, designation, and extensible classifications. The latter is a general dimensional construct allowing for extending the set of dimensions by domain-specific dimensions. For each of the high-level categories further dimensions might be defined. In the sample file in Table 34, the dimension “search” is defined for the operational classification, the dimensions “CRM” and “Public Sector” for the designation classification, and “NameSearch”, “AddressSearch”, and “IDSearch” are defined for the extensible classification. Furthermore, a detailed description of the relevant ontological dimensions can be found in the Annex in section 9.1.2.

Table 34: Functional classification of the sample activity "Find citizen in CRM"

```
lpmfc:hasBusinessDomainOperationalClassification
lpmfc:BusinessDomainFunctionalClassificationSearch;

lpmfc:hasBusinessDomainDesignationClassification
lpmfc:BusinessDomainFunctionalClassificationCRM;

lpmfc:hasBusinessDomainDesignationClassification
lpmfc:BusinessDomainFunctionalClassificationPublicSector;

lpmfc:hasBusinessDomainExtensibleFunctionalClassification
lpmfc:BusinessDomainServiceGoalAssistBusinessRegistrationN
ameSearch

lpmfc:hasBusinessDomainExtensibleFunctionalClassification
lpmfc:BusinessDomainServiceGoalAssistBusinessRegistrationA
ddressSearch

lpmfc:hasBusinessDomainExtensibleFunctionalClassification
lpmfc:BusinessDomainServiceGoalAssistBusinessRegistrationI
```

```
DSearch;
```

In order to precise the service search, a goal is instantiated in an intermediate step. The goal concentrates on the basic functionality and the postcondition of an activity. In the example above in Table 34, the functional classifications “search”, “CRM”, and “public sector” are used to search for a goal. The extension classifications “NameSearch”, “AddressSearch”, and “IDSearch” are refinements of the classification and used at a later stage in order to bind appropriate services to the goal.

For the further annotations, such as non-functional properties and metadata, the annotation mechanism is similar.

The ontological activity description might now be used to formulate a query in order to discover a goal. In the current example, the goal is to find a citizen in the CRM system. The according SPARQL query that is fed into a discovery engine is described in Table 35, again based on the work described in (Vogel et al., 2010).

Table 35: Simple query for goal discovery based on functional classification

```
SELECT ?goal ?goalOntologyConceptRefURI
?goalDeploymentURI
WHERE {
     ?goal rdf:type lpm:Goal ;
     lpm:hasFunctionalClassification
?functionalClassification .

     ?goal rdf:type lpmgoal:Goal;
     rdfs:isDefinedBy ?goalDeploymentURI .

     ?functionalClassification rdf:type
lpmfc:BusinessDomainFunctionalClassification ;

     lpmfc:hasBusinessDomainOperationalClassification
lpmfci:BusinessDomainFunctionalClassificationSearch ;

     lpmfc:hasBusinessDomainDesignationClassification
lpmfci:BusinessDomainFunctionalClassificationCRM ;

     lpmfc:hasBusinessDomainDesignationClassification
lpmfci:BusinessDomainFunctionalClassificationPublicSector
;
}
```

The discovery engine returns the goal "FindCitizenInCRMGoal". The goals have a more detailed, predefined structure than the activities. Hence, service discovery queries might be derived through a standard generation script. Particularly, the input and output parameter specification further supports the service discovery. Goals serve as service classes and a kind of filter in order to select a set of services with the required functional classification. The set of services attached to a goal is further investigated in order to figure out the best fitting service based on preconditions, postconditions, input and output parameters, and non-functional properties.

Based on the goal description, a more refined service search query might be formulated. The refined SPARQL search query is listed in Table 36, again based on (Vogel et al., 2010). As aforementioned, the search query is extended by functional classifications for the search by name and first name, address, and ID. While the functional classifications for the goal discovery are mandatory functionalities, the extension functionalities for discovering services are optional. Hence, for example, services might be found that fulfil the name search classification but not the ID search classification.

Table 36: Refined query for service discovery based on goal description

```
SELECT ?service ?operation ?serviceOntologyConceptRefURI
?serviceDeploymentURI
WHERE {
     ?service rdf:type msm:Service ;
     msm:hasOperation ?operation ;
     sawsdl:modelReference ?serviceOntologyConceptRefURI ;
     msm:hasFunctionalClassification
?functionalClassification .

     ?service rdf:type wsl:Service ;
     rdfs:isDefinedBy ?serviceDeploymentURI .

     ?functionalClassification rdf:type
lpmfc:BusinessDomainFunctionalClassification ;

     lpmfc:hasBusinessDomainOperationalClassification
lpmfc:BusinessDomainFunctionalClassificationSearch ;

     lpmfc:hasBusinessDomainDesignationClassification
lpmfc:BusinessDomainFunctionalClassificationCRM ;

     lpmfc:hasBusinessDomainDesignationClassification
lpmfc:BusinessDomainFunctionalClassificationPublicSector ;
// This is the new part refining the search query by the
type of search

lpmfc:hasBusinessDomainExtensibleFunctionalClassification
lpmfc:
BusinessDomainServiceGoalAssistBusinessRegistrationNameSea
rch ;

lpmfc:hasBusinessDomainExtensibleFunctionalClassification
lpmfc:
BusinessDomainServiceGoalAssistBusinessRegistrationAddress
Search ;

lpmfc:hasBusinessDomainExtensibleFunctionalClassification
lpmfc:
BusinessDomainServiceGoalAssistBusinessRegistrationIDSearc
h .

// Here ends the refining functional classification

     ?operation msm:hasInputMessage ?input ;
     msm:hasOutputMessage ?output .
}
```

In the example "FindCitizenInCRM", the service search returns two services, one service searching for the citizen by name or address and the other service searching by ID. As described in the selection criteria (see Constraint C5 above), the service searching for the citizen by name or address is preferred.

As aforementioned, a prerequisite of the functioning of the LPMS is the existence of semantically described web services. For WSDL services, the file containing the semantic annotations has to be referenced through the SAWSDL annotation *modelReference*. For RESTful services the Micro-WSMO annotation *modelReference* references the location of the file containing the semantic annotations. The two potential services to find a citizen in a CRM system –

FindCitizenInCRMByNameOrAddress and FindCitizenInCRMByID – are described by a WSDL interface including SAWSDL references and can be found in the documentation of the SOA4All project[23].

6.3 Required Functionalities to Processes and Services

In this section, required functionalities for the public sector are described that are not yet covered by the general requirements described in section 3.2. Table 37 presents those requirements and derives specific requirements for LPM and the LPMS. The requirements are based on public sector scenarios, such as registering a new business as described in Vogel et al. (Vogel et al., 2009).

Table 37: Public sector requirements to the LPM language and tools

Requirement Category	Public Sector Requirement	LPM Requirement
User Management	Identification and Authentication: Manage user access based on profile and account data. Hence, a username and a password are required.	Provide a reference to a role model where user data is stored. The role model is implemented according to a standard identity and access management specification.
	Profile Management: Associate essential information to each user profile, such as ID, name, email, and role	
	Authorization: Manage access rights to resources according to the user profile	
	Auditing: Provide a tracking of user actions for accounting	

[23] Taken out of the SOA4All project, see www.soa4all.eu. The services are SAP Enterprise Services, see www.sap.com

	reasons	
	Preference Management: Provide simple storage and retrieval of user-based and/or application-based settings. These settings might be used to customize the tools or to fill in context-information into process models.	
Approval	Provide a special functionality to approve modelled processes concerning correctness, legal aspects, compliance etc.	Provide information about process approval. Process states should include ready for approval, approved, and rejected.
Versioning	Allow for a special tracking of versions and the according editors.	Provide information about process model version and the editors
Human task management	Allow for the marking of activities as human tasks. Further, notify the according user about a required action	Provide information whether an activity is a human task. Further allow for notifying a user that has to perform the task.

A proper user management is an important requirement to the public sector services and processes. The user management is normally handled by Identity and Access Management (IAM) solutions. A standard IAM solution is referred to in order to implement the user management. However, the LPML has to be extended by a process attribute referencing the user model.

6.4 LPML AND TOOL CUSTOMIZING

This section describes how the LPM language and tools implement the required functionalities. Table 38 lists the requirements as defined in the previous section and describes the according LPMS.

Table 38: LPM language and tools implementing the public sector requirements

Requirement Category	**LPM Requirement**	**Implementation through the LPMS**
User Management	Provide a reference to a role model where user data is stored. The role model is implemented according to a standard IAM specification.	Attach semantic annotation of type *metadata* to the process element referencing the role model URI

Approval	Provide information about the process approval state. State values should include ready for approval, approved, rejected.	Attach a semantic annotation of type *metadata* for the approval state to the process element. The values representing the states are ready for approval, approved, and rejected
Versioning	Provide information about the process model version and the editors	Add a semantic annotation of type *metadata* for version and editor
Human task management	Provide information whether an activity is a human task.	Attach an attribute "HumanTask" of type *boolean* to activities.
	Provide information about the user calling the human task server for a callback mechanism	Attach an attribute of type *ID* to activities indicating the calling user
	Allow for notifying a user that has to perform the task.	Attach an attribute "NotifiedUser" of type *ID* to activities

6.5 Business Aspects

In this section, an exploitation option for the LPMS is presented. The LPMS might be implemented as process delivery platform where users might provide and consume services in terms of processes. The process delivery platform might be implemented within an organization or across organizations. For the public sector, both scenarios might be implemented.

Besides the LPM tools described in section 5.2, the process delivery platform comprises a component for discovering services and processes, a repository, and an execution infrastructure for scalable web service calls.

In this thesis, the potential roles and their according rights and duties in the context of a process delivery platform are described in detail.

Table 39 presents these roles and the according description according to the description in Vogel et al. (Vogel et al., 2009). Each of the roles has a different motivation to use the LPMS serving as a process delivery platform.

Table 39: Roles Involved in the process delivery platform

Role	Description
Service developer	Designs and implements new services to be consumed through processes.
Service and process provider	Offers and supplies services and processes to consumers. The services might be consumed by users or processes, the processes by users or other processes. Both services and processes are provided on the process delivery platform.
Service and process broker	The service broker bundles existing services and processes to new services and processes that are provided through the process delivery platform.
Consultant	Consultants support service providers and consumers performing their tasks. For example, service providers are supported in describing or categorizing their services or defining price models. Service consumers might be supported in composing services or processes according to their business needs.
Service and process composer	The service and process composer orchestrates existing services and processes to new processes. In contrast to the mere bundling of services and processes that are situated on a business level, the service and process composer acts on both the business and the technical level to develop new services and processes. The composer either provides the new compositions himself or sells them to service and process providers.
Service and process annotator	The service and process annotator attaches semantic annotations in terms of metadata to services or processes. The metadata might describe technical and business information or information about the roles and actors having a stake in the service or process. For example, the metadata might comprise information about the process composer, the services involved, the versioning of the process, or the editing history.
Service and process consumer	The service and process consumer demands and uses services and processes. He searches for, selects, combines, and executes services and processes.
Platform vendor	The platform vendor designs and implements the components and tools for the service and process delivery platform. The vendor either performs the role of the platform provider himself or sells the platform.

Platform provider	The platform provider hosts the service and process delivery platform.

6.6 CONCLUSION

This section addressed the application of the LPMS in a use case, namely the application in the public sector. Typical public administrations interact with a high amount of citizens, businesses, and other administrations in a plethora of administrative procedures. Existing IT infrastructure however, doesn't support these procedures in an efficient and effective way since island solutions exist and services are still executed manually.

Due to the inflexible IT infrastructure and applications, service costs are high and the quality of services provided to citizens is often low. Difficulties in internally developing or adjusting IT-based services arise. As well, small public administrations often do not have resources and know-how to develop services.

The LPMS promises to contribute to the effective and efficient management of these challenges. To customize the LPMS to the public sector, user roles have been defined. These roles are differentiated according to their task duties and IT-skills and comprise process modellers, process experts, process users, legal experts, reviewers, and approvers. Further, context information has been defined covering aspects specific to the public sector, such as the profile and organizational context.

A concrete public sector process for the registration of a new business in the context of the EU Services Directive has been selected. Thanks to this sample process, this thesis revealed, how activities, goals, services, gateways, semantic annotations, and flows are represented in the LPML.

Further, specific extensions required in the public sector and their implementations through the LPML and the tools have been presented. These extensions address the user, approval, versioning, and human task management through the LPMS.

Finally, a process delivery platform as an exploitation option of the LPMS has been presented.

The application to the public sector use case revealed that the LPMS adds value to a business scenario. Further, the use case revealed that the LPMS might be extended according to new, specific requirements. Hence, the application is not limited to specific scenarios.

According to the design science phases, this section covers the demonstration of the use of the artefact to solve a problem (Peffers et al., 2008).

The following section covers the evaluation of the LPMS based on well-defined metrics. The evaluation again is a well-defined phase in the design science process according to Peffers et al.

7 EVALUATION

The design science research methodology comprises the two major processes *build* and *evaluate* (March & Smith, 1995). The evaluation is an observation and measurement "*how well an artefact supports a solution to the problem*" (Peffers et al., 2008). After describing the creation and use case demonstration of the artefact, in this section, the evaluation of the design of the LPMS comprising the LPML and tools follows. This proceeding is in line with the design science guideline "*design evaluation*" as introduced in Hevner et al. (Hevner et al., 2004). The evaluation is based on the requirements that are described in section 2 and the observed results of using the LPMS in the use case described in section 1. As aforementioned, this thesis, and hence the evaluation, focuses on the design of the LPMS. In addition, the evaluation section describes, how the evaluation of the use of the LPMS should be performed.

Basically, two categories of evaluation approaches exist: Analytical and empirical evaluation (Fettke & Loos, 2003). While analytical approaches focus on logical reasoning in terms of a descriptive evaluation, the empirical evaluation covers objective observations, such as case studies, surveys, or experiments. Hevner et al. (Hevner et al., 2004) further structure these two approaches into evaluation methods. Table 40 reveals how these evaluation methods are applied to the LPM and LPMS evaluation. The last column of Table 40 references the sections where the evaluation method is applied.

Table 40: Evaluation methods in this thesis

Evaluation Method according to (Hevner et al., 2004)		**Application to this thesis**	**Sections applying the method**
Observational	Case Study and Field Study	In this thesis, a case study applying the LPMS to the public sector is presented. Various aspects are evaluated by this study.	7.3.1, 7.3.3 - 7.3.5
Analytical	Static analysis (e.g. complexity)	Evaluation in terms of completeness, expressiveness, adaptability, extensibility	7.3.1 - 7.3.5
	Architecture analysis	Evaluation in terms of the fit to SOA principles, fit to the SOA4All architecture, and fit	7.3.1

		to the internet and web serving as an architecture for LPM since services are available in the web	
	Optimization	Optimization is not yet subject to the design of LPM and the LPMS	
	Dynamic analysis (e.g. performance)	Evaluation in terms of runtime behaviour and service selection. The dynamic analysis is not focused in this thesis and depends on the LPMS implementation and the user.	7.3.1
Experimental	Controlled experiment	Evaluation through Workshops	7.3.1, 7.3.4, 7.3.5
	Simulation	Simulation is not applied in this thesis.	
Testing		Not focused in this thesis, since no special testing methods are required.	
Descriptive	Informed argument	Evaluation in terms of literature research. The arguments provided in this thesis are based on interviews, literature, and surveys.	7.3.1 - 7.3.5
	Scenarios	No scenarios are constructed. LPM is applied to the public sector to demonstrate usability.	

This section 1 starts with an introduction of the evaluation metrics, their categorization, and structuring in section 7.1. Each metric category refers to a strategy of technical, individual, organisational, or economic evaluation. In addition, for each metric is specified whether the evaluation is performed based on literature and general requirements or based on the public sector use-case.

Besides the static analysis, the evaluation is performed dynamically and observation-based through surveys and modelling workshops. In this thesis, the dynamic evaluation is focusing on the user behaviour to model and execute processes. The user is the main part of the proposed software solution. Hence, the evaluation metrics for the dynamic evaluation are defined with respect to the user interaction.

The proceeding for the empirical evaluation is described in section 7.2. The evaluation itself is then covered by section 7.3. For each metric the evaluation means and the result are presented. Besides the enablers of the LPMS, potential risks are discussed in section 7.4. The conclusion in section 7.5 closes this section 1.

Table 41 presents an overview of the evaluation structure of this thesis. For each of the evaluation strategies, the beneficiary perspective, the evaluation approach, and the application area as described above are indicated.

Table 41: Overview of evaluation structuring

		Evaluation strategies			
		Technical	Individual	Organisational	Economic
Beneficiaries	Service Technology	●			
	Organisations			●	●
	End-user		●	●	●
Evaluation approach	Analytical	●	●	●	●
	Empirical		●	●	●
Application area	General	●	●	●	●
	Public sector use case		●	●	●

7.1 Evaluation Metrics

Table 42 presents the metrics for evaluating LPM and the LPMS. Like the requirements, the metrics are grouped into technical, individual, organisational, and economic metrics. However, this distinction is not always unambiguous, several metrics fit to more than one category. The very left column indicates the metric. The column in the middle describes the application of the design science evaluation method of Table 40 to the metric. Finally, the right column describes the type of evaluation. The type of evaluation is differentiated according to the evaluation based on general requirements and literature as well as the public sector (PS) use-case-based evaluation. How the metric is concretely measured in this thesis is described in section 7.3. In general, the metrics evaluate the envisaged improvement of interacting with a

BPM solution. Hence, the realization of the LPM design principles and the LPMS are evaluated.

An international standard for the evaluation of software quality is the ISO/IEC 9126[24]. In this thesis, functionality and usability are the main criteria from the proposed set of characteristics. This is in line with the description of a new research paradigm and the according prototype rather than the specification of a mature software product.

Table 42: Evaluation metrics for the LPMS

Metric	Design Science Type	Type of evaluation
Technical metrics		
Consistent BPM stack	Static analysis	General analysis
Consistency of design process to create an executable process model.	Static and observational analysis	General analysis
Handling of semantic annotations	Static and observational analysis	General analysis, PS use case
Integration of heterogeneous services	Static analysis	General analysis
Service selection, binding, replacement, and adaptation at various stages	Static analysis	General analysis
Individual metrics		
Executable BPM functionality for business users	Static analysis	General analysis
Potential number of users without IT knowledge	Static analysis	PS use case
Usability, Simplicity, and	Observational, dynamic, and	General analysis, PS

[24] Detailed information about ISO/IEC 9126 can be found here: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22749

understandability	static analysis	use case
Simple graphical symbols	Static and observational analysis	General analysis, see survey
Suitable abstraction views	Static analysis	General analysis
Easy information search	Static analysis	General analysis, PS use case
Type of interaction (commercial, prolonged, multiple, minimal)	Static and dynamic analysis	PS use case
Hurdle of taking process ownership	Observational and dynamic analysis	PS use case
Organisational metrics		
Potential number of users according to task duties	Static analysis	PS organizations and processes
Community acceptance, suitability for collaborative modelling	Static analysis	General analysis
Economic metrics		
Strategic importance (core strategic, important, useful, contributing, exploratory)	Observational and dynamic analysis	PS use case
Generality and applicability to various scenarios according to March and Smith (March & Smith, 1995)	Static analysis	General analysis, PS use case
Degree of reuse as part of efficiency and effectiveness according to March and Smith (March & Smith, 1995)	Observational, dynamic, and static analysis	General analysis, PS use case
Modelling time as part of efficiency and effectiveness according to March and Smith (March & Smith, 1995)	Observational and dynamic analysis	PS use case processes
Needed expertise, training effort as part of efficiency and effectiveness according to March and Smith (March & Smith, 1995)	Observational and dynamic analysis	PS use case
Utilization of existing infrastructure and software assets (Palmer, 2009)	Static analysis	General analysis
Use of internal development and support resources and avoid high workload for IT resources (Palmer, 2009)	Static analysis	General analysis
Faster Time to market (Palmer, 2009)	Static analysis	General analysis
Lower initial application integration costs	Static analysis	General analysis

(Palmer, 2009)		

For the evaluation of the LPML, special language evaluation categories are applied. Table 43 covers the LPML evaluation. Hence, for each metric the metric categories, the design science type, and the type of evaluation are presented. The design science type follows the LPM evaluation as presented in Table 40. Again, the type of evaluation covers the fact to be based on general requirements and literature or on the use-case.

Table 43: Special evaluation metrics for the LPML

Metric category	Metric	Design Science Type	Type of evaluation
Correctness	Syntactic and semantic correctness	Static analysis	General analysis
	Uniqueness and canonical, exchangeable format	Static analysis	General analysis
	Coherency of abstraction layers	Static analysis	General analysis
Completeness and expressiveness	Ontological completeness	Static analysis	General analysis
	Pattern-based completeness	Static analysis	General analysis
	Use case scenario coverage	Static analysis	PS use case
Adaptability and Extensibility		Static analysis	General analysis, PS use case

7.2 PROCEEDING FOR EMPIRICAL EVALUATION

Besides the static evaluation, this section describes how the dynamic evaluation of the LPMS is performed. According to the design science approach described by Hevner et al. (Hevner et al., 2004), this section describes the setup of the experimental, dynamic analysis. Both descriptive and analytic design, as described by Oppenheim (Oppenheim, 1992), is applied to this thesis through a survey and experiments. While the purpose of a descriptive design is to identify the amount of participants to be assigned to various groups, the analytical design seeks to reveal explanations, causalities, and relationships. The main means to perform an evaluation are surveys and experiments. Best evaluation practice is a combination of both means in order to use the results of the one to improve the other (Oppenheim, 1992). The reason behind is on the one hand that an experiment is often unrepresentative, deals with artificial situations, and fails to achieve an appropriate degree of precision and control. On the other hand surveys provide a limited ability to control or manipulate variables and are limited in proving causal relationships.

Mainly, two steps have been followed to perform the evaluation of this thesis: a focus group evaluation and a comprehensive expert and business user evaluation. While the focus group evaluation is targeting business users, as described in the previous section, the expert and business user evaluation targets all kind of users.

- Focus group: In a focus group evaluation a special group of users is questioned about their opinions on the research topic (Nielsen, 1993). Such a focus group evaluation has been conducted during the EUD4Services Workshop held in conjunction with AVI 2010 in Rome in May 2010. The workshop participants have been researchers and end-users of various organisations. Before the discussion, a couple of introductory slides have been presented[25]. The feedback the workshop participants gave was generally positive. They liked the idea of having a process modelling solution for business users. Further, they supported the idea of keeping business users free from execution details.
- Experts and business users: Expert-based evaluation is often guided by heuristics for usability evaluation. A reality-based, typical scenario that has to be performed by a business user is given to the experts. The evaluation feedback should then comprise comments about usability problems that might be assigned to the usability guidelines described in the heuristics. The expert and business user evaluation is applied in a broader sense in this thesis. Besides usability, various aspects are evaluated.

[25] The introductory slides shown in the workshop can be found here: http://cslab.dico.unimi.it/EUD4Services/slides/EUD4Services-lombardi-demo.pdf.

For LPM, the expert and business user evaluation is not differentiated. In the following is described how a survey and workshops gather feedback from expert and business user evaluation target groups.

The aim of the survey set up for the expert and business user evaluation has been to figure out the usefulness of end-user empowerment in the BPM area and what users think about the main LPM ideas. A detailed description of the hypothesis is covered by section 7.2.2. For each of the evaluation metrics defined in section 7.1, a variable is defined that is statically analysed, questioned in the survey, or measured in a workshop. However, before describing the survey, the setup of modelling workshops is covered by the following section 7.2.1.

7.2.1 Workshops

The evaluation of some metrics is best performed by modelling workshops. In these workshops, users have to conduct modelling experiments. The workshops should include all three target groups as described in the previous section. However, the workshops do not explicitly differentiate according to the target group. In the workshops, the user profiles are tracked in order to allow for an analysis based on the target user groups.

A typical workshop should include both the focus group evaluation and the expert and business user evaluation. Since users without any previous knowledge participate in the workshops, an introduction is needed. As introductory steps the workshop participants should be asked about business user development of service-based software in general. In terms of process modelling, information about ways to dynamically compose services should be gathered. Besides control-flow modelling, data-flow modelling as well as assisted modelling is proposed for LPM. Further, experiments should be set-up in order to gather a possibly complete list of requirements and feedback about early design stages. A workshop helps potential users to learn about the LPMS and its benefits for dynamically composing processes.

General requirements for conducting workshops concern the amount and the profiles of participants. A number of at least 12 participants that might be divided into groups might fit very well the purposes. The participants' profiles should match the target group profiles.

A description of a sample workshop that has been conducted in the context of SOA4All (see section 3.4.1) can be found in section 9.1.2. This workshop has been conducted at an early stage of the LPMS. It has covered the general evaluation of service composition approaches and the LPM approach. Besides the software usage, the participants' backgrounds, their requirements, and their ideas have been analysed. Further, a couple of risks have been identified that are described in section 7.4.

The modelling experiments for the LPM evaluation are described in section 7.3 in the context of the according evaluation metric.

7.2.2 Survey

The survey conducted in the context of this thesis had been designed for expert and business users as described in section 1.3. The participants surveyed are expert and business users from SAP, SAP Research, SAP customers in the public sector, University of St. Gallen, City of Winterthur (Switzerland), City of Muenster (Germany), and the SAP BPX Community. They all have to fulfil BPM-related tasks in their work environment. The complete survey questionnaire can be found in the Annex in section 9.3.

In order to brief the survey participants about LPM, an introductory video and a set of slides has been provided on the start page of the survey. To check the participants' understanding the first questions have been set around the video respectively the set of slides. Afterwards, the main questions about the LPM design principles have been asked. The results of the survey questions are presented in section 7.3 in the context of the according evaluation metrics.

In order to figure out the target user group a potential respondent belongs to, the user profile including modelling and IT experience has been surveyed as well. Hence, the answers given in the survey could be related to the target user group as described in section 1.3. Another option to assign users to one of the target groups is to let them define the term "service" and then analyse the IT relation given in the answer.

As previously mentioned, the aim of the survey has been to figure out the usefulness of end-user empowerment in the BPM area and what users think about the main LPM ideas. In the following Table 45 the hypotheses for the survey are listed. In the right column, the related research question (RQ) (see Table 44) is assigned to the hypothesis.

Table 44: Research Questions (RQ) as stated in section 1.2

RQ1: How might business users be enabled to model executable processes in a lightweight way?
RQ2: What are the design principles for artefacts supporting the business user in process modelling and executing?

RQ3: How do artefacts, such as a language and tools for LPM, look like?
RQ4: How does the LPML reflect these design principles?

The survey only covers hypotheses for RQ1 and RQ2. Questions about RQ3 and RQ4 require a deep understanding of the subject area and hence would require a high effort to brief the survey participants. Further, the respondents targeted are typically business users that are by the definition of LPM not expected to have that deep understanding of process modelling.

Table 45: Survey hypotheses

Hypothesis	**Related research question**
Business-user empowerment in the BPM area is useful for organisations.	RQ1
LPM is applicable to various, heterogeneous business areas.	RQ1
The user prefers to model processes without high training effort.	RQ1
The user prefers to be kept free from execution details.	RQ2
The user prefers a graphical abstraction model to a textual programming model. The graphical abstraction comprises a minimal element set the business user might understand.	RQ2
Providing semantic annotations to activities serving as service categorization, rather than providing service specifications, is easier for the user.	RQ2
Semantic annotations support the user in modelling data flow, facilitate optimization through non-functional properties, and facilitate reuse.	RQ2
Context information supports the user in the modelling procedure.	RQ2

The survey had been set up as an online survey. 21 participants answered all questions and terminated the survey. Special questions have been answered by up to 35 participants. An introducing question about the respondents' understanding of the subject revealed a good understanding. The average value has been 3.97 on a 5-point Likert scale as shown in

Table 46.

Table 46 : 5-point Likert scale as applied in this thesis

Value	Meaning
1	Strongly disagree
2	Disagree
3	Undecided
4	Agree
5	Strongly agree

The main subject of education of the respondents has been Economy and Management (31%), Computer Science (56%), and Information Management (13%). The according main subjects of work have been Economy and Management (35%), Computer Science (43%), and Information Management (22%). Concerning IT training, 30% of the respondents had a self-taught training, 13% an introduction to office software, 17% a significant IT training, and 74% an IT-related degree. The following Table 47 and Table 48 indicate the software and BPM experience of the surveyed users for the entire group of respondents and for business users.

Table 47: Software experience of surveyed users

Software	Amount of users experienced in	Amount of business users experienced in
Windows	100%	100%
Office software	96%	80%
Programming software (Visual Basic, Java, C, C++, SQL, etc.)	78%	20%
Web applications (iGoogle, Facebook, etc.)	91%	100%
Mashups (Yahoo Pipes, etc.)	39%	20%
Service composition	43%	20%

Table 48: BPM experience of surveyed users

Software	Amount of users experienced in	Amount of business users experienced in
Petri nets	43%	0%
UML activity diagrams	83%	20%
BPMN	78%	60%
WS-BPEL	35%	0%
YAWL	9%	0%
Flow charts	70%	60%
EPC	26%	0%

The respondents have been aged between 20-30 (55%), 30-39 (32%), 40-49 (9%), and >59 (4%). Their main professions have been student (14%), researcher (55%), and non-researcher employee (27%).

Table 49 : Favourite modelling languages or tools of respondents

Modelling language or tool	Percentage of respondents feeling the language or tool highly usable		Percentage of respondents expecting a high training effort	
	All users	Business users	All users	Business users
Flow charts in Office Software, e.g. in MS Power Point	52%	80%	4%	0%
Flow charts in MS Visio	35%	40%	17%	0%
EPC and ARIS	17%	20%	22%	20%
BPMN	43%	60%	35%	20%
WS-BPEL	17%	Not applicable	30%	Not applicable
BPM Suite (e.g. SAP NetWeaver, IBM Rational, etc.)	22%	20%	22%	20%

Table 49 reveals the survey respondents' experience in handling existing modelling languages and tools. Since no responding business user used WS-BPEL, no results might be shown for the evaluation of WS-BPEL by business users.

Especially the evaluation of BPMN is interesting for this thesis. 20% of business users heard about it and used it. The business user respondents feel it highly usable (60%), don't think that high training effort is required (20%), and none of them faces difficulties. In contrast, only 45% of IT-experts feel BPMN highly usable and 45% think that the language requires a high training effort. The interpretation of the differences is that business users see BPMN as a graphical modelling tool that is easy to use. However, the IT people see BPMN as a graphical modelling tool that has to be enhanced by execution information. Since this requires much more effort, the IT people feel BPMN less usable and expect more training effort to use the language effectively and efficiently.

As well, none of the business users heard about and used BPEL and YAWL that are executable languages. Again, this shows that business users don't think at process execution when modelling processes.

The survey further questioned the use of Eclipse-based modelling tools. However, none of the respondents used one of those tools. Unfortunately none of them used Lombardi Blueprint either that claims as well being a tool for lightweight process modelling.

A detailed report of the survey results and an assignment to the evaluation metrics is given in the sections 7.3 and 7.5.

7.3 LPM Evaluation Results

After having introduced the evaluation metrics, the target users, and the evaluation proceeding, this section covers the evaluation results. The section starts with the evaluation of the LPM approach to compose services. Afterwards, the evaluation of the LPMS is addressed in terms of correctness, completeness, and expressiveness, adaptability and extensibility, and usability.

As described in Table 42 and Table 43, a couple of metrics are measured by observational analysis. The observation of these metrics is performed in workshops as described in section 7.2.1. In the following, the experiments that have to be conducted in these modelling workshops are presented. These experiments serve for evaluating certain aspects of the LPMS and the LPML. The experiments haven't been conducted yet, since this thesis focuses on the design of LPM. Furthermore, at the time when this thesis has been written, no integrated prototype has existed that could have been used in experiments. In the context of the project SOA4All these experiments will be conducted as soon as an integrated prototype will exist.

For those metrics that are evaluated through the survey, the 5-point Likert scale as presented in Table 46 is applied.

7.3.1 LPM Approach to Compose Services

Here, the results of the general evaluation of the LPM approach to compose services are presented. The evaluation metrics presented in section 7.1 are measured by evaluation means that are described in this section 7.3.1. Further, the results of the measurements are described. For those evaluation metrics that should be analysed dynamically the evaluation is based on the survey and the public sector use case. If a use-case-based evaluation is not feasible for a certain metric, a static analysis will be performed. The feedback given by the focus group of the EUD4Services Workshop (see section 7.2) is integrated into the result descriptions of the evaluation metrics if applicable.

Table 50 covers the evaluation of the technical metrics as described in section 7.1. All technical metrics are analysed statically through reasoning on literature and the LPM approach objectives. The consistency of the LPMS should be analysed based on a use case as well. For example, the public sector processes and services might be analysed in order to figure out the average user interaction or automated execution.

Table 50: Evaluation of technical metrics

Metric	Evaluation Means	Result
Consistent BPM language stack	Static: Analysis of the LPML metamodel	Through the use of a common metamodel, the abstract and the canonical layer are in line.
Consistency of the design process and the tooling support	In order to guarantee the consistency of the design process to make the process model executable, a static analysis of the metamodel is applied. Survey: Question about understanding of shown scenario and underlying design process.	Static analysis: The shared canonical process model guarantees consistency. The transformation of the LPML model into an executable process language uses all information stored in the LPML models.

		The LPML and tools are strongly aligned and built with respect to each other. Survey: The question about the understanding of the shown scenario and the underlying design process results in an average value of 3.97 revealing that the design process is consistent.
Handling of semantic annotations	Static: Describe the handling of the semantic annotations by the LPMS. Workshop experiment: Let users provide semantic annotations. Check whether these annotations might be used by the tools for service search and composition.	Static analysis: The LPML provides elements for semantic annotations of various types. The LPM tools either process the semantic annotations themselves or interact with additional tools, such as reasonners. Focus group: Semantic annotations are a valuable approach to specify services.
Integration of heterogeneous services	Static: Describe service interface handling through tools	Static analysis: The LPMS works with an abstraction of services that is independent of the service implementation technology. Currently, the handling of WSDL, REST, and semantically described services is allowed.
Service selection, binding, replacement, and adaptation at various stages	Static: Describe abstraction concept	Static analysis: The LPMS allows for optional service specification and binding through favourites from the process editor, manual search, goal and activity descriptions to be instantiated at runtime, and the interaction of the discovery engine and the execution engine at runtime.

The evaluation of the individual metrics is covered by Table 51.

Table 51: Evaluation of individual metrics

Evaluation Metric: Executable BPM functionality for business users
Evaluation Means: The evaluation of the executable BPM functionality has been performed through a static comparison of the functionality of existing BPM tools in terms of modelling, execution, service integration, and business-user focus. Furthermore, a focus group has been asked about the BPM approach. Lastly, the survey conducted in the context of this thesis shows valuable results as well.
Evaluation Result: General analysis: A comparison of the LPMS to other BPM suites reveals various benefits of using the LPMS. The LPMS might be used in an open environment and allows for the integration of heterogeneous web services. Furthermore, the LPMS targets the business users through providing an easily understandable process modelling language and keeping the user free from execution details. Focus group: The participants of the focus group agreed that the LPMS is a very useful approach. **Survey results:** A question whether the LPMS is a real innovation revealed an average value of 3.66, whether the solution would be potentially used for business tasks 3.45 (business users 2.80), whether in a non-business context 3.07 (business users 3.20), and whether people would share services and processes an average value of 3.28, each on a 5-point Likert scale (see Table 46). Furthermore, the question of whether business users would benefit from the LPMS as a BPM solution revealed an average value of 3.78. Each presented value is based on the 5-point Likert scale (see Table 46).
Evaluation Metric: Potential number of users without IT knowledge
Evaluation Means: The evaluation of the potential number of users without IT knowledge should be performed through workshops. In these workshops an analysis of the used functionalities in the process editor should be performed. By relating the used functionalities to the respondent's experience, the potential number of users might be estimated. Further, the use of wizards should be analysed. Users using wizards will potentially prefer from the LPMS. Another workshop experiment to be performed is to analyse the process model quality of users having joined different trainings and or having different experiences. During the workshop experiment, those heterogeneous user groups should be asked to model a given scenario from scratch. Based on the relation of the experience and the modelling results, an analysis might be performed about how much expertise is required for specific modelling procedure steps.
Evaluation Results:

The target users of the LPMS are business users. Hence, in order to identify potential users, the survey questioned the BPM experience of the respondents. In particular, the answers of the respondents with an economic background are of interest. The five respondents with an economic background are experienced in – the number in brackets indicate the amount of respondents - Petri Nets (0), UML Activity Diagrams (1), BPMN (3), WS-BPEL (0), YAWL (0), Flow charts (3), and EPC (0). The average value of whether the LPMS process editor is easy to use from a business user perspective is 3.40 which is almost the same as for all participants (3.52). The value of whether they understand easily the graphical process representation is 3.40 for business users, compared to 4.00 for all respondents, of whether they don't want to be informed about technical details is 3.50 for business users, compared to 3.38 for all respondents, and whether they miss important information is 2.40 for business users, compared to 2.78 for all respondents. Each presented value is based on the 5-point Likert scale (see Table 46).

Evaluation Metric:

Usability, simplicity, and understandability: See section 7.3.5 for a detailed evaluation.

Evaluation Metric:

Suitable abstraction views

Evaluation Means:

The evaluation of the abstraction views is performed by a static analysis, a workshop experiment, and the evaluation survey.

Evaluation results:

In terms of the static analysis, the fact is accounted, that the LPMS comprises two abstraction layers. One of the layers is abstract and dedicated to business users, the other one is canonical, hidden from the user, and only processed by tools. Through the application of simplicity and understandability principles, the graphical abstraction layer is suitable for the business user. The canonical formats of the LPML process models are suitable for the LPMS tools.

The workshop experiment should check whether users are able to provide the necessary annotations in order to make processes executable. Hence, the users are asked to model activities based on a given scenario with specific information about the activities. By analyzing whether the modelled information is sufficient, evaluation information about the suitability of the abstraction layers might be gathered.

In terms of the survey results, the question about whether the graphical symbols were clearly and intuitively understandable revealed an average value of 3.65. A question about users preferring graphical models to textual models resulted in an average value of 3.91. Finally, the question about whether users missed important information in the graphical model revealed an average value of 2.78. Each presented value is based on the 5-point Likert scale (see Table 46).

Evaluation Metric:

Easy information search

Evaluation Means:

The information search mainly focuses on the search for services to be bound to process activities. The evaluation of the service search is performed by a workshop experiment and survey questions.

Evaluation Result:

Focus group: The participants agreed on a high value of having automated search based on semantic annotations.

Workshop: Like for the evaluation of the suitable abstraction views, a workshop experiment should be performed to check whether the information provided by the users is appropriate to find fitting services. Hence, it might be figured out whether the service search is easy to perform.

Survey results:

Users have been asked whether they prefer to indicate a service category, rather than a concrete service. The average result value is 3.70. However, the trust that search tools will find the best fitting service is low with an average value of 2.87. The business users show more trust with an average value of 3.20. The average result of the question about the preference to select a concrete service by the user himself is 3.74 and for business users 3.40. Each presented value is based on the 5-point Likert scale (see Table 46).

Evaluation Metric:

Type of interaction (commercial, prolonged, multiple, minimal)

Evaluation Means:

To evaluate the type of interaction with the LPMS, survey questions about the potential use of the LPMS are asked.

Evaluation Result:

The type of interaction has been mainly differentiated into business and non-business usage. The survey revealed an average value for business usage of 3.45 and for non-business tasks of 3.07, each on a 5-point Likert scale. Furthermore, the survey asked about potential usage areas. The following values indicate the amount of all users intending to use the LPMS, in brackets the amount of business users is shown. 70% (60%) of the respondents intend to use the LPMS in the user’s functional area, 39% (20%) in HR processes, 65% (40%) in administrative processes, and 26% (0%) for private use.

Evaluation Metric:

Hurdle of taking process ownership

Evaluation Means:

The evaluation of hurdles of taking the process ownership has been performed by the

survey.
Evaluation Result: The survey revealed that 30% of the respondents fear a lack of process understanding preventing them from taking the process ownership. Only 17% of the respondents don't want to take the ownership due to the fact that they didn't model the processes themselves. 22% fear various process performers, 30% think that the integrated processes are not controllable, 35% fear the lack of exception and error handling, and 39% fear that processes could be error-prone. Further answers of the respondents had been that they don't want to be responsible for a process if it is not related to personnel core activities, that interaction with processes of other humans cannot be modelled, and that the work environment changes too fast for effective process modelling.

Table 52: Evaluation of organisational metrics

Evaluation Metric: Potential number of users according to task duties
Evaluation Means: The potential number of users according to their task duties is mainly evaluated by survey questions about the usage areas and the frequency of process modelling.
Evaluation Result: Focus group: The focus group agreed that a high number of users are expected. **Survey results:** The survey question about the usage area resulted in an average value for business usage of 3.45 and for non-business tasks of 3.07. Both presented values are based on the 5-point Likert scale (see Table 46). As already described for the type of interaction, the following values indicate the amount of all users intending to use the LPMS, in brackets the amount of business users is shown. 70% (60%) of the respondents intend to use the LPMS in the user's functional area, 39% (20%) in HR processes, 65% (40%) in administrative processes, and 26% (0%) for private use. In addition, the survey questioned the frequency of process modelling. 39% of the respondents answered to model processes once a year, 35% once a month, 17% once a week, and 9% once a day.
Evaluation Metric: Community acceptance, suitability for collaborative modelling
Evaluation Means: The evaluation of the community acceptance is performed by a static analysis of the LPMS in terms of potential extensions for collaborative modelling. Further, the survey asked a question about whether people desire to share services and processes.
Evaluation Result:

Static analysis: In order to allow for collaborative modelling, additional information about process models has to be attached. Concretely spoken the LPMS has to support sharing, searching, reusing, and discussing process models. This requires information, such as key words, tags, plain text (description, explanation), categories (like in Forums), descriptions dedicated for reuse of process models, explanations, discussions, document editing history, monitoring information, marks, comments, recommendations, ratings, or votes. Through the semantic annotations and, in particular, the annotation type *metadata* this information might easily be attached. **Survey results:** A survey question about the desire to share services and processes resulted in an average value of 3.28 (2.60 for business users). Each presented value is based on the 5-point Likert scale (see Table 46).

Table 53: Evaluation of economic metrics

Evaluation Metric: Strategic importance (core strategic, important, useful, contributing, exploratory)
Evaluation Means: The evaluation of the strategic importance of the LPMS has been performed through the survey.
Evaluation Results: The survey revealed that 13% of the respondents think that the LPMS could be core strategic, 87% think it should be supporting. The average value of whether service and process composition by business users is useful is 4.0, and the value of whether service and process composition by business users could break organizational rules and policies is 3.55. Each presented value is based on the 5-point Likert scale (see Table 46).
Evaluation Metric: Generality and applicability to various scenarios according to March and Smith (March & Smith, 1995)
Evaluation Means: The evaluation of the generality is performed by a static analysis and survey questions.
Evaluation Result: Static analysis: The LPMS is not restricted to a specific domain or scenario. General extension mechanisms are provided through the semantic annotations. Annotation types might be further added. In addition, the LPMS might integrate all kinds of services. **Survey results:** The survey revealed that respondents intend to use the LPMS in business and non-business contexts. In terms of the business context, the LPMS is applicable to various areas. The survey question about the usage area resulted in an average value for

business usage of 3.45 and for non-business tasks of 3.07. Each presented value is based on the 5-point Likert scale (see Table 46). The following values indicate the amount of all users intending to use the LPMS, in brackets the amount of business users is shown. 70% (60%) of the respondents intend to use the LPMS in the user's functional area, 39% (20%) in HR processes, 65% (40%) in administrative processes, and 26% (0%) for private use.
Evaluation Metric: Degree of reuse
Evaluation Means: The degree of reuse has been evaluated through survey questions. Through the pattern and template repository as a potential extension of the LPMS, an additional mechanism for reusing process parts might be provided. This repository could be analysed as well in order to evaluate reuse of processes and its parts. However, the repository analysis doesn't make sense until a critical mass of processes and parts of it to be reused is available in the repository.
Evaluation Result: The survey asked various questions related to reusing processes and parts of it. A basic question about whether people suppose themselves to be able to open and adjust a predefined process model indicated an average value of 3.86. The question about people preferring to model their processes themselves revealed an average value of 3.23. Although this value seems to be low, the interpretation doesn't contradict the reuse paradigm. Reuse might save modelling time through integrating existing parts of the process or adjusting existing processes. However, the users still model the processes themselves. Further, examples of successful business process modelling stimulate users to model their processes (3.78). Survey respondents trust other users to model processes and parts to be reused (3.78). In addition, users think it's useful to search for existing processes and parts to be reused before starting modelling (4.13). Only few respondents (average of 2.13) think, that there's no use of searching for existing process models. The average value of people that only trust process models that are in line with well-defined quality standards is 3.23. Each presented value is based on the 5-point Likert scale (see Table 46).
Evaluation Metric: Modelling time
Evaluation Means: An important criterion for the success of the LPMS is the time reduction to model an executable process. Therefore, a workshop experiment is set up letting various user groups model the same business process in different languages, such as LPML, BPEL, YAWL, and BPMN. The time to model the target processes is measured. Furthermore, the effectiveness and efficiency of modelling executable processes might be evaluated by comparing the procedure to model processes and make them executable for various languages.

Evaluation Result: In this thesis, the design of the LPMS is focused. Hence, the described modelling workshop has to be conducted as soon as the LPMS is implemented.
Evaluation Metric: Needed expertise, training effort
Evaluation Means: The evaluation of the needed expertise and training effort is performed in a modelling workshop and by survey questions. A workshop experiment might be set up targeting users with low BPM experience. Two groups of users get the same general BPM training. Afterwards, one group models a process with one of the existing process modelling languages and another group with the LPML. Hence, the training effort might be compared by analysing the quality of the modelled processes. Furthermore, an option to figure out the training effort is to provide different trainings or introductions to the users. Provided that the two user groups have similar modelling expertise, the modelling results of the two groups might be compared.
Evaluation Result: The survey asked various general questions about a potential modelling training. The question about whether people prefer to use the LPMS without training resulted in an average value of 3.57. The average value of people saying that attending a training course helps them to start modelling is 3.48 and 4.40 for business users. Hence, a training for the LPMS should be provided, however, the training effort should be kept low. Each presented value is based on the 5-point Likert scale (see Table 46).
Evaluation Metric: Utilization of existing infrastructure and software assets
Evaluation Means: The evaluation of the utilization of the existing infrastructure and software assets has been performed by a static analysis of the LPMS.
Evaluation Result: The LPMS is fully service-oriented. A general analysis revealed that through the integration of various service types, such as WSDL, REST, or SWS, existing infrastructures and software assets might be reused.
Evaluation Metric: Use of internal development and support resources and avoid high workload for IT resources
Evaluation Means: The evaluation of the distribution of the internal development and support resources

has been performed by a static analysis of the LPMS.
Evaluation Result: The goal of the LPMS is to enable business users to model executable processes. Hence, through shifting programming effort from the IT department to business departments, the workload for the IT department might be reduced. Furthermore, through the intuitive use of the LPMS, the support through the IT department might be kept low.
Evaluation Metric: Faster Time to market
Evaluation Means: The evaluation of the time to market has been performed by a static analysis.
Evaluation Result: Again, the goal of the LPMS is repeated: Enable business users to model executable processes. Through the LPMS, business users might model and execute new processes related to product management. Time-consuming information exchange procedures between business and IT departments might be avoided. New software artefacts and processes might be created more quickly. Hence, the product's time to market might be reduced.
Evaluation Metric: Lower initial application integration costs
Evaluation Means: The evaluation of the initial application integration costs has been performed by a static analysis.
Evaluation Result: The LPMS is based on services that facilitate the integration of applications. Furthermore, the discovery, binding, and composition of services are supported by tools. Ideally, these activities are performed completely automatically. Hence, the initial application integration costs might be kept low.

In order to evaluate the LPM approach of composing processes through control-flow modelling, an experiment has been conducted comparing the control-flow modelling to data-flow modelling and assisted modelling. The experiment is described in (Abdallah Namoune et al., 2009) and documented according to the Design Rationale (Jarczyk, Loeffler, & Iii, 1992). Design rationale aims to support system designers and documents the decisions for the selected approach regarding alternative ways.

The experiment focused on three approaches to compose processes:

- Data-flow to create service mashups
- Control-flow as in typical process models
- Assisted modelling using wizards that guide the user in a stepwise procedure

The main results that emerged from the workshop revealed that the data-flow representation was most difficult to model while assisted composition was easiest. The control-flow representation however, had not been ranked significantly more difficult than assisted modelling. During the experiment, the users' backgrounds have been analized. It came out that modellers without IT knowledge favoured the assisted modelling, while the control-flow approach was favoured by people that understood programming. Data-flow modelling was regarded to require high understanding effort due to service interoperability and the according data type and value matching. Hence, the results of this workshop lead to the decision of providing a tool for control-flow modelling including wizards to support assisted modelling and to strongly support the user in data flow modelling.

7.3.2 LPML Correctness

The evaluation of the LPML correctness is subject of this section. As described in Table 54, the LPML is evaluated in terms of syntactic and semantic correctness, uniqueness, and coherency.

Table 54: Evaluation of the LPML correctness

Metric	Evaluation means	Result
Syntactic correctness	Static analysis of the LPML metamodel and the design rules	The syntactic correctness of the process models is made sure through the LPML metamodel and the according design rules as described in section 4.2.2.1 and in section 4.2.2.2.
Semantic correctness	Static analysis of the LPML metamodel	The semantic correctness of the LPML is given through the definitions in the metamodel.
Uniqueness and canonical, exchangeable format	Static analysis of the LPML metamodel	The process models are represented by a canonical process representation. Any element of the LPML is characterized by an unambiguous ID. Further, the canonical format of the LPML process ensures the exchangeability.
Coherency of different layers	Static analysis of the LPML metamodel	The two abstraction layers (graphical and canonical layer) are not different

		models that have to be transformed. Both layers have the same underlying process model (the canonical format).

7.3.3 Completeness and Expressiveness of the LPML

In this section, the LPML is evaluated in terms of completeness and expressiveness, as partially described in Schnabel et al. (Schnabel, Born et al., 2009). In particular, the completeness and expressiveness is evaluated in terms of ontological completeness and of a pattern-based analysis. The evaluation based on ontological completeness has been performed in various studies, such as a study for reference models by Fettke and Loos (Fettke & Loos, 2003). In line with the design science, the completeness and expressiveness is evaluated through a static analysis. Therefore, as defined in Schnabel et al., a reference set of BPM concepts and language constructs has been prepared in a first step. This first step is in line with the approach of Weber et al. (Weber et al., 2008) who defined a framework for the evaluation of the expressiveness of Process-aware Information Systems (PAIS) in terms of suitability to support process changes. The reference set has been built with respect to the target usage of business users. Therefore, the well-established Bunge-Wand-Weber (BWW) models (Bunge, 1977; Wand & Weber, 1989) have been used. In particular, the representation model has been used in order to measure the ontological completeness. Further, a benchmark set of 20 control flow patterns from workflow systems (van der Aalst, ter Hofstede, Kiepuszewski et al., 2003) and six communication patterns from Enterprise Application Integration (EAI) systems (Ruh, Maginnis, & Brown, 2001) has been selected as part of the reference set.

In the second step, an attempt has been performed to model the previously defined concepts and constructs of the reference set using the LPML. Three resulting cases emerged, the direct match, the indirect match through another LPML construct or combination of LPML constructs, and a gap.

Finally, the indirect mappings and gaps had been presented to experts from the public sector use case (see section 1). The presentation had been performed in order to gather feedback about the significance of any mismatches in real scenarios and hence, to allow for a judgement about the scenario coverage and the language richness.

In the following, the results of this process are described for ontological completeness and pattern-based analysis of the LPML.

7.3.3.1 Ontological Completeness

The proceeding to evaluate ontological completeness has already been described in Schnabel et al. (Schnabel, Born et al., 2009). In order to gather a reference foundation for analysing the concepts coverage for the development of information systems, Wand and Weber (Wand & Weber, 1989) have studied the philosophical works of Bunge (Bunge, 1977) and produced the Bunge-Wand-Weber (BWW) ontology. The BWW ontology comprises three parts as follows for the description of information systems (IS) (Gehlert, Pfeiffer, & Becker, 2007; Wand & Weber, 1993):

- The representation model stating that an IS should be a faithful representation of a real world proportion.
- The state tracking model stating that an IS has to be embedded into the real world and must track its changes.
- The good decomposition model stating that a good decomposition is strongly in line with the structure and the dynamics of the modelled real system.

Wand and Weber propose a 1:1 mapping of the representational model to any modelling grammar used to model ISs. By reaching this 1:1 mapping, the grammar is said to be ontologically complete. Hence, the representational model of the BWW models is a well-established approach for evaluating the completeness of a language. In the following, the representation model is focused. It comprises a set of constructs that are sufficient to represent the structure and the behaviour of an arbitrary system (Peter Green, Rosemann, Indulska, & Manning, 2007). The use for such evaluations has been extensively validated through the application in over 30 projects analysing various grammars (Peter Green et al., 2007). In the area of process modelling, Green et al. (Peter Green et al., 2007) have mapped the BWW representation model in one such study to the constructs of BPEL. In another study, Recker et al. have mapped BPMN to BWW constructs (J. Recker et al., 2008). The authors of the BPMN-BWW-mapping see the BWW representation model as a benchmark for the evaluation of representational capabilities of modelling techniques. Furthermore, Recker and Mendling (J. Recker & Mendling, 2006) have compared BPMN and BPEL based on the BWW models. In this thesis, these analyses have been adjusted for the LPML serving as a starting point for the evaluation of the ontological completeness.

As previously mentioned, the BWW constructs are part of the reference set used to evaluate the LPML completeness and expressiveness. Table 55 presents the mapping of a core set of the BWW representation model constructs (according to (J. Recker et al., 2008)) to the LPML constructs reflecting the first two steps of the evaluation proceeding.

Table 55: Mapping the BWW representation model to the LPML

BWW Ontological construct	LPML construct
Thing (as well composite thing, component thing)	Not explicitly represented by the LPML
Property	Represented through attributes and semantic annotations of Conversation
Class	Represented through Conversation
Subclass	Not explicitly represented by the LPML
Kind	Not explicitly represented by the LPML
State	Not explicitly represented by the LPML
Conceivable state and state space	
State law	
Lawful state space	
Event	Represented through Activities
Conceivable event space	Not explicitly represented by the LPML
Lawful event space	Not explicitly represented by the LPML
Transformation	Represented through Activities. Could be represented through a pattern or template as well.
Lawful transformation	Represented through preconditions and postconditions, conditions in Gateways, Connectors and Flows
History	Not explicitly represented by the LPML
Acts-on	Not explicitly represented by the LPML
Coupling	Represented through Flow and Connector
System	Represented through Process
System composition	Represented through Process and Conversation
System environment	Represented through the context file referenced by each process element. However a clear distinction of external and intrinsic

	entities is not provided.
System structure	Not explicitly represented by the LPML
Subsystem	Represented through Process describing patterns, and templates
System decomposition	Not explicitly represented by the LPML
Level structure	Not explicitly represented by the LPML
Stable state	Not explicitly represented by the LPML
Unstable state	Not explicitly represented by the LPML
External event	Entry point in Process
Internal event	Not explicitly represented by the LPML
Well-defined event	Entry point in Process
Poorly-defined event	Not explicitly represented by the LPML

Having conducted the second step of the evaluation proceeding, the result reveals that 20 out of the 31 core BWW constructs do not have a direct LPML representation.

A remarkable gap is the lack of a representation of *thing* and its *kind*. Thing represents objects of the real world, kind the sort of thing. However, the lack of thing is common amongst orchestration languages of electronic services, such as BPEL. A couple of discussions exist stating the lack of thing may cause a lack of clarity in describing stakeholders of the process or in relating class instances (Peter Green et al., 2007). However, the semantic annotations in the LPML provide a means to replace descriptions of the construct thing. For example, a description of participants might be defined through semantic annotations for process or activity stakeholder, such as author, service provider, or process owner.

Another gap is *state* and its related constructs. Hence, the specification of business rules is not possible with the LPML. However, with respect to our target user group the specification of business rules would require too much skills or training effort. In the LPML, states might be simulated by preconditions and postconditions in combination with environment variables. Furthermore, constructs related to event spaces are not supported by the LPML. For reasons of abstraction and late service binding, the event spaces cannot be predefined. History-related constructs are missing as well which could lead to issues in recovery and reliability. However, the LPMS performs tasks related to history, hence, a tracking through an LPML construct is not necessary. The support for *system structure* is not a purpose of a process modelling language.

Also notable is the lack of a representation of *system environment*. Green et al. (Peter Green et al., 2007) argue this might lead users to lack a clear distinction of things inside and outside the system. This lack of a clear identification of external things might lead to a difficult identification of entities that might generate external events.

Transformation might be represented by a single activity or a set of activities and gateways resp. connectors. The following section about pattern-based analysis further details this aspect.

In the third step of the evaluation proceeding, these findings have been analysed in the context of the public sector processes. In various expert interviews, the lack of LPML constructs to cover the BWW constructs has been discussed. The feedback indicated that the lack of these constructs is not crucial for the regarded process models.

7.3.3.2 Pattern-based Analysis of the LPML

In order to express patterns and templates of both control and data flow, the BWW construct *transformation* is used. This proceeding has as well been partially described in Schnabel et al. (Schnabel, Born et al., 2009). Through the LPML, transformation might be represented by a set of activities and gateways, respectively connectors, the patterns, and templates. In this section is described which patterns and templates the LPML supports. A similar approach has already been performed for BPMN and EPCs as described by Russel et al. and Recker and Dreiling (J. Recker & Dreiling, 2007; N. Russell et al., 2006). Like in the previous section, the evaluation proceeding comprises three steps. The first step is the selection of a benchmark set of 20 control flow patterns based on workflow systems as described in van der Aalst et al. (van der Aalst, ter Hofstede, Kiepuszewski et al., 2003) and a set of six communication patterns based on Enterprise Application Integration (EAI) systems (Ruh et al., 2001). Afterwards, an analysis is performed to figure out which patterns of the two benchmark sets the LPML covers. This approach is similar to the one proposed by Wohed et al. (Wohed, van der Aalst, Dumas, & ter Hofstede, 2003).

Table 56 presents the results of the LPML coverage of the control-flow patterns based on workflow systems, Table 57 the coverage of control-flow patterns found in EAI systems.

Table 56: Control-flow patterns based on workflow systems

Pattern	**Description**	**Implementation using LPML**
Sequence	A cannot start before B completes	Flow
Parallel Split , Synchronization, Synchronizing Merge	Here, concurrent execution is enabled. In case of synchronization C can only start once all the active Ai from $(A_1..A_n)$ complete.	Parallel gateway
Exclusive Choice	At this point, one of $(A_1..A_n)$ is chosen based on data	Exclusive gateway
Simple Merge	C can only start once A or B completes, only A or B can be run	Exclusive gateway
Multi Choice and Multi Merge	At this point, two or more of $(A_1..A_n)$ are chosen based on data resp. C is started once for each completion of active A_i from $(A_1..A_n)$	Parallel gateway in combination with exclusive gateway, both with conditional expressions, in LPML semantic annotation.
Discriminator	C is started just once with the first completion from all active A_i $(i\in \{1..n\})$	Parallel gateway with conditional expressions, in LPML semantic annotation
Arbitrary Cycles	Any portion of the process should be visited repeatedly	Will potentially be supported in an extended version of the LPML
Implicit Termination	Process completes when nothing left to do, without explicit term. activity	Implicit termination is not supported, explicit "End activity" used instead
MI without Synchronization	A number of concurrent (sub)process instances are created	Not supported by default. In case the support is needed, the LPML might be extended.
MI with a Priori Design Time Knowledge	A number of concurrent (sub)process instances are created and their completion synchronised, before proceeding with the rest of the process.	
MI with/without a Priori Runtime Knowledge		
Deferred Choice	Point of choosing A or B is reached before the decision	Parallel gateway with conditional expressions, in

	data is available.	LPML semantic annotation
Interleaved Parallel Routing	Each A_i from (A_1..A_n) is executed exactly once in an order determined just after the previous activity	Might be modelled through preconditions
Milestone	C can only be started if A has finished but a subsequent B has not yet started	Might be modelled through preconditions
Cancel Activity	Terminate activity	Not supported
Cancel Case	Terminate instance	Not supported

Table 57: Control-flow patterns based on EAI systems

Pattern	Description	Implementation using LPML
Request/Reply	Sender waits for a reply before continuing	Depends on the transformation of the LPML in an executable language. In the LPML, the activity handles this pattern.
OneWay	Sender waits for an acknowledgment before continuing	
Synchronous Polling	Sender polls for a response whilst receiving one.	
Message Passing	Sender sends a message and continues processing	Depends on the service executing the activity. However, this is not explicitly modelled.
Publish/Subscribe	Request sent to all receivers which have previously declared interest	Might be modelled implicitly by data split or control flow split.
Broadcast	Request sent to all receivers in a network, each decides whether to act	Not supported

The results show that most of the patterns can be supported through the LPML. However, the implementation of the patterns in LPML process models might be difficult and complicated through sophisticated conditional expressions in semantic annotations. With respect to the target user, the application of most of these complicated patterns is not expected. A business user normally doesn't have the skills to model those patterns. A significant simplification and user support to model these patterns is not followed regarding the assumed need in very few cases. The feedback given by the experts from the public sector use case supported these arguments.

7.3.4 Adaptability and Extensibility

The adaptability and extensibility of the LPML is analysed statically according to the design science. In general, the semantic annotations allow for adjusting and extending the LPML. In the standard version, semantic annotations are predefined for discovering and selecting services. These annotations are hence mostly addressed towards service descriptions. However, the semantic annotation type *metadata* allows for the definition of heterogeneous descriptions and information. An example of an extension has been given in the public sector use case by an annotation referencing the user management file. Table 58 discusses the LPML adaptability and extensibility facing the public sector extensions. The results show that the LPMS might be easily customized to the public sector and hence, very likely to other scenarios as well.

Table 58: LPM language and tool evaluation according to public sector requirements

Evaluation Criteria	**LPM Evaluation**
User Management	Through separating the user model from the process model and only providing a reference in the process model, flexibility is ensured. Any change in the user model doesn't impact the process model. Further, the process model might be used in various process contexts.
Approval	A special approval handling has been implemented in the LPMS. Further, through the simple definition of a semantic annotation for approval, this requirement might be handled easily by the LPMS.
Versioning	An easy change management and tracking is available through versioning and logging of the editing history. By simply customizing semantic annotations for versioning, this requirement is easily handled by the LPMS.
Human task management	Through the separation of the human task management from the process model, standard execution engines might be used. This keeps the LPML compatible to other existing process modelling languages. By providing service interfaces for the LPMS, the integration of a human task server is easily feasible.

7.3.5 Usability

The usability evaluation of the LPMS including the LPML is strongly aligned with the simplicity and understandability evaluation. This section covers all of them. First, a heuristic evaluation is performed based on literature criteria addressing a static evaluation according to the design science. Afterwards, an observational analysis is described in terms of a user survey and workshops focussing on the evaluation of the design of the LPMS. The usability of the prototype of the LPMS is not part of this thesis and will be conducted in the context of the SOA4All project.

The heuristic usability evaluation is based on the work performed by Nielsen (Nielsen, 1993) who describes heuristics for usability engineering focusing on user interfaces. In this thesis, these usability heuristics are applied to the LPMS. Table 59 describes the static analysis of these heuristics.

Table 59: LPM evaluation based on usability heuristics (Nielsen, 1993)

Usability heuristic	LPM evaluation
Visibility of system status	The visibility is implemented in the navigation scheme of the process editor. Further, the proceeding steps of the background tools (composition component, execution engine) might be monitored through the process editor.
Match between system and the real world	Direct match through semantic annotations rather than service specification
User control and freedom	Undo and redo functionality in the process editor
Consistency and standards	Ontology-based terms guarantee a common language.
Error prevention	Design time support for the user to produce sound process models. Wizards guide the user. A validation functionality is implemented in the process editor.
Recognition rather than recall	Separate data area indicates process-related data.
Flexibility and efficiency of use	Various user modes: Users without experience are guided through wizards, experienced users might model directly.
Aesthetic and minimalist design	Data sections to be opened and closed avoid the display of unnecessary information.
Help users recognize, diagnose, and recover from errors	Wizards guiding the user in case of errors

Help and documentation	Help section foreseen in a commercial version of the LPMS.

The observational analysis of these heuristics is performed by a survey and workshops that are described in the following. The survey showed mock-ups and snapshots of the current version of the process editor and the LPML and asked according questions. According to Nielsen (Nielsen, 1990), such a heuristic evaluation is performed at an early stage of the tool implementation as it is in the context of this thesis.

Normally, a heuristic evaluation produces a list of identified issues, for example usability problems in the user interface of the process editor. These identified issues are assigned to the well-known usability problems as described by Nielsen (Nielsen, 1993). Through a systematic structure, the usability issues might often easily be solved. For the evaluation of the LPMS, a similar proceeding is proposed. The usability strongly depends on the stability of the supporting modelling tools. As soon as the integrated prototype of the modelling tools exists, a usability testing workshop should be performed.

Focus group evaluation

In terms of the observational evaluation, the focus group described in section 7.2 has been surveyed. The survey revealed that the LPM approach and design process are easily understandable and highly usable. Further, the focus group agreed that the graphical abstraction and the LPML symbols are very understandable.

Survey questions and results

The survey covered questions about the usability and simplicity of the process editor and the understandability and simplicity of the graphical representation of the LPML. The survey respondents had been shown a video and a set of slides introducing the process editor and the LPML. Since this thesis describes the design of the LPMS, a video and a set of slides are appropriate means to evaluate the LPMS. Further, the survey questions targeted the evaluation of the LPM design principles. In the following, the survey questions and the corresponding results are described. Each presented value is based on the 5-point Likert scale (see Table 46).

- The survey question about whether the process editor is easily usable from a business user perspective revealed an average value of 3.52. Similarly, the question about whether the graphical process representation is easily understandable revealed an average value of 4.00. These two results show a high acceptance of the design of the process editor.

- The average value of whether people are capable of opening and adjusting predefined processes is 3.86. This means that people might easily understand existing processes which, again, reveals a high usability of the process editor and the LPML.
- The average value of people that like the graphical layout of the modelling environment is 3.74. This shows as well the high usability of the LPML.
- However, the average value of people preferring to have an introduction (video, user manual) to the LPMS is 3.71. Hence, the LPMS should be introduced with the target to keep the needed introduction time short.

In the following, the results for the evaluation of the LPM design principles are presented.

Abstraction concept

In terms of the abstraction concept, the survey showed that the LPML uses clear graphical symbols with an average value of 3.65 and that users prefer a graphical model to a textual model with an average value of 3.91. Further, the average value of people missing important information in the graphical model is only 2.78. These results verify the decision to use a graphical representation for the users to model their processes.

Semantic annotations

Concerning semantic annotations, the average value of people preferring to have additional descriptions indicating the service function is 3.57. This reveals that the mere service function signature is not sufficient for people to understand the service functionality. People prefer not to have to specify the data type for input and output data which is shown by the average value of 3.61. These two survey results show in general that the targeted users of the LPMS have difficulties in understanding the technical descriptions and prefer to have additional annotations in a business language. Further, the average value of people preferring to specify non-functional properties is 3.61. This shows that services are not only selected by their functionality but as well by non-functional properties, such as price or availability. Lastly, people prefer to get provided metadata, such as author or history data, which is shown by an average value of 3.61.

Goals

Concerning goals, the average value of people preferring to specify a service category rather than a service is 3.70. As had been shown by the semantic annotations, here again, the result shows that people prefer to be kept free from execution details. However, the average value of people trusting that search tools will find the best fitting service is only 2.87. A clear distinction between technical and business people came out for this question. The corresponding value for business users is 3.20. Thus, the business users as the main target group of the LPMS expect to benefit from the automatic service selection. Same as for the previous question, the average value of people preferring to select a concrete service themselves is 3.74 and 3.40 for business users. In the context of the overall survey results, these values seem too high and not in line with the other results of the goal design principle evaluation. An interpretation might be that the respondents of the survey thought about the service functionality independent of the representation through a service category or a concrete service. Finally, respondents don't prefer to specify detailed information about service binding and execution which is shown by an average value of 2.65 and an average value of 2.40 for business users.

Context information

Concerning context information, the average value of people that won't allow a tool to use personal profile information for privacy reasons is 2.91 and don't want to publish business information such as the industry, company, or department, is as well 2.91. This shows that the majority of potential users of the LPMS expect to benefit from context information. The average value of respondents trusting the software to observe the privacy policy is 3.09. However, the average value of respondents who don't want a tool to use information about their personal history is 3.17. All these results show that privacy has to be respected when using personal information for contextual information.

Patterns and templates

Concerning patterns and templates, the average value of people trusting other users to model processes and parts that might be reused is 3.78. This shows that the survey respondents are highly interested in reusing processes and parts of it. As well, the average value of people thinking that it's useful to search for existing processes and parts to be reused before starting modelling is 4.13. Another aspect of reuse is the provisioning of existing processes and parts as examples. The survey showed that most of the respondents prefer to check existing process models serving as examples (average value of 4.17). Further, the average value of people thinking that processes are completely different and that hence, there would be no use of searching for existing process models is only 2.13. Respondents think that process patterns and templates have to be easily retrievable (average value of 4.09). Through the semantic annotations and the corresponding, various search mechanisms, the LPMS respects this result. Moreover, the respondents rely on process models and parts of them that are in line with well-defined quality standards, but reuse as well models with unknown quality. The according average value is 3.48. However, the respondents don't want to spend a lot of time to understand existing process models (the average value is 4.00). Since the other survey results revealed a high usability and simplicity of both the LPMS and the LPML, the needed time to understand existing process models should be low.

Data connectors

In terms of data connectors, people desire the LPMS to handle lists of data entries. The according average value is 3.77. The average value of respondents preferring to define data manipulation activities is 3.43. This shows that the respondents prefer to have explicit activities allowing for the handling of complex data flow operations. As well, the respondents prefer to use predefined data operators. The according average value is 3.78. Lastly, most of the users prefer not to specify a data mapping (average value of 3.39).

Gateways

The average value of respondents preferring to have no explicit gateways is 3.23 and preferring to draw multiple outgoing or incoming connections for an activity is 3.26. This legitimates the decision not to explicitly represent gateways in the graphical abstraction of the LPML.

Workshops: Experiments to check simplicity and understandability

An experiment to be conducted concerns the comparison of the LPML to other process modelling languages, such as BPMN. A BPMN and LPML process model comprising the same content could be modelled either by the same user to directly compare the two languages or by different users in order to measure usability and simplicity.

An additional experiment should be to let two groups with different modelling expertise model the BPMN process and two groups model the LPML process. By doing so, it might be figured out whether differences in the usability of the two languages exist in user groups with low or high experience.

In a further experiment comparing LPML to other modelling languages, the users are asked to model process splits or merges as a typical workflow process pattern according to van der Aalst (van der Aalst, ter Hofstede, Kiepuszewski et al., 2003). By measuring the time to model the process as well as potential documentation consultations, the usability and simplicity of the LPML might be evaluated.

Furthermore, by comparing the LPML to other modelling languages, the users might be asked about their intuitive understanding of the two process models.

7.4 RISKS

Besides the presented enablers of BPM for business users, a couple of risks exist that might lead to a missing acceptance amongst potential users.

Users are afraid of losing control over information, both personal and business information. For example, in the area of social interaction, the users are afraid of friends or service providers publishing or forwarding confidential information to people or organizations that do not stick to data-protecting principles. Further, a potential risk is the availability of services when they are needed. Service Level Agreements (SLA) could be difficult to negotiate in an open environment, such as the web.

The SOA4All workshop described in Namoune et al. (Abdallah Namoune et al., 2009) showed in addition that the LPMS should foster the awareness of consequences of one's actions. This had been felt to be the most significant difference between non-experienced and experienced users. In addition, complete automation of the modelling proceeding could frustrate users due to missing control options. Hence, the LPMS should indicate the automated steps at any point in time if this is requested by the user. And finally, the users feared a lack of clarity in using context information, personalizing the system, or reusing modelled processes and parts of it.

In terms of reuse of process models, a couple of additional risks might emerge. Denning et al. (Denning, Horning, Parnas, & Weinstein, 2005; Kittur, Suh, & Chi, 2008) described in their work potential risks for collaboration platforms, such as Wikipedia. This thesis is evaluated with respect to these risks. According to the authors, an issue might emerge by not knowing which of the process models are accurate. Further, the modelling experience of the authors might not be known which leads to difficulties in assessing the process quality. However, before reusing process models the soundness should be guaranteed.

Another issue might occur in terms of process model stability. By enabling business users to modify, adjust, and update process models, versioning control could be difficult to implement. Hence, the models could be instable. As described in the public sector use case in section 1, versioning should be implemented for process models. This is as well important for the process owners in order to keep control of their process models.

7.5 CONCLUSION

This section presented the evaluation of the design of the LPMS. First, the evaluation metrics have been specified. These metrics are based on the requirements defined in section 3.2 and are grouped into technical, individual, organisational, and economic metrics. The type of evaluation is differentiated according to their basement on general requirements and literature and on the use case.

Afterwards, the target users have been defined that are beneficiaries of the LPMS. The users have been distinguished into IT experts, business users, and casual business users. The evaluation has been performed with respect to these target users.

Besides the static evaluation, mainly two steps have been followed to perform the observational evaluation of this thesis: the focus group evaluation and a comprehensive expert and business user evaluation. While the focus group evaluation is targeting business users, the expert and business user evaluation targets all kind of users. For the expert and business user evaluation, a survey has been conducted. In addition, workshop experiments have been described that eventually evaluate an implementation of the LPMS. The main finding of the focus group evaluation has been that business user empowerment in the BPM area is useful for organisations and that the selected design principles for LPM seem to be capable to fulfil the needs of business users.

The survey has been based on a couple of hypothesis giving statements about the applicability of the LPMS to various, heterogeneous business areas and about the users' preferences to model and execute processes.

The results of the static evaluation revealed that all requirements are satisfied. The survey results revealed that the respondents are mostly satisfied with the design of the LPMS. A couple of aspects have to be explained in more detail.

A low value (2.80) resulted in the usage for business tasks by the respondents with economic background, compared to an overall value of 3.45. However, the usage in a non-business context is expected higher (value of 3.20), compared to the overall value of 3.07. An interpretation might be that business users don't expect needs to execute processes in their daily business. Another low value (2.60) resulted in the potential sharing of services and processes for respondents with economic background, compared to an overall value of 3.28. Each presented value is based on the 5-point Likert scale (see Table 46).

As well, the results revealed that values about publishing information about the user profile, business information, or tool usage history are lower for respondents with an economic background than the overall values. While in IT-related areas the community-driven approaches seem to be wide-spread, the business areas are not yet convinced about sharing information.

The general risks seen by the users confirm these feelings that came out in the survey. The users fear to lose control over important information and not to be able to rely on correct and stable information.

The further evaluation tasks concern the conduction of the described workshops and experiments. As soon as an implementation of the LPMS will exist, in particular, the usability has to be evaluated through modelling experiments. Another interesting evaluation is the comparison of the LPML to other process modelling languages, such as BPMN, YAWL, or BPEL. Further experiments have to be conducted comparing the new LPML to those existing languages.

8 CONCLUSION AND OUTLOOK

This thesis gave an insight into the specification of Lightweight Process Modelling (LPM). The according implementation of a language and tools for LPM has been named LPMS. Mainly three components for LPM have been designed. The set of LPM design principles, structured by the LPM metamodel, forms the first component. The other two components are part of the design of the LPMS and comprise the LPML and the set of according tools.

In the context of the design science, this section covers the phase of "*learn and theorize*" according to Rossi and Sein (Rossi & Sein, 2003). This phase comprises the reflection on the artefact, the generalisation of the findings, and the confirmation or rejection of the original assumptions. In addition, in this thesis, the research contribution and the future work based on the research results is presented in this section.

8.1 REFLECTION ON THE ARTEFACT

LPM promises to enable business users to model and execute processes. This is a challenging goal that is not completely new to research but that hasn't been investigated yet in a holistic approach. The evaluation of this thesis revealed that all requirements have been fulfilled by the LPMS.

However, the critical point of a new language and new tools is user acceptance. The usefulness and usability of the new features has to be proved in order to achieve user acceptance. This proof has to be performed as soon as the integrated implementation of the LPMS design has been completely realized. New requirements might arise that are not yet reflected by the LPMS.

The user capability to provide semantic annotations is a key to make processes executable for the LPMS. For sure, the provisioning of semantic annotations is easier than providing execution information. However, probably training effort is needed for the user in order to be able to provide suitable service descriptions in terms of semantic annotations.

Within a research environment, the benefit of the reuse design principle cannot be completely evaluated. In order to provide full benefit of the user support through patterns, templates, and goals, a critical mass of existing artefacts has to be provided.

In particular, domain-specific patterns, templates and goals that have to be created in practice will provide an added value to the user.

Another aspect to reflect on is the evaluation of the LPML. The LPML has been evaluated using the concepts of ontological completeness and coverage. An analysis with respect to the target usage revealed satisfactory coverage of the exhaustive sets of concepts and patterns used for evaluation. The most important differences concern the lack of representation for environment and things interacting with the process. This seems to be common with other execution-oriented process modelling languages, such as BPEL. The LPML is based on a lightweight nature. This means the language aims at being fit for purpose rather than being complete. This explains that such missing concepts have been dismissed in the LPML.

In order to support the use of the LPMS, existing examples of successfully created and executed process models might be provided. This helps the user in understanding the modelling steps. Further, often, people are less afraid after having seen other users successfully using a tool. A community-based feedback and recommendation tool supports this as well.

8.2 GENERALISATION OF FINDINGS

In this thesis, the LPMS design has been applied to only one industry. For this industry, the public sector, the LPMS revealed the capability, to overcome current issues and to add value to organisations and users of the public sector. Future work should apply the LPMS to further industries. Hereby, the LPMS shows high potential in adding value to further industries. The target group of the LPMS is selected according to the users' IT skills. However, the users' IT skills do not depend significantly on the industry they work in. Further, the LPMS doesn't restrict the type of services or processes. Any kind of process integrating any kind of service is allowed by the LPMS. Lastly, as shown by the public sector use case, the LPMS might be extended according to special user or industry requirements.

These theoretical arguments are confirmed by the project SOA4All where the LPMS is applied to one scenario in the telecommunications industry and to one targeting small and medium-sized enterprises.

8.3 Evaluation of Original Assumptions

One of the challenges for the success of the LPMS is the existence of semantically described services. Currently, most of the existing services are only described syntactically. The research project SOA4All has recognised the need for a tool allowing the easy creation of semantic service descriptions. In the context of the project, SWEET[26] (Semantic Web sErvice Editing Tool) has been developed. SWEET is "*an editor for supporting the semantic annotation of Web APIs and RESTful services*."

The ability to be transformed into other process modelling languages, both executable and documentation languages, is key to the LPML. In this thesis, a transformation of the LPML into BPEL has been assumed. However, the LPML is designed to be flexible enough in order to be easily transformed into various, existing process modelling languages.

Further, for this thesis has been assumed that users prefer graphical modelling languages to text-based languages. This has been confirmed by the survey results described in the evaluation section. However, cases might appear where the information provided by the graphics is not sufficient. Additional textual representations might be needed.

The application of ontologies in process modelling languages has to be further evaluated in the future. For the LPML, a pragmatic approach has been selected. Hereby, ontologies are only applied for specifying activities. At the moment, this approach seems to be most appropriate in order to achieve a working solution that might be handled by business users. However, as semantic descriptions will evolve and be attached to more and more artefacts, other full-fledged ontological approaches, such as BPMO-based approaches, might be applicable as well.

[26] See http://sweet.kmi.open.ac.uk/

8.4 Research Contribution

This thesis contributes to the research community through Design Research. Design Research is the process of constructing and evaluating an artefact. The research results provided by this thesis are principles for lightweight process modelling and the LPMS comprising the LPML, a design process to make processes executable, and the tools for execution.

This thesis has been created in the context of the EU-funded research project SOA4All. Parts of the thesis had been published in deliverables of SOA4All which reveals the strong research contribution.

8.5 Future Work

This final section covers potential application scenarios of the LPMS that might be elaborated in the future. By enabling business users to model and execute processes, the amount of well-structured processes will increase. This could be used to leverage bottom-up approaches, e.g. to derive an organisational structure as described in section 8.5.1 or to enable a collaborative process development as described in section 8.5.2.

8.5.1 Organizational Structure

Currently, in most organisations that envisage to systematically model their services and processes an expert modelling team is requested. These modelling experts request business process and task information from process owners. The process owners are typical business users whereas the modelling experts have expertise in process modelling and IT yet no specific business knowledge. In order to gather information about the processes and tasks, the modelling experts have to closely collaborate with the process owner. The modelling experts have to ask for the processes, tasks, and their structural organisation. Based on this information, they create process and task models. Since often the gathered information is not enough, the modelling experts have to conduct multiple iterations of questioning.

According to the experiences of Lanker (Lanker, 2008), the current information gathering procedure might be ineffective and inefficient. Quite often, discussions about processes and services arise that are not goal-oriented but time consuming. The achieved results seem obvious which makes it difficult for the expert team to ask for peoples' time. Especially modelling workshops are rather time-consuming. Further, the processes and their lifecycle have multiple dimensions that are not easy to be transferred from the business expert to the modelling expert. Thus, there's often a loss of information in discussing. In addition, an iterative proceeding is required capturing a lot of resources. And finally, there is not the “one and only” solution. A couple of sufficient solutions and loads of bad solutions exist.

In order to avoid time-consuming discussions and the loss of information between business experts and IT experts, this thesis envisages enabling the process owners to model their processes themselves.

By using the LPMS, a process owner is enabled to model its process. Therefore, the process models might be enriched and used in order to derive an organisational structure. This organisational structure comprises the documentation of the processes (what), the stakeholders (for whom), the services (for what), the process context (depending from), and the process steps (how). This documentation will have to reveal the right granularity in various views and align the models, the terminology, and the form of representation. An existing business design according to High et al. (High et al., 2005) is assumed. However, the business design is not well documented.

The organisational structure comprises processes, services, process groups, service groups, and the relations between them. A prerequisite thus is that the processes are identified, modelled, and accordingly described by a semantic annotation.

In order to build the organisational structure, additional information is needed that is only partly in the process models. By using the semantic annotations, the services and processes might be aggregated into process and service groups. This is performed by providing process and service categories. According to the context-awareness principle, the linkage to a business entity might be used to optimize the assignment to existing groups. Process owners, responsibles, and constituents might be assigned according to the information provided by the process metadata.

By using the previously gathered information, the organisational structure might be built. Therefore, the individual process and task models and process and service groups are combined.

8.5.2 Processpedia

By providing an easy methodology, language, and tools for process modelling, the amount of well-modelled and automatically executable processes will increase. As well, a significant higher group of information worker will gain knowledge about process modelling. In order to further spread and share this kind of knowledge, a collaboration platform is needed where users might publish their process models. Further, existing BPM suites have only limited functionality for discovering processes

(Richardson et al., 2009). Similar to Wikipedia[27] as a community-driven online dictionary a platform to model, share, search, and discuss process models, the so called *Processpedia*, might be created. The term Processpedia has been announced by the SAP Research Team in the CEC St. Gallen. Through various feedbacks within the BPM community, Processpedia will further improve the quality and foster the reuse of process models. A BPM community might be built within organisations as well as across organisations in networks. Concretely spoken, the LPMS might be extended to support sharing, searching, reusing, and discussing process models. This comprises new functionalities for storage and easy retrieval of processes as well as a suitable security mechanism. Further, mechanisms to ensure trust are important. Such mechanisms might comprise public user profiles, the assignment of process models to profiles, access restrictions, different views on process models, or documentation of the editing history.

Additional functionalities for the collaboration might comprise monitoring, notification mechanisms, and functionalities for marks, commenting, recommendations, ratings, votes, or file references.

[27] See www.wikipedia.org

9 ANNEX

9.1 PUBLIC SECTOR PROCESS MODELS AND SERVICES

9.1.1 LPML model of the process “Registration of a business”

This section shows a simplified version of the canonical format of the abstract LPML model “Registration of a business” in Java-Notation. The representation below is the textual equivalent to the graphical version as shown in section 6.2. The process model has been created by the author of this thesis in cooperation with the SAP Research SOA4All team. At the beginning of the file a couple of classes have to be imported that aren’t depicted for visibility reasons. The process model is implemented as Java Class.

```
// The imported classes have been deleted for visibility
reasons
public class BusinessRegistrationTest {
private static final String targetNamespaceURIPrefix =
"http://org.soa4all.eu/wp7#";
private Process process = null;
private Process deserializedProcess = null;
     }
public Process createProcessFixtureForWP7Scenario() {

//////////////////////////////////////////
// creation of a new process
//////////////////////////////////////////

Process process = new ProcessImpl();

SemanticAnnotation sa = new SemanticAnnotationImpl();
sa.setReferenceURI(targetNamespaceURIPrefix +
"transactional");
sa.setType(AnnotationType.NON_FUNCTIONAL_PROPERTY);
process.addSemanticAnnotation(sa);

// creation of Start/End process element
Activity start = process.setStartElement();
Activity end = process.setEndElement();

// ---- Find Citizen in CRM: Activity bound to a
conversation
Conversation findCitizenInCRMConversation = new
ConversationImpl();
```

```
Goal findCitizenInCRMGoal = new GoalImpl();
Service findCitizenInCRMService = new ServiceImpl();
findCitizenInCRMService.setServiceReference();
findCitizenInCRMConversation.addService(findCitizenInCRMSe
rvice);
Activity findCitizenInCRM = new ActivityImpl();
findCitizenInCRM.setName("Find Citizen in CRM");
findCitizenInCRM.setOperation();
findCitizenInCRM.setStartElement(false);
findCitizenInCRM.setEndElement(false);
findCitizenInCRM.setSynchronous(true);
findCitizenInCRM.setConversation(findCitizenInCRMConversat
ion);
findCitizenInCRM.setHumanTask(false);
process.addProcessElement(findCitizenInCRM);

// Input/Output
Parameter citizenInCRMforkParameter = new ParameterImpl();
SemanticAnnotation citizenInCRMforkConditionAnnotation =
new SemanticAnnotationImpl();
citizenInCRMforkConditionAnnotation.setReferenceURI(target
NamespaceURIPrefix + "citizenInCRMforkParameter");
citizenInCRMforkConditionAnnotation.setType(AnnotationType
.META_DATA);
citizenInCRMforkParameter.addSemanticAnnotation(citizenInC
RMforkConditionAnnotation);
findCitizenInCRM.addOutputParameter(citizenInCRMforkParame
ter);

// Gateway
// Citizen in CRM exclusive fork gateway
ExclusiveGateway citizenInCRMforkGateway = new
ExclusiveGatewayImpl();
citizenInCRMforkGateway.setSplit(true);
citizenInCRMforkGateway.setCondition(citizenInCRMforkCondi
tionAnnotation);
process.addProcessElement(citizenInCRMforkGateway);

// Citizen in CRM exclusive merge gateway
ExclusiveGateway citizenInCRMMerge = new
ExclusiveGatewayImpl();
citizenInCRMMerge.setSplit(false);
process.addProcessElement(citizenInCRMMerge);

// ---- Create Citizen in CRM: Activity bound to a
conversation
Conversation createCitizenInCRMConversation = new
ConversationImpl();
```

```
Service createCitizenInCRMService = new ServiceImpl();
createCitizenInCRMService.setServiceReference();
createCitizenInCRMConversation.addService(createCitizenInC
RMService);
Activity createCitizenInCRM = new ActivityImpl();
createCitizenInCRM.setName("Create Citizen in CRM");
createCitizenInCRM.setOperation();
createCitizenInCRM.setSynchronous(true); // blocking
operation (default)
createCitizenInCRM.setConversation(createCitizenInCRMConve
rsation);
process.addProcessElement(createCitizenInCRM);

// Input/Output
createCitizenInCRM.addOutputParameter(citizenInCRMforkPara
meter);

// ---- Read bank details of citizen: Activity bound to a
conversation
Conversation readBankDetailsOfCitizenConversation = new
ConversationImpl();
Service readBankDetailsOfCitizenService = new
ServiceImpl();
readBankDetailsOfCitizenService.
setServiceReference();
readBankDetailsOfCitizenConversation.addService(readBankDe
tailsOfCitizenService);

Activity readBankDetailsOfCitizen = new ActivityImpl();
readBankDetailsOfCitizen.setName("Read Bank Details Of
Citizen");
readBankDetailsOfCitizen.setOperation();
readBankDetailsOfCitizen.setSynchronous(true); // blocking
operation (default)
readBankDetailsOfCitizen.setConversation(readBankDetailsOf
CitizenConversation);
process.addProcessElement(readBankDetailsOfCitizen);

// Input/Output
readBankDetailsOfCitizen.addInputParameter(citizenInCRMfor
kParameter);
Parameter bankDetailsOfCitizenForkParameter = new
ParameterImpl();
SemanticAnnotation
bankDetailsOfCitizenForkAnnotationCondition = new
SemanticAnnotationImpl();
```

```
bankDetailsOfCitizenForkAnnotationCondition.setReferenceUR
I(targetNamespaceURIPrefix +
"bankDetailsOfCitizenForkParameter");
bankDetailsOfCitizenForkAnnotationCondition.setType(Annota
tionType.META_DATA);
bankDetailsOfCitizenForkParameter.addSemanticAnnotation(ba
nkDetailsOfCitizenForkAnnotationCondition);
readBankDetailsOfCitizen.addOutputParameter(bankDetailsOfC
itizenForkParameter);

// CitizenBankDetailsAvailable exclusive fork gateway
ExclusiveGateway citizenBankDetailsAvailableFork = new
ExclusiveGatewayImpl();
citizenBankDetailsAvailableFork.setSplit(true);
citizenBankDetailsAvailableFork.setCondition(bankDetailsOf
CitizenForkAnnotationCondition);
process.addProcessElement(citizenBankDetailsAvailableFork)
;

// CitizenBankDetailsAvailable exclusive merge gateway
ExclusiveGateway citizenBankDetailsAvailableMerge = new
ExclusiveGatewayImpl();
citizenBankDetailsAvailableMerge.setSplit(false);
process.addProcessElement(citizenBankDetailsAvailableMerge
);

// ---- Create bank details for citizen: Activity bound to
a
// conversation
Conversation createBankDetailsForCitizenConversation = new
ConversationImpl();
Service createBankDetailsForCitizenService = new
ServiceImpl();
createBankDetailsForCitizenService.setServiceReference();
createBankDetailsForCitizenConversation.addService(createB
ankDetailsForCitizenService);

Activity createBankDetailsForCitizen = new ActivityImpl();
createBankDetailsForCitizen.setName("Create Bank Details
For Citizen");
createBankDetailsForCitizen.setOperation();
createBankDetailsForCitizen.setSynchronous(true); //
blocking operation (default)
createBankDetailsForCitizen.setConversation(readBankDetail
sOfCitizenConversation);
process.addProcessElement(createBankDetailsForCitizen);

// Input/Output
```

```
createBankDetailsForCitizen.addInputParameter(citizenInCRM
forkParameter);
createBankDetailsForCitizen.addOutputParameter(bankDetails
OfCitizenForkParameter);

// ---- Check location: Activity performed by human
// Binding of human tasks not further specified in this
thesis

// Service checkLocationService = new ServiceImpl();
// checkLocationService.setServiceReference();
Activity checkLocation = new ActivityImpl();
checkLocation.setName("Check Location");
checkLocation.setHumanTask(true);
process.addProcessElement(checkLocation);

// ---- Check lawfulness: Activity performed by human
// Binding of human tasks not further specified in this
thesis

// Service checkLawfulnessService = new ServiceImpl();
// checkLawfulnessService.setServiceReference();
Activity checkLawfulness = new ActivityImpl();
checkLawfulness.setName("Check Lawfulness");
checkLawfulness.setHumanTask(true);
process.addProcessElement(checkLawfulness);

// Input/Output
Parameter preChecksParameter = new ParameterImpl();
SemanticAnnotation preChecksAnnotationCondition = new
SemanticAnnotationImpl();
preChecksAnnotationCondition.setReferenceURI(targetNamespa
ceURIPrefix + "preChecksParameter");
preChecksAnnotationCondition.setType(AnnotationType.META_D
ATA);
preChecksParameter.addSemanticAnnotation(preChecksAnnotati
onCondition);
checkLawfulness.addOutputParameter(preChecksParameter);

// Gateway
// Pre-Check exclusive fork gateway
ExclusiveGateway preCheckForkGateway = new
ExclusiveGatewayImpl();
preCheckForkGateway.setSplit(true);
preCheckForkGateway.setCondition(preChecksAnnotationCondit
ion);
process.addProcessElement(preCheckForkGateway);
```

```
// Pre-Check exclusive merge gateway
ExclusiveGateway preCheckMerge = new
ExclusiveGatewayImpl();
preCheckMerge.setSplit(false);
process.addProcessElement(preCheckMerge);

// ---- Send Denial: Activity bound to a conversation
Conversation sendDenialConversation = new
ConversationImpl();
Service sendDenialService = new ServiceImpl();
sendDenialService.setServiceReference();
sendDenialConversation.addService(sendDenialService);
Activity sendDenial = new ActivityImpl();
sendDenial.setName("Send Denial");
sendDenial.setOperation(); // Add operation
sendDenial.setSynchronous(true); // blocking operation
(default)
sendDenial.setConversation(sendDenialConversation);
process.addProcessElement(sendDenial);

// Input/Output
sendDenial.addInputParameter(citizenInCRMforkParameter);
sendDenial.addOutputParameter(citizenInCRMforkParameter);
sendDenial.addOutputParameter(preChecksParameter);

// ---- Check identity: Activity performed by human
// Binding of human tasks not further specified in this
thesis

// Service checkIdentityService = new ServiceImpl();
// checkIdentityService.setServiceReference();
Activity checkIdentity = new ActivityImpl();
checkIdentity.setName("Check Identity");
checkIdentity.setHumanTask(true);
process.addProcessElement(checkIdentity);

// ---- Check legal form: Activity performed by human
// Binding of human tasks not further specified in this
thesis

// Service checkLegalFormService = new ServiceImpl();
// checkLegalFormService.setServiceReference();
Activity checkLegalForm = new ActivityImpl();
checkLegalForm.setName("Check Legal Form");
checkLegalForm.setHumanTask(true);
process.addProcessElement(checkLegalForm);
```

```
// ---- Check operation allowance: Activity performed by
human
// Binding of human tasks not further specified in this
thesis

// Service checkOperationAllowanceService = new
ServiceImpl();
// checkOperationAllowanceService.setServiceReference();
Activity checkOperationAllowance = new ActivityImpl();
checkOperationAllowance.setName("Check Operation
Allowance");
checkOperationAllowance.setHumanTask(true);
process.addProcessElement(checkOperationAllowance);

// Input/Output
Parameter mainCheckParameter = new ParameterImpl();
SemanticAnnotation mainCheckAnnotationCondition = new
SemanticAnnotationImpl();
mainCheckAnnotationCondition.setReferenceURI(targetNamespa
ceURIPrefix + "mainCheckParameter");
mainCheckAnnotationCondition.setType(AnnotationType.META_D
ATA);
mainCheckParameter.addSemanticAnnotation(mainCheckAnnotati
onCondition);
checkOperationAllowance.addOutputParameter(mainCheckParame
ter);
sendDenial.addOutputParameter(mainCheckParameter);

// Gateway
// Main check exclusive fork gateway
ExclusiveGateway mainCheckForkGateway = new
ExclusiveGatewayImpl();
mainCheckForkGateway.setSplit(true);
mainCheckForkGateway.setCondition(mainCheckAnnotationCondi
tion);
process.addProcessElement(mainCheckForkGateway);

// Main Check exclusive merge gateway
ExclusiveGateway mainCheckMerge = new
ExclusiveGatewayImpl();
mainCheckMerge.setSplit(false);
process.addProcessElement(mainCheckMerge);

// ---- Search for tax office in charge & notify tax
office: Activity bound to a conversation
Conversation searchForTaxOfficeConversation = new
ConversationImpl();
Service searchForTaxOfficeService = new ServiceImpl();
```

```
searchForTaxOfficeService.setServiceReference();
searchForTaxOfficeConversation.addService(searchForTaxOffi
ceService);

Activity searchForTaxOffice = new ActivityImpl();
searchForTaxOffice.setName("Search For Tax Office");
searchForTaxOffice.setOperation();
searchForTaxOffice.setSynchronous(true); // blocking
operation (default)
searchForTaxOffice.setConversation(searchForTaxOfficeConve
rsation);
process.addProcessElement(searchForTaxOffice);

// Input/Output
searchForTaxOffice.addInputParameter(citizenInCRMforkParam
eter);

// ---- Create sales order for service Business
Registration Activity bound to a conversation
Conversation createSalesOrderForServiceConversation = new
ConversationImpl();
Service createSalesOrderForServiceService = new
ServiceImpl();
createSalesOrderForServiceService.setServiceReference();
createSalesOrderForServiceConversation.addService(createSa
lesOrderForServiceService);

Activity createSalesOrderForService = new ActivityImpl();
createSalesOrderForService.setName("Create Sales Order For
Service");
createSalesOrderForService.setOperation();
createSalesOrderForService.setSynchronous(true); //
blocking operation (default)
createSalesOrderForService.setConversation(createSalesOrde
rForServiceConversation);
process.addProcessElement(createSalesOrderForService);

// Input/Output
createSalesOrderForService.addInputParameter(citizenInCRMf
orkParameter);

Parameter salesOrder = new ParameterImpl();
SemanticAnnotation so = new SemanticAnnotationImpl();
so.setReferenceURI(targetNamespaceURIPrefix +
"salesOrder");
so.setType(AnnotationType.META_DATA);
salesOrder.addSemanticAnnotation(so);
createSalesOrderForService.addOutputParameter(salesOrder);
```

```
// ---- Send confirmation: Activity bound to a
conversation
Conversation sendConfirmationConversation = new
ConversationImpl();
Service sendConfirmationService = new ServiceImpl();
sendConfirmationService.setServiceReference();
sendConfirmationConversation.addService(sendConfirmationSe
rvice);

Activity sendConfirmation = new ActivityImpl();
sendConfirmation.setName("Send Confirmation");
sendConfirmation.setOperation();
sendConfirmation.setSynchronous(true); // blocking
operation (default)
sendConfirmation.setConversation(sendConfirmationConversat
ion);
process.addProcessElement(sendConfirmation);

// Input/Output
sendConfirmation.addInputParameter(citizenInCRMforkParamet
er);
sendConfirmation.addInputParameter(salesOrder);
sendConfirmation.addOutputParameter(citizenInCRMforkParame
ter);
sendConfirmation.addOutputParameter(salesOrder);

// ---- Archive: Activity bound to a conversation
Conversation archiveConversation = new ConversationImpl();
Service archiveService = new ServiceImpl();
archiveService.setServiceReference();
archiveConversation.addService(archiveService);

Activity archive = new ActivityImpl();
archive.setName("Archive");
archive.setOperation();
archive.setSynchronous(true); // blocking operation
(default)
archive.setConversation(archiveConversation);
process.addProcessElement(archive);

// Input/Output
archive.addInputParameter(citizenInCRMforkParameter);
archive.addInputParameter(preChecksParameter);
archive.addInputParameter(mainCheckParameter);
archive.addInputParameter(salesOrder);
```

```
//////////////////////////////////////////
// Flows
//////////////////////////////////////////

// Start (Receive from) -> Find citizen in CRM
process.addProcessElement(new FlowImpl(start,
findCitizenInCRM));

// Find Citizen in CRM
process.addProcessElement(new FlowImpl(findCitizenInCRM,
citizenInCRMforkGateway));

// Citizen in CRM Fork
Flow citizenInCRMforkOutgoingFlow1 = new
FlowImpl(citizenInCRMforkGateway, citizenInCRMMerge);
Flow citizenInCRMforkOutgoingFlow2 = new
FlowImpl(citizenInCRMforkGateway, createCitizenInCRM);
citizenInCRMforkOutgoingFlow1.setCondition(citizenInCRMfor
kOutgoingFlow1Condition);
citizenInCRMforkOutgoingFlow2.setCondition(citizenInCRMfor
kOutgoingFlow2Condition);
process.addProcessElement(citizenInCRMforkOutgoingFlow1);
process.addProcessElement(citizenInCRMforkOutgoingFlow2);


// Create citizen in CRM
process.addProcessElement(new FlowImpl(createCitizenInCRM,
citizenInCRMMerge));

// Citizen in CRM Merge
process.addProcessElement(new FlowImpl(citizenInCRMMerge,
readBankDetailsOfCitizen));

// Read Bank Details of Citizen
process.addProcessElement(new
FlowImpl(readBankDetailsOfCitizen,
citizenBankDetailsAvailableFork));

// Citizen Bank Details Available Fork
Flow bankDetailsOfCitizenForkPositiveFlow = new
FlowImpl(citizenBankDetailsAvailableFork,
citizenBankDetailsAvailableMerge);
Flow bankDetailsOfCitizenForkNegativeFlow = new
FlowImpl(citizenBankDetailsAvailableFork,
createBankDetailsForCitizen);
bankDetailsOfCitizenForkPositiveFlow.setCondition(bankDeta
ilsOfCitizenForkAnnotationCondition);
```

```
bankDetailsOfCitizenForkNegativeFlow.setCondition(bankDeta
ilsOfCitizenForkAnnotationCondition);
process.addProcessElement(bankDetailsOfCitizenForkPositive
Flow);
process.addProcessElement(bankDetailsOfCitizenForkNegative
Flow);

// Create Bank Details for Citizen
process.addProcessElement(new
FlowImpl(createBankDetailsForCitizen,
citizenBankDetailsAvailableMerge));

// Citizen Bank Details Available Merge
process.addProcessElement(new
FlowImpl(citizenBankDetailsAvailableMerge,
checkLocation));

// Check Location
process.addProcessElement(new FlowImpl(checkLocation,
checkLawfulness));

// Check Lawfulness
process.addProcessElement(new FlowImpl(checkLawfulness,
preCheckForkGateway));

// Pre-Check Fork
Flow preCheckForkPositiveFlow = new
FlowImpl(preCheckForkGateway, checkIdentity);
Flow preCheckForkNegativeFlow = new
FlowImpl(preCheckForkGateway, preCheckMerge);
preCheckForkPositiveFlow.setCondition(preChecksAnnotationC
ondition);
preCheckForkNegativeFlow.setCondition(preChecksAnnotationC
ondition);
process.addProcessElement(preCheckForkPositiveFlow);
process.addProcessElement(preCheckForkNegativeFlow);

// Check Identity
process.addProcessElement(new FlowImpl(checkIdentity,
checkLegalForm));

// Check Legal Form
process.addProcessElement(new FlowImpl(checkLegalForm,
checkOperationAllowance));

// Check OperationAllowance
process.addProcessElement(new
FlowImpl(checkOperationAllowance,mainCheckForkGateway));
```

```
// Main Check Fork
Flow mainCheckFormPositiveFlow = new
FlowImpl(mainCheckForkGateway, searchForTaxOffice);
Flow mainCheckFormNegativeFlow = new
FlowImpl(mainCheckForkGateway, preCheckMerge);
mainCheckFormPositiveFlow.setCondition(mainCheckAnnotation
Condition);
mainCheckFormNegativeFlow.setCondition(mainCheckAnnotation
Condition);
process.addProcessElement(mainCheckFormPositiveFlow);
process.addProcessElement(mainCheckFormNegativeFlow);

// Pre-Check Merge
process.addProcessElement(new FlowImpl(preCheckMerge,
sendDenial));

// Send Denial
process.addProcessElement(new FlowImpl(sendDenial,
mainCheckMerge));

// Main Check Merge
process.addProcessElement(new FlowImpl(mainCheckMerge,
archive));

// Search for Tax Office
process.addProcessElement(new FlowImpl(searchForTaxOffice,
createSalesOrderForService));

// Create Sales Order
process.addProcessElement(new
FlowImpl(createSalesOrderForService, sendConfirmation));

// Send Confirmation
process.addProcessElement(new FlowImpl(sendConfirmation,
mainCheckMerge));

// Archive
process.addProcessElement(new FlowImpl(archive, end));

return process;
}
}
```

9.1.2 Ontology Sample File

The following example shows an ontological description of a dimension definition. Further, the example reveals how a query could be created that uses that dimension example. The example is taken out of the project SOA4All.

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-
flight"
namespace {
_"http://www.soa4all.eu/ontologies/kb/examples/agegroup#"
 }

ontology example1

/* The present WSML file is to demonstrate a simple
example of dimension definition and how a query could be
constructed to use it.

The scenario is intended to illustrate how to indicate a
service's sensitivity to an age group and how to simply
associate a person with that service (as user.

Scenario:
- There are 4 persons of age, respectively 6, 23, 40 and
66.
- Four age groups are defined as follows:
 age group 1: 0 to 17
 age group 2: 18 to 24
 age group 3: 25 to 65
 age group 4: over 66
- Two services differ in their sensitivity to age groups.

Service 1 is sensitive to a dimension that includes age
group 2 and 3. For example, it is geared towards active
adults.
Service 2 is sensitive to a dimension that includes only
age group 4. For example, it is geared towards 'senior
citizens'.

*/

/* Domain ontology */

concept Person subConceptOf GeoLocated
   hasGender ofType Gender
   hasAgeInYears ofType _integer
   hasAgeGroup ofType AgeGroup
```

```
concept Gender
instance Male memberOf Gender
instance Female memberOf Gender

concept AgeGroup
   hasInRange ofType _integer
instance AgeGroup0to17 memberOf AgeGroup
instance AgeGroup18to24 memberOf AgeGroup
instance AgeGroup25to65 memberOf AgeGroup
instance AgeGroup66to memberOf AgeGroup

axiom agegroup0to17def
 definedBy
   ?x[hasAgeGroup hasValue AgeGroup0to17]
   impliedBy ?x[hasAgeInYears hasValue ?a] and ?a =< 17.

axiom agegroup18to24def
 definedBy
  ?x[hasAgeGroup hasValue AgeGroup18to24]
  impliedBy ?x[hasAgeInYears hasValue ?a] and ?a >= 18 and
?a =< 24.

axiom agegroup25to65def
 definedBy
  ?x[hasAgeGroup hasValue AgeGroup25to65]
  impliedBy ?x[hasAgeInYears hasValue ?a] and ?a >= 25 and
?a =< 65.

axiom agegroup66todef
 definedBy
  ?x[hasAgeGroup hasValue AgeGroup66to]
  impliedBy ?x[hasAgeInYears hasValue ?a] and ?a >= 66.

/* Person KB */
instance MrsGoggins memberOf Person
   hasGender hasValue Female
   hasAgeInYears hasValue 66
instance Patrick memberOf Person
   hasGender hasValue Male
   hasAgeInYears hasValue 40
instance Amy memberOf Person
   hasGender hasValue Female
   hasAgeInYears hasValue 23
instance Julian memberOf Person
   hasGender hasValue Male
   hasAgeInYears hasValue 6

/* Sensitivity (of Service) to agegroup */
```

```
concept Service
   hasSensitivityTo ofType Dimension

concept Dimension
   hasDimensionValue impliesType DimensionValue

concept DimensionValue
instance AgeGroupDimensionUnder65 memberOf Dimension
   hasDimensionValue hasValue {AgeGroup0to17,
AgeGroup18to24, AgeGroup25to65}
   hasAssociatedSlot hasValue hasAgeGroup
instance AgeGroupDimensionOver18Under65 memberOf Dimension
   hasDimensionValue hasValue {AgeGroup18to24,
AgeGroup25to65}
   hasAssociatedSlot hasValue hasAgeGroup
instance AgeGroupDimensionOver66 memberOf Dimension
   hasDimensionValue hasValue AgeGroup66to
   hasAssociatedSlot hasValue hasAgeGroup
instance Service1 memberOf Service
   hasSensitivityTo hasValue
AgeGroupDimensionOver18Under65
instance Service2 memberOf Service
   hasSensitivityTo hasValue AgeGroupDimensionOver66

/* select is the service-person dummy relation for this
example.

The two more specific relations, select1 and select 2, are
only used here to illustrate specifically two axioms for
infering select-relationships. The first axiom relies
merely on the fact that dimensions have associated values.
The second axiom abstract over the slot associated with a
dimension and is therefore the only one reusable without
modification.  * /

relation select (ofType Service, ofType Person)
relation select1 (ofType Service, ofType Person)
subRelationOf select
relation select2 (ofType Service, ofType Person)
subRelationOf select

 /* SIMPLE:

This axiom allows to infer (Service,Person) relationships.
A relationship between a service S and a person P will be
inferred when S is sensitive to a defined dimension that
has as one of its values the age group to which P is
```

```
associated through the hasAgeGroup slot of the concept
Person.
 */

axiom selectdefsimple
definedBy
   select1(?x, ?y)
   impliedBy
    ?x[hasSensitivityTo hasValue ?d]
    and ?d[hasDimensionValue hasValue ?v]
    and ?y[hasAgeGroup hasValue ?v].

 /* General (with property variable)

The following axiom, selectdefwithslot, is an abstraction
over the previous axiom, selectdefsimple. In
selectdefwithslot ?s is a variable and its value is found
as the value for the 'hasAssociatedSlot' property of a
dimension which is a value for the variable ?d.
*/

axiom selectdefwithslot
definedBy
   select2(?x, ?y)
   impliedBy
      ?x[hasSensitivityTo hasValue ?d]
      and ?d[hasDimensionValue hasValue ?v]
      and ?d[hasAssociatedSlot hasValue ?s]
      and ?y[?s hasValue ?v] .

/* Query example (tested in the WSMT2.0 reasonner's tab)

Query: select(?x , ?y )
Results:
?x: Service1  ?y: Patrick
?x: Service1  ?y: Amy
?x: Service2  ?y: MrsGoggins

(The same results are obtained for the relations select1
and select2.)
*/
```

9.2 Evaluation Workshops

As described in section 7.2.1 a focus group evaluation workshop has been conducted in the context of the project SOA4All in order to gather feedback about the main ideas of the LPMS and service composition in general.

The main objectives of the workshop had been to

- Obtain general opinions of the business users about end-user development of service-based software;
- Evaluate and compare the current mock-ups of the process editor within a participatory design process;
- Gather as many LPMS requirements as possible.

The workshop execution has been as follows:

- At least 12 participants for the planned session that had been divided into 3 groups, plus one moderator for each group.
- The session lasted for about 3.5 hours in a large seminar room with 3 round tables seating 6 each, including a 20-minutes break in the middle.
- A 30-minutes introductory talk has been followed by a 20-minutes discussion on the perceptions about risks and benefits of the envisioned mode of user-driven service composition, and on existing practices and proposed supporting actions
- A short notational study has discovered how participants understand core proposed representations of the LPML
- After the break, the discussion focused on alternative designs for a business user tool for service composition. Questionnaires and audio tapes have been used to record the participants' responses for the analysis.

In order to gather information about the background of business users, their general ideas and thoughts about service composition specific questions have been asked, such as:

- What information/ system parts do you consider is important for your job?
- What features do you like about your current tools?
- What features do you dislike about your current tools?
- What aspects do you consider problematic in your current tools?

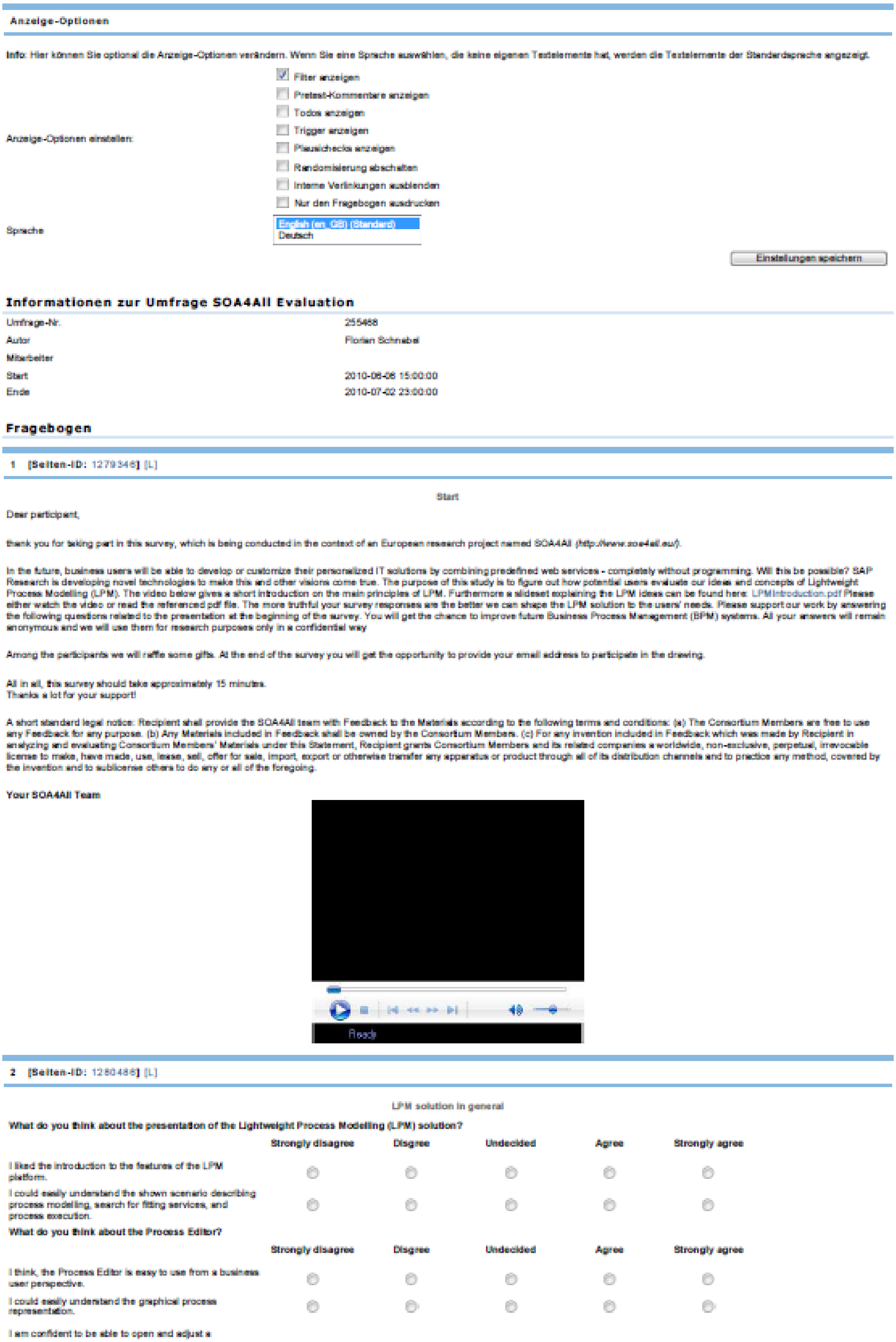
Other questions have focused on end user development issues:

- What are the benefits of end user development?
- What are the risks of end user development?
- What strategies / approaches do end users follow when developing applications?

9.3 EVALUATION SURVEY

In the following, the evaluation survey questions conducted in the context of this thesis are illustrated. The survey has been implemented through the survey tool Unipark[28].

[28] See www.unipark.de



Anzeige-Optionen

Info: Hier können Sie optional die Anzeige-Optionen verändern. Wenn Sie eine Sprache auswählen, die keine eigenen Textelemente hat, werden die Textelemente der Standardsprache angezeigt.

Anzeige-Optionen einstellen:

☑ Filter anzeigen
□ Pretest-Kommentare anzeigen
□ Todos anzeigen
□ Trigger anzeigen
□ Plausichecks anzeigen
□ Randomisierung abschalten
□ Interne Verlinkungen ausblenden
□ Nur den Fragebogen ausdrucken

Sprache: English (en_GB) (Standard) / Deutsch

Informationen zur Umfrage SOA4All Evaluation

Umfrage-Nr.	255468
Autor	Florian Schnabel
Mitarbeiter	
Start	2010-06-06 15:00:00
Ende	2010-07-02 23:00:00

Fragebogen

1 [Seiten-ID: 1279346**]** [L]

Start

Dear participant,

thank you for taking part in this survey, which is being conducted in the context of an European research project named SOA4All (http://www.soa4all.eu/).

In the future, business users will be able to develop or customize their personalized IT solutions by combining predefined web services - completely without programming. Will this be possible? SAP Research is developing novel technologies to make this and other visions come true. The purpose of this study is to figure out how potential users evaluate our ideas and concepts of Lightweight Process Modelling (LPM). The video below gives a short introduction on the main principles of LPM. Furthermore a slideset explaining the LPM ideas can be found here: LPMIntroduction.pdf Please either watch the video or read the referenced pdf file. The more truthful your survey responses are the better we can shape the LPM solution to the users' needs. Please support our work by answering the following questions related to the presentation at the beginning of the survey. You will get the chance to improve future Business Process Management (BPM) systems. All your answers will remain anonymous and we will use them for research purposes only in a confidential way

Among the participants we will raffle some gifts. At the end of the survey you will get the opportunity to provide your email address to participate in the drawing.

All in all, this survey should take approximately 15 minutes.
Thanks a lot for your support!

A short standard legal notice: Recipient shall provide the SOA4All team with Feedback to the Materials according to the following terms and conditions: (a) The Consortium Members are free to use any Feedback for any purpose. (b) Any Materials included in Feedback shall be owned by the Consortium Members. (c) For any invention included in Feedback which was made by Recipient in analyzing and evaluating Consortium Members' Materials under this Statement, Recipient grants Consortium Members and its related companies a worldwide, non-exclusive, perpetual, irrevocable license to make, have made, use, lease, sell, offer for sale, import, export or otherwise transfer any apparatus or product through all of its distribution channels and to practice any method, covered by the invention and to sublicense others to do any or all of the foregoing.

Your SOA4All Team

2 [Seiten-ID: 1280486**]** [L]

LPM solution in general

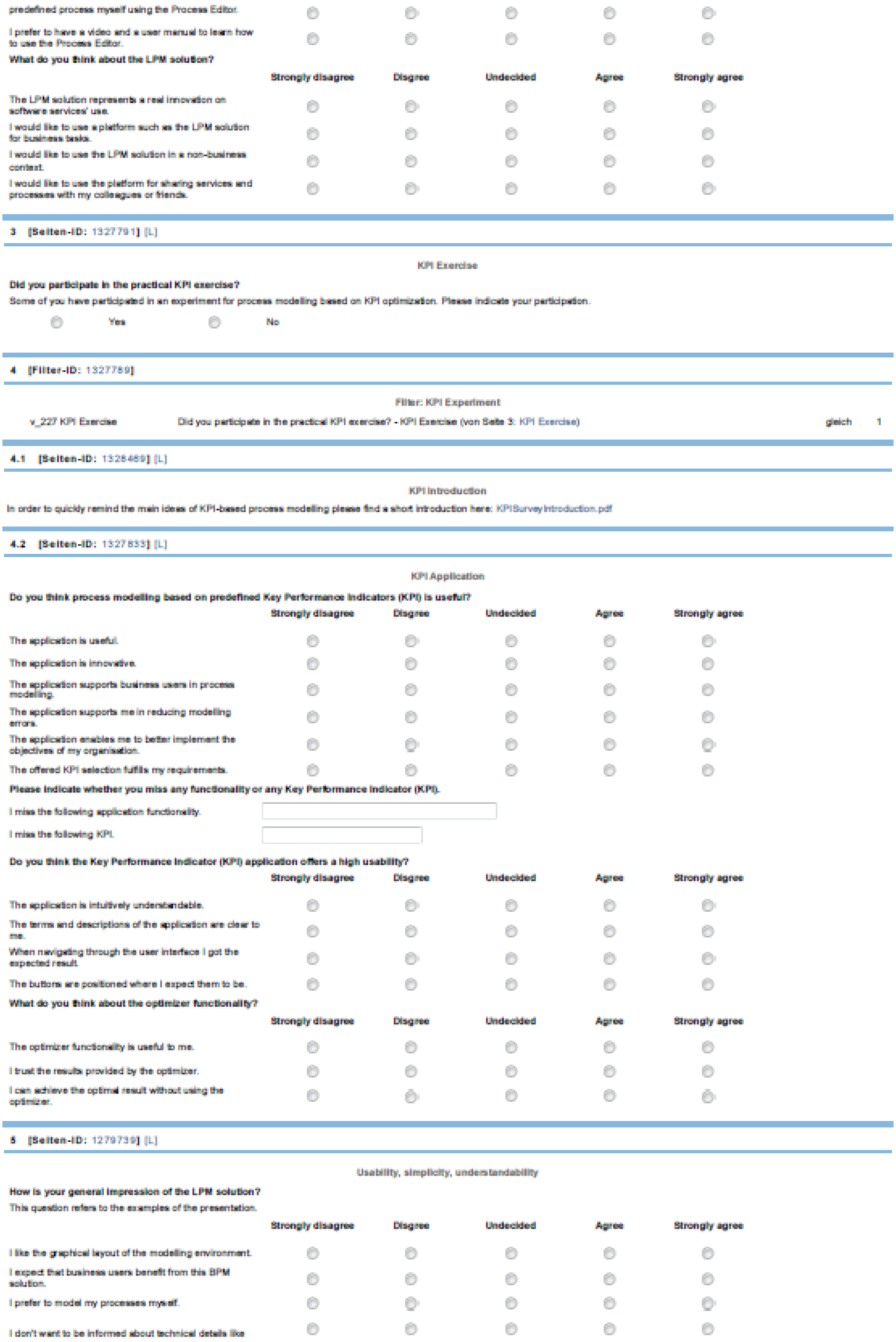
What do you think about the presentation of the Lightweight Process Modelling (LPM) solution?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
I liked the introduction to the features of the LPM platform.	○	○	○	○	○
I could easily understand the shown scenario describing process modelling, search for fitting services, and process execution.	○	○	○	○	○

What do you think about the Process Editor?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
I think, the Process Editor is easy to use from a business user perspective.	○	○	○	○	○
I could easily understand the graphical process representation.	○	○	○	○	○

I am confident to be able to open and adjust a



predefined process myself using the Process Editor.	○	○	○	○	○
I prefer to have a video and a user manual to learn how to use the Process Editor.	○	○	○	○	○

What do you think about the LPM solution?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
The LPM solution represents a real innovation on software services' use.	○	○	○	○	○
I would like to use a platform such as the LPM solution for business tasks.	○	○	○	○	○
I would like to use the LPM solution in a non-business context.	○	○	○	○	○
I would like to use the platform for sharing services and processes with my colleagues or friends.	○	○	○	○	○

3 [Seiten-ID: 1327791] [L]

KPI Exercise

Did you participate in the practical KPI exercise?

Some of you have participated in an experiment for process modelling based on KPI optimization. Please indicate your participation.

○ Yes ○ No

4 [Filter-ID: 1327789]

Filter: KPI Experiment

v_227 KPI Exercise	Did you participate in the practical KPI exercise? - KPI Exercise (von Seite 3: KPI Exercise)	gleich	1

4.1 [Seiten-ID: 1328469] [L]

KPI Introduction

In order to quickly remind the main ideas of KPI-based process modelling please find a short introduction here: KPISurveyIntroduction.pdf

4.2 [Seiten-ID: 1327833] [L]

KPI Application

Do you think process modelling based on predefined Key Performance Indicators (KPI) is useful?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
The application is useful.	○	○	○	○	○
The application is innovative.	○	○	○	○	○
The application supports business users in process modelling.	○	○	○	○	○
The application supports me in reducing modelling errors.	○	○	○	○	○
The application enables me to better implement the objectives of my organisation.	○	○	○	○	○
The offered KPI selection fulfils my requirements.	○	○	○	○	○

Please indicate whether you miss any functionality or any Key Performance Indicator (KPI).

I miss the following application functionality. ____________

I miss the following KPI. ____________

Do you think the Key Performance Indicator (KPI) application offers a high usability?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
The application is intuitively understandable.	○	○	○	○	○
The terms and descriptions of the application are clear to me.	○	○	○	○	○
When navigating through the user interface I got the expected result.	○	○	○	○	○
The buttons are positioned where I expect them to be.	○	○	○	○	○

What do you think about the optimizer functionality?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
The optimizer functionality is useful to me.	○	○	○	○	○
I trust the results provided by the optimizer.	○	○	○	○	○
I can achieve the optimal result without using the optimizer.	○	○	○	○	○

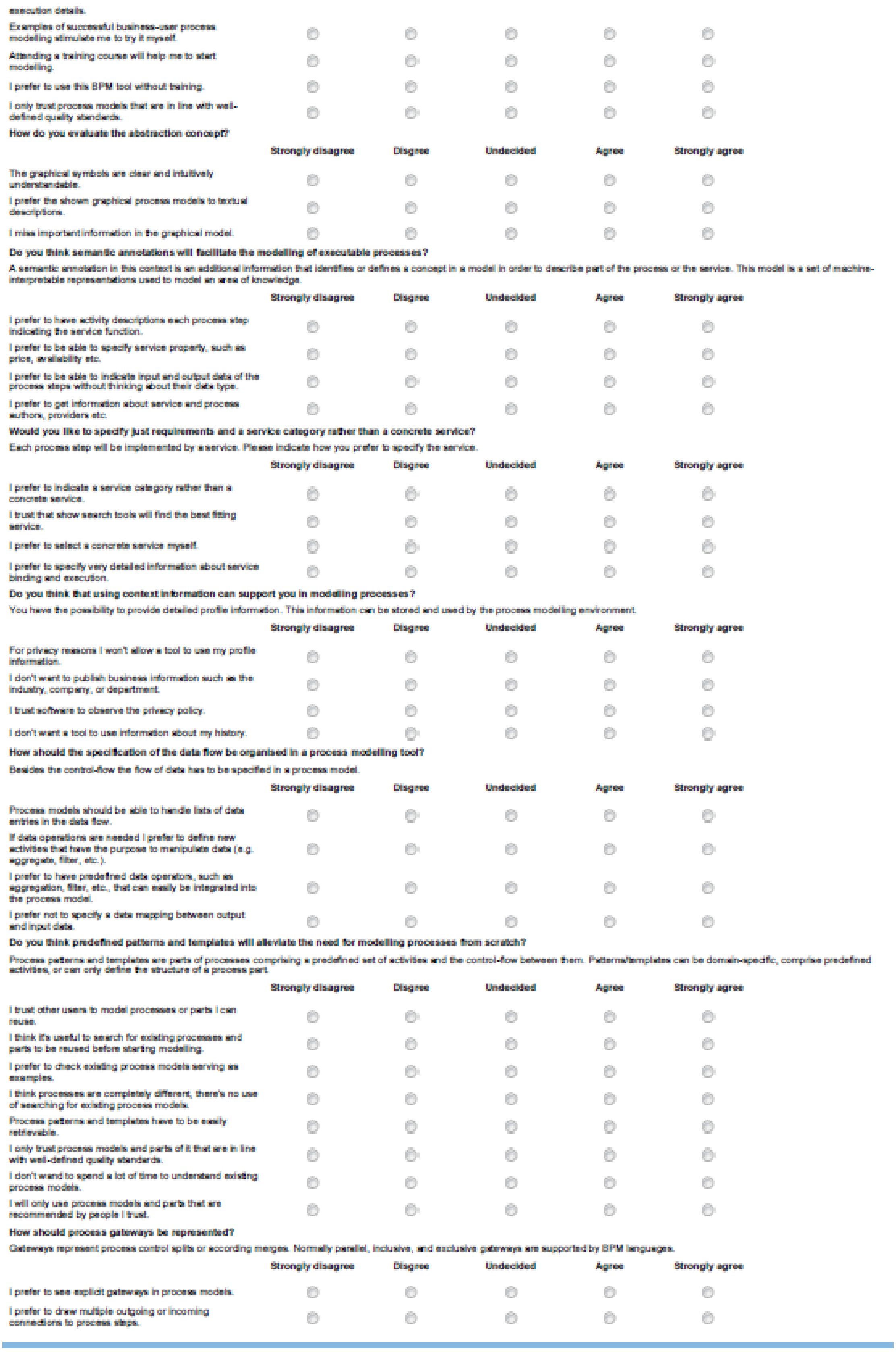
5 [Seiten-ID: 1279739] [L]

Usability, simplicity, understandability

How is your general impression of the LPM solution?

This question refers to the examples of the presentation.

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
I like the graphical layout of the modelling environment.	○	○	○	○	○
I expect that business users benefit from this BPM solution.	○	○	○	○	○
I prefer to model my processes myself.	○	○	○	○	○
I don't want to be informed about technical details like	○	○	○	○	○



execution details.

Examples of successful business-user process modelling stimulate me to try it myself.	○	○	○	○	○
Attending a training course will help me to start modelling.	○	○	○	○	○
I prefer to use this BPM tool without training.	○	○	○	○	○
I only trust process models that are in line with well-defined quality standards.	○	○	○	○	○

How do you evaluate the abstraction concept?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
The graphical symbols are clear and intuitively understandable.	○	○	○	○	○
I prefer the shown graphical process models to textual descriptions.	○	○	○	○	○
I miss important information in the graphical model.	○	○	○	○	○

Do you think semantic annotations will facilitate the modelling of executable processes?

A semantic annotation in this context is an additional information that identifies or defines a concept in a model in order to describe part of the process or the service. This model is a set of machine-interpretable representations used to model an area of knowledge.

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
I prefer to have activity descriptions each process step indicating the service function.	○	○	○	○	○
I prefer to be able to specify service property, such as price, availability etc.	○	○	○	○	○
I prefer to be able to indicate input and output data of the process steps without thinking about their data type.	○	○	○	○	○
I prefer to get information about service and process authors, providers etc.	○	○	○	○	○

Would you like to specify just requirements and a service category rather than a concrete service?

Each process step will be implemented by a service. Please indicate how you prefer to specify the service.

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
I prefer to indicate a service category rather than a concrete service.	○	○	○	○	○
I trust that show search tools will find the best fitting service.	○	○	○	○	○
I prefer to select a concrete service myself.	○	○	○	○	○
I prefer to specify very detailed information about service binding and execution.	○	○	○	○	○

Do you think that using context information can support you in modelling processes?

You have the possibility to provide detailed profile information. This information can be stored and used by the process modelling environment.

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
For privacy reasons I won't allow a tool to use my profile information.	○	○	○	○	○
I don't want to publish business information such as the industry, company, or department.	○	○	○	○	○
I trust software to observe the privacy policy.	○	○	○	○	○
I don't want a tool to use information about my history.	○	○	○	○	○

How should the specification of the data flow be organised in a process modelling tool?

Besides the control-flow the flow of data has to be specified in a process model.

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
Process models should be able to handle lists of data entries in the data flow.	○	○	○	○	○
If data operations are needed I prefer to define new activities that have the purpose to manipulate data (e.g. aggregate, filter, etc.).	○	○	○	○	○
I prefer to have predefined data operators, such as aggregation, filter, etc., that can easily be integrated into the process model.	○	○	○	○	○
I prefer not to specify a data mapping between output and input data.	○	○	○	○	○

Do you think predefined patterns and templates will alleviate the need for modelling processes from scratch?

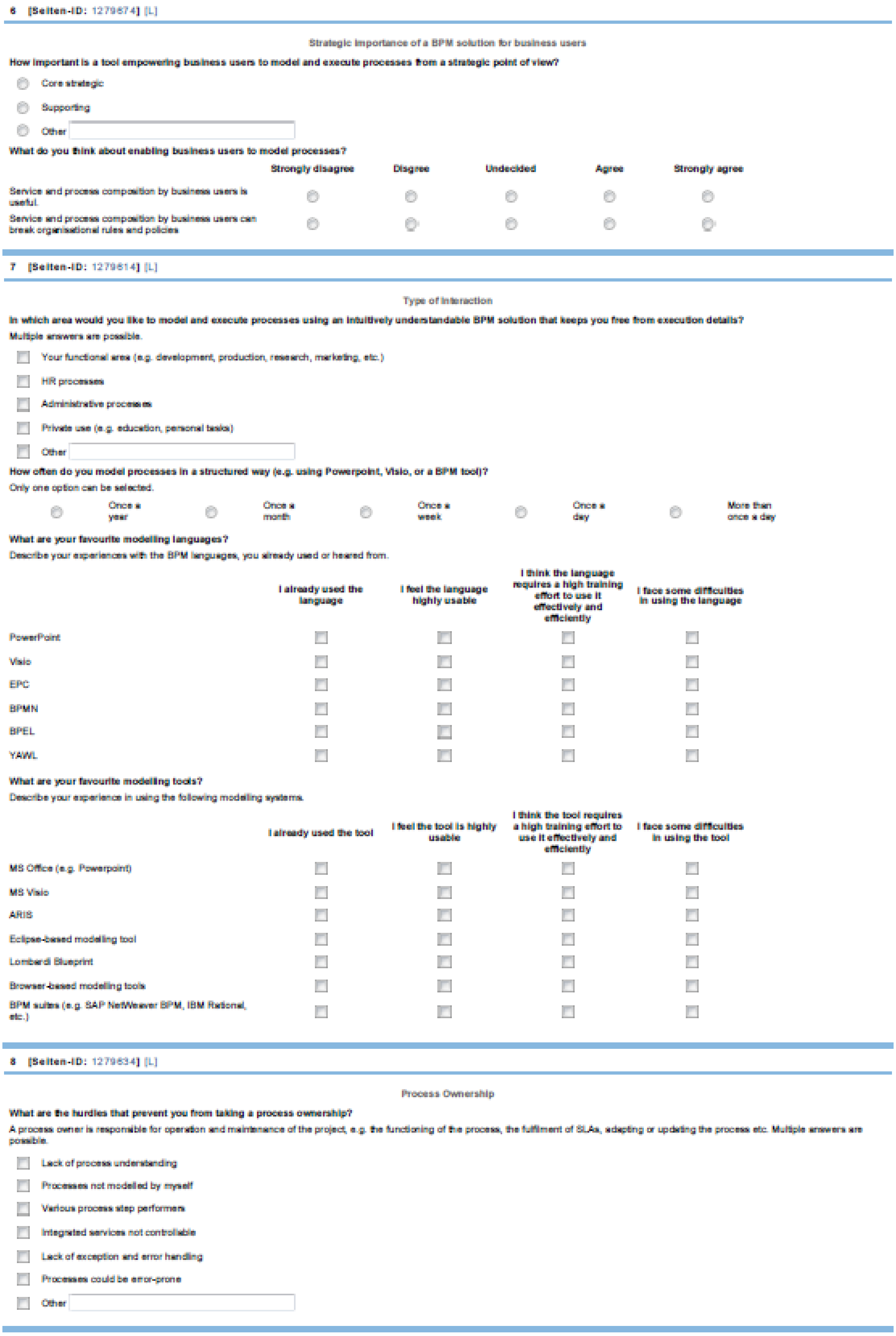
Process patterns and templates are parts of processes comprising a predefined set of activities and the control-flow between them. Patterns/templates can be domain-specific, comprise predefined activities, or can only define the structure of a process part.

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
I trust other users to model processes or parts I can reuse.	○	○	○	○	○
I think it's useful to search for existing processes and parts to be reused before starting modelling.	○	○	○	○	○
I prefer to check existing process models serving as examples.	○	○	○	○	○
I think processes are completely different, there's no use of searching for existing process models.	○	○	○	○	○
Process patterns and templates have to be easily retrievable.	○	○	○	○	○
I only trust process models and parts of it that are in line with well-defined quality standards.	○	○	○	○	○
I don't wend to spend a lot of time to understand existing process models.	○	○	○	○	○
I will only use process models and parts that are recommended by people I trust.	○	○	○	○	○

How should process gateways be represented?

Gateways represent process control splits or according merges. Normally parallel, inclusive, and exclusive gateways are supported by BPM languages.

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
I prefer to see explicit gateways in process models.	○	○	○	○	○
I prefer to draw multiple outgoing or incoming connections to process steps.	○	○	○	○	○



6 [Seiten-ID: 1279674] [L]

Strategic importance of a BPM solution for business users

How important is a tool empowering business users to model and execute processes from a strategic point of view?

○ Core strategic

○ Supporting

○ Other ________

What do you think about enabling business users to model processes?

	Strongly disagree	Disgree	Undecided	Agree	Strongly agree
Service and process composition by business users is useful.	○	○	○	○	○
Service and process composition by business users can break organisational rules and policies	○	○	○	○	○

7 [Seiten-ID: 1279614] [L]

Type of interaction

In which area would you like to model and execute processes using an intuitively understandable BPM solution that keeps you free from execution details?

Multiple answers are possible.

□ Your functional area (e.g. development, production, research, marketing, etc.)

□ HR processes

□ Administrative processes

□ Private use (e.g. education, personal tasks)

□ Other ________

How often do you model processes in a structured way (e.g. using Powerpoint, Visio, or a BPM tool)?

Only one option can be selected.

○ Once a year ○ Once a month ○ Once a week ○ Once a day ○ More than once a day

What are your favourite modelling languages?

Describe your experiences with the BPM languages, you already used or heared from.

	I already used the language	I feel the language highly usable	I think the language requires a high training effort to use it effectively and efficiently	I face some difficulties in using the language
PowerPoint	□	□	□	□
Visio	□	□	□	□
EPC	□	□	□	□
BPMN	□	□	□	□
BPEL	□	□	□	□
YAWL	□	□	□	□

What are your favourite modelling tools?

Describe your experience in using the following modelling systems.

	I already used the tool	I feel the tool is highly usable	I think the tool requires a high training effort to use it effectively and efficiently	I face some difficulties in using the tool
MS Office (e.g. Powerpoint)	□	□	□	□
MS Visio	□	□	□	□
ARIS	□	□	□	□
Eclipse-based modelling tool	□	□	□	□
Lombardi Blueprint	□	□	□	□
Browser-based modelling tools	□	□	□	□
BPM suites (e.g. SAP NetWeaver BPM, IBM Rational, etc.)	□	□	□	□

8 [Seiten-ID: 1279634] [L]

Process Ownership

What are the hurdles that prevent you from taking a process ownership?

A process owner is responsible for operation and maintenance of the project, e.g. the functioning of the process, the fulfilment of SLAs, adapting or updating the process etc. Multiple answers are possible.

□ Lack of process understanding

□ Processes not modelled by myself

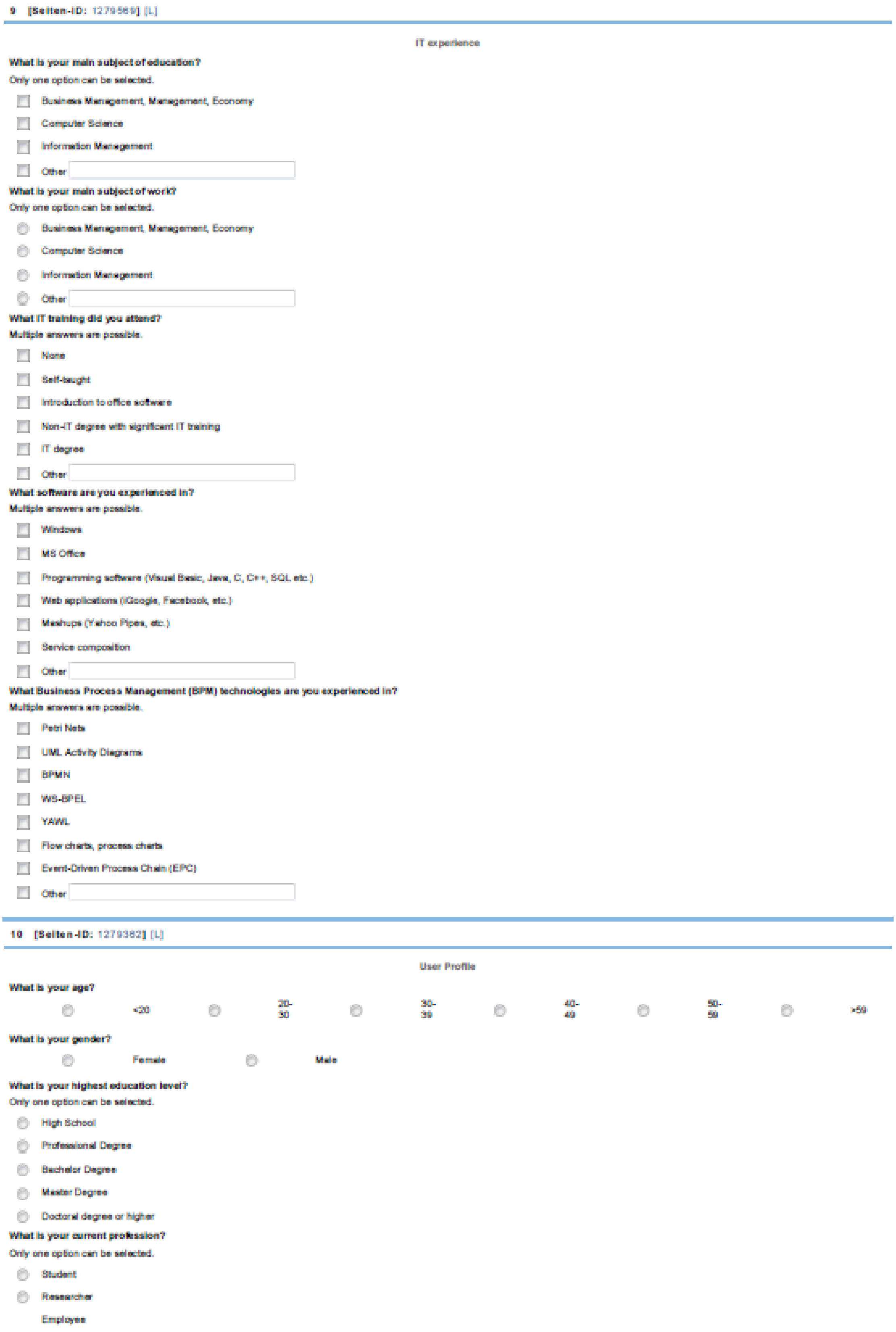
□ Various process step performers

□ Integrated services not controllable

□ Lack of exception and error handling

□ Processes could be error-prone

□ Other ________



9 [Seiten-ID: 1279569] [L]

IT experience

What is your main subject of education?

Only one option can be selected.

□ Business Management, Management, Economy

□ Computer Science

□ Information Management

□ Other

What is your main subject of work?

Only one option can be selected.

○ Business Management, Management, Economy

○ Computer Science

○ Information Management

○ Other

What IT training did you attend?

Multiple answers are possible.

□ None

□ Self-taught

□ Introduction to office software

□ Non-IT degree with significant IT training

□ IT degree

□ Other

What software are you experienced in?

Multiple answers are possible.

□ Windows

□ MS Office

□ Programming software (Visual Basic, Java, C, C++, SQL etc.)

□ Web applications (iGoogle, Facebook, etc.)

□ Mashups (Yahoo Pipes, etc.)

□ Service composition

□ Other

What Business Process Management (BPM) technologies are you experienced in?

Multiple answers are possible.

□ Petri Nets

□ UML Activity Diagrams

□ BPMN

□ WS-BPEL

□ YAWL

□ Flow charts, process charts

□ Event-Driven Process Chain (EPC)

□ Other

10 [Seiten-ID: 1279382] [L]

User Profile

What is your age?

○ <20 ○ 20-30 ○ 30-39 ○ 40-49 ○ 50-59 ○ >59

What is your gender?

○ Female ○ Male

What is your highest education level?

Only one option can be selected.

○ High School

○ Professional Degree

○ Bachelor Degree

○ Master Degree

○ Doctoral degree or higher

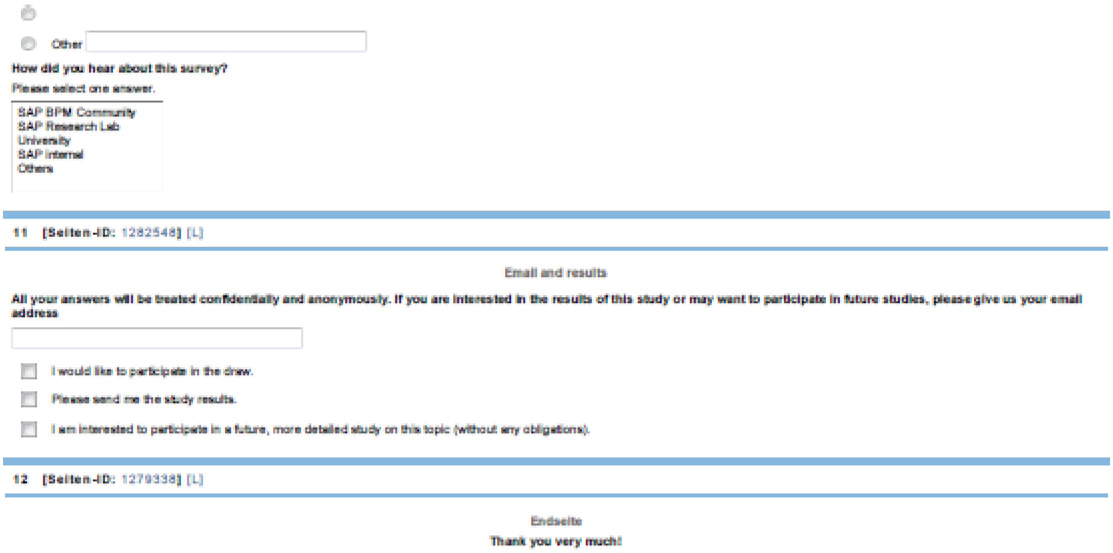
What is your current profession?

Only one option can be selected.

○ Student

○ Researcher

Employee



○

○ Other

How did you hear about this survey?

Please select one answer.

SAP BPM Community
SAP Research Lab
University
SAP internal
Others

11 [Seiten-ID: 1282548] [L]

Email and results

All your answers will be treated confidentially and anonymously. If you are interested in the results of this study or may want to participate in future studies, please give us your email address

□ I would like to participate in the draw.

□ Please send me the study results.

□ I am interested to participate in a future, more detailed study on this topic (without any obligations).

12 [Seiten-ID: 1279338] [L]

Endseite

Thank you very much!

10 BIBLIOGRAPHY

Adams, M., Ter, Edmond, D., & van der Aalst, W. (2006). Worklets: A Service-Oriented Implementation of Dynamic Flexibility in Workflows. In *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE* (pp. 291-308).

Adobe, S. I. (2008). *LiveCycle ES Overview*.

Allaire, J. (2002). *Macromedia MX - A next generation rich client*. San Francisco.

Allen, R. (2001). *Workflow: An Introduction*: Workflow Management Coalition.

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). *Web Services: Concepts, Architectures and Applications*. Berlin: Springer Verlag.

Andrikopoulos, V., Bertoli, P., Bindelli, S., Di Nitto, E., Gehlert, A., Germanovich, L., et al. (2008). *State of the art report on software engineering design knowledge and Survey of HCI and contextual Knowledge*: S-CUBE.

Appian. (2010). *Appian Business Process Management Suite*, from www.appian.com

Ardissano, L., Furnari, R., Goy, A., Petrone, G., & Segnan, M. (2007). *A framework for the mangement of context-aware workflow systems.* Paper presented at the Third International Conference on Web Information Systems and Technologies, Barcelona, Spain.

Armistead, C., Machin, S., & Pritchard, J. P. (1997). *Approaches to business process management.* Paper presented at the International Conference of the European Operations Management Association, IESE, Barcelona, Spain.

Barricelli, B. R., Mussio, P., Valtolina, S., Padula, M., Scala, P. L., & Piccinno, A. (2010). *Visual Workflow Composition through Semantic Orchestration of Web Services.* Paper presented at the EUD4Services Workshop in conjunction with AVI2010, Rome.

Becker, J., Rosemann, M., & Kugeler, M. (2003). *Process Management*: Springer-Verlag New York, Inc.

Becker, J., Rosemann, M., & Schütte, R. (1995). Grundsätze ordnungsgemäßer Modellierung. *Wirtschaftsinformatik, 37*.

Benslimane, D., Dustdar, S., & Sheth, A. (2008). Services Mashups: The New Generation of Web Applications. *IEEE Internet Computing, 12*(5), 13-15.

Bittinger, S., & Di Maio, A. (2010). *Hype Cycle for Government Transformation, 2010*.

Blackwell, A. F. (2002). First Steps in Programming: A Rationale for Attention Investment Models. In *Proceedings of the IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC'02)* (pp. 2): IEEE Computer Society.

Blechar, M. (2007). *Magic Quadrant for Business Process Analysis Tools, 2H07-1H08*.

Blythe, J., Deelman, E., & Gil, Y. (2003). *Planning for workflow construction and maintenance on the grid.* Paper presented at the ICAPS 03 - Workshop on Web Services Composition.

Böhmann, T., Junginger, M., & Krcmar, H. (2003). *Modular Service Architectures: A concept and method for engineering IT services.* Paper presented at the 36th Hawaii international conference on system sciences (HICSS'03), Hawaii.

Born, M. (2009). *Towards efficient user guidance in Business Process Modeling.* Unpublished Dissertation Thesis, TU Clausthal, Clausthal.

Born, M., Hoffmann, J. r., Kaczmarek, T., Kowalkiewicz, M., Markovic, I., Scicluna, J., et al. (2009). Supporting Execution-Level Business Process Modeling with Semantic Technologies. In *Proceedings of the 14th International Conference on Database Systems for Advanced Applications* (pp. 759-763). Brisbane, Australia: Springer-Verlag.

Bouquet, P., Fausto, G., Van Harmelen, F., Serafini, L., & Stuckenschmidt, H. (2003). C-OWL: Contextualizing Ontologies. In *Journal of Web Semantics* (pp. 164-179): Springer Verlag.

Brancheau, J. C., & Brown, C. V. (1993). The management of end-user computing: status and directions. *ACM Comput. Surv., 25*(4), 437-482.

Bretzke, W.-R. (1980). *Der Problembezug von Entscheidungsmodellen*. Tübingen: Mohr.

Brinkkemper, S., Saeki, M., & Harmsen, F. (1999). Meta-modelling based assembly techniques for situational method engineering. *Information Systems, 24*(3), 209-228.

Brockhaus, D. (2002). *Der Brockhaus Computer und Informationstechnologie*. Leipzig - Mannheim: F.A. Brockhaus GmbH.

Brogi, A., & Popescu, R. (2007). Service adaptation through trace inspection. *International Journal of Business Process Integration and Management (IJBPMIM), 2*(1), 9-16.

Brown, W. A., & Cantor, M. (2006). *SOA governance: how to oversee successful implementation through proven best practices and methods*: IBM.

Bunge, M. (1977). Treatise on Basic Philosophy. *Ontology I: The furniture of the world, 3*.

Buschmann, F., Henney, K., & Schmidt, D. C. (2007). Past, Present, and Future Trends in Software Patterns. *IEEE Software, 24*(7/8), 31-37.

Canning, R. G. (1981a). Programming by end users. *EDP Analyzer, 19*(5).

Canning, R. G. (1981b). Supporting end-user programming. *EDP Analyzer, 19*(6).

Cantara, M. (2009). *Research Index: A Guide to Principles of Business Process Management*: Gartner Research.

Cantara, M., Plummer, D. C., Kenney, L. F., White, A., Wilson, D. R., Lheureux, B. J., et al. (2009). *Hype Cycle for Business Process Management, 2009*.

Casati, F., Ilnicki, S., Jin, L.-j., Krishnamoorthy, V., & Shan, M.-C. (2000). Adaptive and Dynamic Service Composition in eFlow. In *Proceedings of the 12th International Conference on Advanced Information Systems Engineer ing* (pp. 13-31): Springer-Verlag.

Cavallaro, L., & Nitto, E. D. (2008). An approach to adapt service requests to actual service interfaces. In *Proceedings of the 2008 international workshop on Software engineering for adaptive and self-managing systems* (pp. 129-136). Leipzig, Germany: ACM.

Cavallaro, L., Ripa, G., & Zuccala, M. (2009). Adapting service requests to actual service interfaces through semantic annotations. In *Proceedings of the 2009*

ICSE Workshop on Principles of Engineering Service Oriented Systems (pp. 83-86): IEEE Computer Society.

Chandrasekaran, B. (1990). Design problem solving: a task analysis. *AI Mag., 11*(4), 59-71.

Chang, J. F. (2005). *Business Process Management Systems: Strategy and Implementation*: Auerbach Publications.

Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001). *Web Service Definition Language (WSDL)*.

Cimpian, E., Moran, M., Oren, E., Vitvar, T., & Zaremba, M. (2005). *D13.0 v0.2 Overview and Scope of WSMX*: WSMO Working Group.

Colombo, M., Di Nitto, E., & Mauri, M. (2006). SCENE: A Service Composition Execution Environment Supporting Dynamic Changes Disciplined Through Rules, *Service-Oriented Computing â€" ICSOC 2006* (pp. 191-202).

Commission, E. (2006). *Directive 2006/123/EC of the European Parliament and of the Council of 12 December 2006 on Services in the Internal Market, OJ L376*.

Commission, E. (2007). *Handbook on Implementation of the Services Directive*.

Cordys. (2010). *Cordys Business Process Management Suite*.

Curtis, B., Kellner, M. I., & Over, J. (1992). Process modeling. *Commun. ACM, 35*(9), 75-90.

Czarnecki, K., Eisenecker, U., Glueck, R., Vandevoorde, D., & Veldhuizen, T. (2000). Generative Programming and Active Libraries. In M. Jazayeri, R. Loos & D. Musser (Eds.), *Generic Programming* (Vol. 1766, pp. 25-39): Springer Berlin / Heidelberg.

Davenport, T. H. (1993). *Process Innovation: Reengineering Work through Information Technology*. Boston: Harvard Business School Press.

Davis, R. (2001). *Business Process Modelling with ARIS*. London: Springer-Verlag.

Denning, P., Horning, J., Parnas, D., & Weinstein, L. (2005). Wikipedia risks. *Commun. ACM, 48*(12), 152-152.

Destatis. (2010). *Peronnel of public service institutions*. Retrieved 02/08/2010, from http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/EN/Content/Statistics/FinanzenSteuern/PublicService/PublicServicePersonnel/Aktuell,templateId=renderPrint.psml

Dey, A. K. (2001). Understanding and Using Context. *Personal Ubiquitous Comput., 5*(1), 4-7.

Dimitrov, M., Simov, A., Stein, S., & Konstantinov, M. (2007). A BPMO Based Semantic Business Process Modelling Environment. In *SBPM*.

Dostal, W., Jeckle, M., Melzer, I., & Zengler, B. (2005). *Service-orientierte Architekturen mit Web Services*. München: Spektrum Akademischer Verlag.

Driver, E., & Rogowski, R. (2007). *RIAs bring people-centered design to information workplaces*.

Duesseldorf. (2009). *oEPK Prozessmodell "Elektronische Gewerbeanmeldung"*. Retrieved 16.07.2010, from http://www.duesseldorf.de/egovernment/pdf/oepkkommunaleverfahren.pdf

Ehrig, M., Koschmider, A., & Oberweis, A. (2007). Measuring similarity between semantic business process models. In *Proceedings of the fourth Asia-Pacific conference on Comceptual modelling - Volume 67* (pp. 71-80). Ballarat, Australia: Australian Computer Society, Inc.

Elzinga, D. J., Horak, T., Chung-Yee, L., & Brunner, C. (1995). Business Process Management: Survey and Methodology. *IEEE Transactions on Engineering Management, 24*(2), 119-128.

Fettke, P., & Loos, P. (2003). *Ontological evaluation of reference models using the Bunge-Wand-Weber model.* Paper presented at the Americas Conference on Information Systems (AMCIS), Tampa.

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures.* Unpublished PhD Thesis, University of California, Irvine.

Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A. G., & Mehandjiev, N. (2004). Meta-design: a manifesto for end-user development. *Commun. ACM, 47*(9), 33-37.

Fötsch, D., Speck, A., & Hänsgen, P. (2005). The Operator Hierarchy Concept for XML Document Transformation Technologies. In R. Eckstein & R. Tolksdorf (Eds.), *Berliner XML Tage 2005* (pp. 59-70). Berlin: Humboldt-Universität zu Berlin.

Frank, J. H., Gardner, T. A., Johnston, S. K., White, S. A., & Iyengar, S. (2004). *Business Process Definition Metamodel - Concepts and overview.* Retrieved 04.12.06, 2006, from http://www.omg.org/docs/bei/04-05-03.pdf

Frank, U. (1998). *Evaluating Modeling Languages: Relevant Issues, Epistemological Challenges and a Preliminary Research Framework*. Koblenz: Universitaet Koblenz Landau, Institut für Wirtschaftsinformatik.

Gadatsch, A. (2005). *Grundkurs Geschäftsprozess-Management*: Vieweg Friedr. + Sohn Ver.

Gao, Y. (2008). *BPMN-BPEL Transformation and Round Trip Engineering*: OASIS.

Geelan, J. (2008). *The Business Value of RIAs: An informal, virtual round table*. Retrieved 08/09/2008, from http://pbdj.sys-con.com/node/492119?page=1

Gehlert, A., Pfeiffer, D., & Becker, J. (2007). *The BWW-Model as Method Engineering Theory.* Paper presented at the 13th Americas Conference on Information Systems (AmCIS 2007), Keystone, CO, USA.

Genovese, Y., Comport, J., & Hayward, S. (2006). *Person-to-Process Interaction Emerges as the "Process of Me"*.

Giaglis, G. M. (2001). A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques, *International Journal of Flexible Manufacturing Systems* (Vol. 13, pp. 209-228): Springer Netherlands.

Gil, Y. (2006). Workflow Composition. In D. Gannon, E. Deelman, M. Shields & I. Taylor (Eds.), *Workflows for e-Science*: Springer Verlag.

González-Cabero, R., Lecue, F., & Villa, M. (2008). *D6.4.1 Specification and First Prototype of Service Composition and Adaptation Environment*.

Gorronogoitia, Y., Radzimski, M., Lecue, F., Villa, M., & Di Matteo, G. (2010). *D6.4.2 Advanced Prototype For Service Composition And Adaptation Environment*.

Green, P., & Rosemann, M. (2000). Integrated Process Modelling: An ontological evaluation. *Information Systems, 25*, 73-87.

Green, P., Rosemann, M., Indulska, M., & Manning, C. (2007). Candidate Interoperability Standards: An Ontological Overlap Analysis. *Data & Knowledge Engineering, 62*(2), 274-291.

Greenfield, J. (2004). Software Factories: Assembling Applications with Patterns, Models, Frameworks and Tools. In G. Karsai & E. Visser (Eds.), *Generative*

Programming and Component Engineering (Vol. 3286, pp. 403-482): Springer Berlin / Heidelberg.

Greiffenberg, S. (2003). *Methoden als Theorien der Wirtschaftsinformatik.* Paper presented at the 6th International Conference Wirtschaftsinformatik, Dresden.

Group, W. C. W. S. A. W. (2004). *Web Services Architecture*, from http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

Group, W. C. W. S. A. W. (2004). *Web Services Glossary*

Gruber, T. R. (1992). *A Translation Approach to Portable Ontology Specifications*. Stanford, California: Knowledge Systems Laboratory.

Gschwind, T., Koehler, J., & Wong, J. (2008). Applying Patterns during Business Process Modeling. In *Proceedings of the 6th International Conference on Business Process Management* (pp. 4-19). Milan, Italy: Springer-Verlag.

Habbel, F.-R. (2008). *EU Service Directive: a challenge for Paneuropean Government*.

Hagemeyer, J., Hermann, T., Just, K., & Rüdiger, S. (1997). Flexibilität bei Workflow-Management-Systemen. *Software-Ergonomie, '97*, 179-190.

Hammer, M. (1990). Re-engineering work: don't automate, obliterate. *Harvard Business Review*(July-August), 104-112.

Hammer, M., & Champy, J. (1993). *Re-engineering the Corporation: A Manifesto for Business Revolution*. London: Nicolas Brearly.

Hammer, M., & Champy, J. (1994). *Business reengineering - Die Radikalkur für das Unternehmen* (2. Auflage ed.). Frankfurt am Main: Campus.

Hammer, M., & Stanton, S. (1999). How process enterprise really work. *Harvard Business Review*(November-December), 108-118.

Han, Y. (1997). *Software Infrastructure for Configurable Workflow Systems.* Unpublished PhD Thesis, Techical University of Berlin, Berlin.

Harrington, H. J. (1991). *Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity and Competitiveness*. New York: McGraw-Hill.

Havey, M. (2005). *Essential Business Process Modeling*: O'Reilly Media, Inc.

Hevner, A., March, S., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly, 28*(1), 75-105.

High, R. J., Kinder, S., & Graham, S. (2005). *IBM's SOA Foundation - An Architectural Introduction and Overview*.

Hill, C., Yates, R., Jones, C., & Kogan, S. L. (2006). Beyond predictable workflows: enhancing productivity in artful business processes. *IBM Syst. J., 45*(4), 663-682.

Hill, J. B., Cantara, M., Kerremans, M., & Plummer, D. C. (2009). *Magic Quadrant for Business Process Management Suites*.

Hoffmann, W., Kirsch, J., & Scheer, A.-W. (1993). Modellierung mit Ereignisgesteuerten Prozessketten. *IWi Heft, 101*.

Hollingsworth, D. (1995). *The Workflow Reference Model*.

Hornung, T., Koschmider, A., & Lausen, G. (2008). Recommendation Based Process Modeling Support: Method and User Experience. In *Proceedings of the 27th International Conference on Conceptual Modeling* (pp. 265-278). Barcelona, Spain: Springer-Verlag.

Hoyer, V., Janner, T., Delchev, I., Fuchsloch, A., L\&\#243;pez, J., Ortega, S., et al. (2009). The FAST Platform: An Open and Semantically-Enriched Platform for Designing Multi-channel and Enterprise-Class Gadgets

10.1007/978-3-642-10383-4_21. In *Proceedings of the 7th International Joint Conference on Service-Oriented Computing* (pp. 316-330). Stockholm: Springer-Verlag.

Hoyer, V., & Stanoevska-Slabeva, K. (2009). *Towards a Reference Model for Grassroots Enterprise Mashup Environments.* Paper presented at the 17th European Conference on Information Systems (ECIS 2009), Verona, Italy.

IRS. (2010). *Internet Reasoning Service III.* Retrieved 29.12.2010, from http://technologies.kmi.open.ac.uk/irs/#irsiii

Jablonski, S., Böhm, M., & Schulze, W. (1997). *Workflow Management - Entwicklung von Anwendungen und Systemen*. Heidelberg: dpunkt.

Jablonski, S., & Bussler, C. (1996). *Workflow management : modeling concepts, architecture and implementation*. London ; Sydney :: International Thomson Computer Press.

Jarczyk, A. P. J., Loeffler, P., & Iii, F. M. S. (1992). Design Rationale for Software Engineering: A Survey: Press L2.

Keller, G., Nüttgens, M., & Scheer, A.-W. (1992). Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)". *IWI-Heft, 89*.

Keller, U., Lara, R., Pollers, A., Toma, I., Kifer, M., & Fensel, D. (2004). *WSMO Web Service Discovery*.

Kim, J., Suh, W., & Lee, H. (2002). Document-based workflow modeling: a case-based reasoning approach. *Expert Systems with Applications, 23*(2), 77-93.

Kiper, J. D., Howard, E., & Ames, C. (1997). Criteria for Evaluation of Visual Programming Languages. *Journal of Visual Languages & Computing, 8*(2), 175-192.

Kircher, M. (2007, 2007/06/27). Guest Editors' Introduction: Software Patterns. *IEEE Software, 24,* 28-30.

Kittur, A., Suh, B., & Chi, E. H. (2008). Can you ever trust a wiki?: impacting perceived trustworthiness in wikipedia. In *Proceedings of the ACM 2008 conference on Computer supported cooperative work* (pp. 477-480). San Diego, CA, USA: ACM.

Klein, R., F., K., & A.-W., S. (2004). Modellierung inter-organisationaler Prozesse mit Ereignisgesteuerten Prozessketten. *IWI Heft, 178*.

Ko, R., Lee, S., & Lee, E. (2009). Business process management (BPM) standards: a survey. *Business Process Management Journal, 15*(5), 744-791.

Koehler, J., & Vanhatalo, J. (2007). Process anti-patterns: How to avoid the common traps of business process modelling. *IBM WebSphere Developer Technical Journal*.

Kopecky, J., Moran, M., Vitvar, T., Roman, D., & Mocan, A. (2007). *D24.2v0.1 WSMO Grounding*.

Kopecky, J., Vitvar, T., Fensel, D., & Gomadam, K. (2009). *hRests & MicroWSMO*.

Krummenacher, R., Domingue, J., Pedrinaci, C., & Simperl, E. (2010). SOA4All: Towards a global service delivery platform. In F. I. Assembly (Ed.), *Future Internet*.

Kueng, P., & Kawalek, P. (1997). Goal-based business process models: creation and evaluation. *Business Process Management Journal, 3*(1), 17-38.

Kurpjuweit, S. (2009). *Stakeholder-orientierte Modellierung und Analyse der Unternehmensarchitektur unter besonderer Berücksichtigung der Geschäfts- und IT-Architektur*. Unpublished Doctoral Thesis, University of St. Gallen (HSG), St. Gallen.

Lampert, D., & Pedrinaci, C. (2010). *WSMO-Lite extras: The minimal service model and service templates*: STI Innsbruck.

Lanker, E. (2008). *Services and processes at University of St. Gallen*.Unpublished manuscript, St. Gallen.

Le Clair, C., & Teubner, C. (2007). *The Forrester Wave: Business Process Management for document processes*.

Lechner, U., Schmid, B. F., Schubert, P., & Zimmermann, H.-D. (1998). Die Bedeutung von Business Communities für das Management der neuen Geschäftsmedien. In M. Englien & K. Bender (Eds.), *Gemeinschaften in neuen Medien (GeNeMe 98)* (pp. 203-219).

Lecue, F., Salibi, S., Bron, P., & Moreau, A. (2008). Semantic and Syntactic Data Flow in Web Service Composition. In *Proceedings of the 2008 IEEE International Conference on Web Services* (pp. 211-218): IEEE Computer Society.

Lee, R. G., & Dale, B. G. (1998). Business process management: a review and evaluation. *Business Process Management Journal, 4*(3), 214-225.

Leganza, G. (2006). *Why is SOA hot in Government*.

Lenat, D. (1998). *The Dimensions of Context-Space*.

Lieberman, H., Paterno, F., & Wulf, V. (2006). *End-User Development*: Springer.

Lillehagen, F., & Krogstie, J. (2008). *Active Knowledge Modeling of Enterprises*: Springer Publishing Company, Incorporated.

Linden, T. A. (1991). Representing software designs as partially developed plans. In M. Lowry & R. McCartney (Eds.), *Automating Software Design* (pp. 603-625): AAAI Press / The MIT Press.

Lo, A., & Yu, E. (2008). From Business Models to Service-Oriented Design: A Reference Catalog Approach. In *Conceptual Modeling - ER 2007* (pp. 87-101).

Lombardi-BPM. (2010). *IBM Lombardi BPM*. Retrieved 12/12/2010, from http://www.lombardisoftware.com/

Maamar, Z., Benslimane, D., & Narendra, N. C. (2006). What can context do for web services? *Commun. ACM, 49*(12), 98-103.

MacCallum, K. J., & Yu, B. (1996). Modelling of Product Configuration Design and Management by Using Product Structure Knowledge. In *Knowledge intensive CAD*. London: Chapman & Hall.

MacLean, A., Carter, K., Loevstrand, L., & Moran, T. (1990). User-tailorable systems: pressing the issues with buttons. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people* (pp. 175-182). Seattle, Washington, United States: ACM.

Madhusudan, T., Zhao, J. L., & Marshall, B. (2004). A case-based reasoning framework for workflow model management. *Data Knowl. Eng., 50*(1), 87-115.

Maleshkova, M., Pedrinaci, C., & Domingue, J. (2010). *Semantic Annotation of Web APIs with SWEET*. Paper presented at the 6th Workshop on Scripting and

Development for the Semantic Web at Extended Semantic Web Conference (ESWC2010), Heraklion, Greece.

Malone, T. W., Crowston, K., & Herman, G. A. (2003). *Organizing Business Knowledge: The MIT Process Handbook*: MIT Press.

Malone, T. W., Lai, K.-Y., & Fry, C. (1992). Experiments with Oval: a radically tailorable tool for cooperative work. In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work* (pp. 289-297). Toronto, Ontario, Canada: ACM.

March, S. T., & Smith, G. F. (1995). Design and Natural Science Research on Information Technology. *Decision Support Systems, 15*.

Martin, J. (1982). *Application Development without Programmers*: Prentice Hall PTR.

McBride, N., & Wood-Harper, A. T. (2002). Towards User-Oriented control of End-User Computing in Large Organisations. *Journal of End user Computing*.

McLean, E. R. (1979). End users as application developers. *MIS Quarterly, 3*(4), 37-46.

Medicke, J., & McDavid, D. (2004). Patterns for Business Process Modelling. *Business Integration Journal, 1*, 32-35.

Meehan, P. (2010). *The Business Units CIO's 2010 Agenda*.

Mehandjiev, N. (2008, 2008/09/15). *End-User Development for task management: Survey of attitudes and practices*. Paper presented at the Visual Languages - Human Centric Computing.

Mehandjiev, N., Sutcliffe, A. G., & Lee, D. (2006). Organisational view of end-user development. In H. Lieberman, F. Paterno & V. Wulf (Eds.), *End User Development* (Vol. 9, pp. 371-393): Human-Computer.

Mendling, J., Lassen, K. B., & Zdun, U. (2005). Transformation Strategies between Block-Oriented and Graph-Oriented Process Modelling Languages: GITO-Verlag.

Mendling, J., & Recker, J. (2008). *Towards Systematic Usage of Labels and Icons in Business Process Models*. Paper presented at the CAiSE 2008 Workshop Proceedings - Twelfth International Workshop on Exploring Modeling Methods in Systems Analysis and Design (EMMSAD 2008).

Mens, T., Czarnecki, K., & Gorp, P. V. (2005). *A taxonomy of model transformations*. Paper presented at the Language Engineering for Model-Driven Software Development - Dagstuhl Seminar, Dagstuhl, Germany.

MIT. (1993). *Watch what I do: programming by demonstration*: MIT Press.

Mittal, S., & Frayman, F. (1989). *Towards a generic model of configuration tasks*. Paper presented at the Eleventh International Joint Conference on AI.

Moody, D., & Shanks, G. (1994). What makes a good data model? Evaluating the quality of entity relationship models, *Entity-Relationship Approach â€" ER '94 Business Modelling and Re-Engineering* (pp. 94-111).

Morch, A., & Mehandjiev, N. (2000). Tailoring as Collaboration: The Mediating Role of Multiple Representations and Application Units. *Computer Supported Cooperative Work, 9*(1), 75-100.

Motta, E. (1999). *Reusable Components for Knowledge Modelling: Case Studies in Parametric Design Problem Solving*: IOS Press.

Namoune, A., Mehandjiev, N., Lecue, F., Wajid, U., Macaulay, L., & Álvaro Rey, G. (2009). *D2.5.1 Formative Evaluation and User-Centred Design*.

Namoune, A., Wajid, U., & Mehandjiev, N. (2009). *Composition of interactive service-based applications by end-users*. Paper presented at the UGS 2009 - 1st International Workshop on User-generated Services.

Nardi, B. A. (1993). *A small matter of programming: perspectives on end user computing*: MIT Press.

Neiger, D., & Churilov, L. (2004a). Goal-oriented business process engineering revisited: a unifying perspective. In *Computer Supported Activity Coordinatin* (pp. 149-163): INSTICC Press.

Neiger, D., & Churilov, L. (2004b). Goal-Oriented Business Process Modeling with EPCs and Value-Focused Thinking, *Business Process Management* (pp. 98-115).

Nielsen, J. (1990). Paper versus computer implementations as mockup scenarios for heuristic evaluation. In *Proceedings of the IFIP TC13 Third Interational Conference on Human-Computer Interaction* (pp. 315-320): North-Holland Publishing Co.

Nielsen, J. (1993). *Usability Engineering*: Morgan Kaufmann Publishers Inc.

Nitzsche, J., & Norton, B. (2009). Ontology-Based Data Mediation in BPEL (For Semantic Web Services), *Business Process Management Workshops* (pp. 523-534).

Nitzsche, J., van Lessen, T., Karastoyanova, D., & Leymann, F. (2007a). *BPEL for Semantic Web Services (BPEL4SWS).* Paper presented at the On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops.

Nitzsche, J., van Lessen, T., Karastoyanova, D., & Leymann, F. (2007b). *BPEL-Light.* Paper presented at the 5th International Conference on Business Process Management (BPM 2007).

Norton, D., Blechar, M., & Jones, T. (2010). *Magic Quadrant for Business Process Analysis Tools*.

Nunamaker, J. F., Chen, M., & Purdin, T. D. M. (1991). Systems development in information systems research. *Journal of Management Information Systems, 7*(3), 89-106.

OASIS. (2001). *ebXML Technical Architecture Specification v1.0.4*: OASIS.

OASIS. (2005). *Reference Model for Service Oriented Architectures*.

OASIS. (2006). *Web Services Business Process Execution Language Version 2.0*, from http://www.oasis-open.org/committees/download.php/18714/wsbpel-specification-draft-May17.htm

OMG. (2004). *Business Process Definition Metamodel* (Vol. Version 1.0.2): OMG.

OMG. (2006). *Business Process Modeling Notation Specification, OMG Final Adopted Specification dtc/06-02-01*. Retrieved 27.04.2006, from http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf

Oppenheim, A. N. (1992). *Questionnaire design, interviewing and attitude measurement*: Pinter Pub.

Ouyang, C., Dumas, M., ter Hofstede, A. H. M., & van der Aalst, W. M. P. (2007). Pattern-based translation of BPMN process models to BPEL web services.

Padovitz, A., Loke, S. W., & Zaslavsky, A. (2004). Towards a Theory of Context Spaces. In *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops* (pp. 38): IEEE Computer Society.

Paige, R., Ostroff, J. S., & Brooke, P. J. (2000). Principles for Modeling Language Design. *Information and Software Technology, 42*(10), 665-675.

Palmer, N. (2009). *2009 BPM State of the Market Report*.

Pankratius, V., & Stucky, W. (2005). *A formal foundation for workflow composition, workflow view definition, and workflow normalization based on petri nets.* Paper presented at the 2nd Asia-Pacific conference on Conceptual modelling, Newcastle, New South Wales, Australia.

Pavlov, G., Genevski, P., Vogel, J., Abels, S., Puram, S., Blood, S., et al. (2010). *D2.6.2 SOA4All Composer First Prototype*.

Pedrinaci, C., Grenon, P., Galizia, S., Gugliotta, A., & Domingue, J. (2010). A Knowledge-Based Framework for Web Service Adaptation to Context. In M. Sheng, J. Yu & S. Dustdar (Eds.), *Enabling Context-Aware Web Services: Methods, Architectures, and Technologies*: Chapman and Hall/CRC.

Pedrinaci, C., & Krummenacher, R. (2009). *Goals in SOA4All.*

Peffers, K., Tuunanen, T., Rothenberger, M., & Chatterjee, S. (2008). A Design Science Research Methodology for Information Systems Research. *J. Manage. Inf. Syst., 24*(3), 45-77.

Peterson, J. L. (1981). *Petri Net Theory and the Modeling of Systems*: Prentice Hall PTR.

Petri, C. (1962). *Kommunikation mit Automaten.* Institut fÃ¼r instrumentelle Mathematik.

PICTURE. (2007). *Report on Current Practice of Process Modelling Projects and Technicques in European Public Administrations*: EU FP6 project PICTURE.

Plummer, D. (2005). *Defining 'Service' is Key to Implementing a Service-Oriented Architecture*.

Plummer, D., & Hill, J. (2009). *Composition and BPM Will Change the Game for Business System Design*.

Powell, A., & Moore, J. E. (2002). The Focus of Research in End User Computing: Where have we come since the 80ties? *Journal of End user Computing, 14*(1), 3-22.

Quinn, K. (2005). *Not everyone who drives a car fixes it themselves - Strategic Information Infrastructure*. Retrieved 22/03/2010, from http://www.dmreview.com/news/1041222-1.html

Rao, J., & Su, X. (2005). A Survey of Automated Web Service Composition Methods. In J. Cardoso & A. Sheth (Eds.), *Semantic Web Services and Web Process Composition* (Vol. 3387, pp. 43-54): Springer Berlin / Heidelberg.

Recker, J. (2008). BPMN Modeling – Who, Where, How and Why. *BPTrends, 5*(3), 1-8.

Recker, J., & Dreiling, A. (2007). *Does it matter which process modelling language we teach or use? An experimental study on understanding process modelling languages without formal education.* Paper presented at the 18th Australasian Conference on Information Systems, Toowoomba, Australia.

Recker, J., Indulska, M., Rosemann, M., & Green, P. (2008). *An Exploratory Study of Process Modelling Practice with BPMN.*

Recker, J., & Mendling, J. (2006). *On the Translation between BPMN and BPEL: Conceptual Mismatch between Process Modeling Languages.* Paper presented at the Workshops and Doctoral Consortium for the 18th International Conference on Advanced Information Systems Engineering.

Recker, J. C., Indulska, M., Rosemann, M., & Green, P. (2005, 2005/01/01). *Do Process Modelling Techniques Get Better? A Comparative Ontological Analysis of BPMN.*

Regev, G., Bider, I., & Wegmann, A. (2007). Defining business process flexibility with the help of invariants. *Software Process: Improvement and Practice, 12*(1), 65-79.

Remme, M. (1997). *Konstruktion von Geschäftsprozessen - ein modellgestützter Ansatz durch Montage generischer Prozesspartikel*. Wiesbaden: Gabler.

Richardson, C., Moore, C., & Nicolson, N. (2009). *Vendor Snapshot: Lombardi Blueprint Bridges Gap Between Process Discovery And Execution*.

Richter-von Hagen, C., & Stucky, W. (2004). *Business-Process- und Workflow-Management, Prozessverbesserung durch Prozess-Management* (Vol. 1): Teubner.

Riehle, D., & Zuellighoven, H. (1996). Understanding and using patterns in software development. *Theor. Pract. Object Syst., 2*(1), 3-13.

Rieper, B. (1992). *Betriebswirtschaftliche Entscheidungsmodelle: Grundlagen*. Herne: Neue Wirtschafts-Briefe.

Ripa, G., De Giorgio, T., Gorronogoitia, Y., Mos, A., & Ravoajanahary, F. (2010). *D6.5.2 Advanced Prototype For Adaptive Service Composition Execution*.

Ripa, G., Zuccala, M., & Mos, A. (2009). *D6.5.1 Specification and first prototype of the composition framework*.

Riss, U. V., Rickayzen, A., Maus, H., & van der Aalst, W. M. P. (2005). Challenges for Business Process and Task Management. *Journal of Universal Knowledge Management, 0*(2), 77-100.

Rolland, C. (1998). A Comprehensive View of Process Engineering. In *Proceedings of the 10th International Conference on Advanced Information Systems Engineering* (pp. 1-24): Springer-Verlag.

Rolland, C., Kaabi, R., & Kraiem, N. (2007). On ISOA: Intentional Services Oriented Architecture, *Advanced Information Systems Engineering* (pp. 158-172).

Rolland, C., & Kaabi, R.-S. (2007). An Intentional Perspective to Service Modeling and Discovery. In *Proceedings of the 31st Annual International Computer Software and Applications Conference - Volume 02* (pp. 455-460): IEEE Computer Society.

Roman, D., Lausen, H., & Keller, U. (2006). *Web Service Modelling Ontology (WSMO).*

Rosemann, M., & Recker, J. (2006). *Context-aware process design: Exploring the extrinsic drivers for process flexibility.* Paper presented at the 18th International Conference on Advanced Information Systems Engineering, Luxembourg.

Rosemann, M., Recker, J., & Flender, C. (2008). Contextualisation of business processes. *International Journal of Business Process Integration and Management (IJBPMIM), 3*(1), 47-60.

Rosemann, M., Sedera, W., & Sedera, D. (2001). *Testing a Framework for the Quality of Process Models – A Case Study.* Paper presented at the PACIS2001.

Roser, S., Lautenbacher, F., & Bauer, B. (2007). *Generation of workflow code from DSMs*. Paper presented at the 7th OOPSLA Workshop on Domain-Specific Modeling.

RosettaNet. (2010). *Partner Interface Processes (PIPs)*, from http://www.rosettanet.org/Standards/RosettaNetStandards/PIPs/tabid/475/Default.aspx

Rosser, B. (2008). *Taking Advantage of User-Friendly Business Process Modelling*.

Rossi, M., & Sein, M. K. (2003). *Design Research Workshop - A Proactive Research Approach*. Retrieved 31/03/2010, from http://www.cis.gsu.edu/~emonod/epistemology/Sein%20and%20Rossi%20-%20design%20research%20-%20IRIS.pdf

Ruh, W. A., Maginnis, F. X., & Brown, W. J. (2001). *Enterprise Application Integration: A Wiley Tech Brief*: John Wiley and Sons Inc.

Russell, N., Arthur, van der Aalst, W., & Mulyar, N. (2006). *Workflow Control-Flow Patterns: A Revised View*.

Russell, N., ter Hofstede, A., Edmond, D., & van der Aalst, W. (2005). Workflow Data Patterns: Identification, Representation and Tool Support, *Conceptual Modeling â€“ ER 2005* (pp. 353-368).

Russell, N., ter Hofstede, A. H. M., Edmond, D., & Aalst, W. M. P. v. d. (2004). *Workflow Data Patterns*.

Russell, S. J., & Norvig, P. (2003). *Artificial Intelligence: A modern approach*.

Sadiq, S., Sadiq, W., & Orlowska, M. (2005). A Framework for Constraint Specification and Validation in Flexible Workflows. *Information Systems, 30*(5), 349-378.

Saidani, O., & Nurcan, S. (2007). *Towards context-aware business process modelling.* Paper presented at the 8th Workshop on Business Processes and Support Systems: Requirements for flexibility and the ways to achieve it.

Scheer, A.-W. (1998). *ARIS - Vom Geschäftsprozess zum Anwendungssystem*. Saarbrücken: Springer-Verlag.

Scheer, A.-W. (2001). *ARIS - Modellierungsmethoden, Metamodelle, Anwendungen*. Heidelberg, Berlin: Springer-Verlag.

Schmelzer, R. (2006). *Rich Internet Applications - Market trends and approaches*.

Schnabel, F., Born, M., Xu, L., González-Cabero, R., Lecue, F., & Mehandjiev, N. (2009). *D6.3.1 First Specification of Lightweight Process Modelling Language*.

Schnabel, F., Xu, L., Born, M., Gorronogoitia, Y., Lecue, F., Ripa, G., et al. (2009). *D6.3.2. Advanced Specification Of Lightweight, Context-aware Process Modelling Language*.

Scholz-Reiter, B., & Stickel, E. (1996). *Business Process Modelling*: Springer-Verlag New York.

Schütte, R. (1997). *Die neuen Grundsätze ordnungsmässiger Modellierung.* Paper presented at the Forschungsforum '97, Leipzig.

Schütte, R. (1998). *Grundsätze ordnungsmässiger Referenzmodellierung: Konstruktion konfigurations- und anpassungsorientierter Modelle*. Wiesbaden: Gabler.

Silver, B. (2008). *More on BPMN-to-BPEL.* Retrieved 28/11/2010, from http://www.brsilver.com/2008/04/04/more-on-bpmn-to-bpel/

Silver, B. (2009). *BPMN Method and Style*: Cody-Cassidy Pre.

Simon, H. A. (1969). *The Sciences of the Artificial*: MIT Press.

Sinz, E. J. (1998). *Modellierung betrieblicher Informationssysteme - Gegenstand, Anforderungen und Lösungsansätze.* Paper presented at the Modellierung '98, GI-Workshops in Muenster, Muenster.

Smith, G. (2010). *BPM, the next stage of end user programming - Part II.* Retrieved 22/01/2010, from http://www.bpm.com/bpm-the-next-stage-of-end-user-programming-part-ii.html

SOA4All. (2010). *SOA4All - Enabling a Web of Billions of Services*. Retrieved 02.01.2011, from www.soa4all.eu

Spahn, M., Dörner, C., & Wulf, V. (2008). *End User Development: Approaches towards a flexible software design.* Paper presented at the 16th European Conference on Information Systems.

Stachowiak, H. (1973). *Allgemeine Modelltheorie*. Wien: Springer.

Staud, J. (2001). *Geschäftsprozessanalyse* (Vol. 2. überarbeitete und erweiterte Auflage): Springer Verlag.

Stein, S., & Ivanov, K. (2007). EPK nach BPEL Transformation als Voraussetzung fuer praktische Umsetzung einer SOA. In W.-G. Bleek, J. Raasch & H. Züllighoven (Eds.), *Software Engineering 2007* (Vol. 105). Hamburg, Germany: GI.

Stein, S., Kuehne, S., & Ivanov, K. (2009). Business to IT Transformations Revisited. In D. Ardagna, M. Mecella & J. Yang (Eds.), *Business Process Management Workshops* (Vol. 17, pp. 176-187): Springer Berlin Heidelberg.

Stoitsev, T., & Scheidl, S. (2008). An Architecture for End-User Driven Business Process Management. In *Proceedings of the 2008 12th International IEEE Enterprise Distributed Object Computing Conference* (pp. 53-62): IEEE Computer Society.

Stoitsev, T., Scheidl, S., Flentge, F., & Muehlhaeuser, M. (2008). From Personal Task Management to End-User Driven Business Process Modeling. In *Proceedings of the 6th International Conference on Business Process Management* (pp. 84-99). Milan, Italy: Springer-Verlag.

Stuhec, G., & Crawford, M. (2007). *Accelerate your Business Data Modeling and Integration Issues by CCTS Modeler Warp 10*.

Sutcliffe, A. (2005). Evaluating the costs and benefits of end-user development. In *Proceedings of the first workshop on End-user software engineering* (pp. 1-4). St. Louis, Missouri: ACM.

Sutcliffe, A. G., Lee, D., & Mehandjiev, N. (2003). Contributions, costs, and prospects for end-user development. *HCI International 2003, 2*.

Tan, P. S. (2007, 2007/06/27). *Context-enabled B2B Collaborations.* Paper presented at the Services Computing, IEEE International Conference on.

TheFreeDictionary. (2011). *Semantics*. Retrieved 30/07/2011, from http://www.thefreedictionary.com/semantics

Thomas, O. (2005). *Das Modellverständnis in der Wirtschaftsinformatik: Historie, Literaturanalyse und Begriffsexplikation*: Universität Saarbrücken, Institut für Wirtschaftsinformatik.

Tran, H. N., Coulette, B., & Dong, B. T. (2007). *Broadening the Use of Process Patterns for Modelling Processes.* Paper presented at the SEKE.

Ukelson, J. (2009). *Five ways to discover unstructured processes*. Retrieved 03.01.2011, from http://www.zdnetasia.com/techguide/techmanagement/0,39044902,62056996,00.htm

UN-CEFACT. (2003). *Core Component Technical Specification*.

van der Aalst, W. M. P., Aldred, L., Dumas, M., & ter Hofstede, A. H. M. (2004). *Design and Implementation of the YAWL system*.

van der Aalst, W. M. P., & Lassen, K. B. (2005). *Translating Workflow Nets to BPEL4WS*: BPM Center.

van der Aalst, W. M. P., & ter Hofstede, A. H. M. (2005). YAWL: Yet Another Workflow Language. *Information Systems, 30*(4), 245-275.

van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow Patterns. *Distributed and Parallel Databases, 14*(3), 5-51.

van der Aalst, W. M. P., ter Hofstede, A. H. M., & Weske, M. (2003). *Business Process Management: A Survey.* Paper presented at the 1st International Conference on Business Process Management.

van der Aalst, W. M. P., & van Hee, K. (2004). *Workflow Management*: MIT Press.

Visser, E. (2001). A Survey of Rewriting Strategies in Program Transformation Systems. *Electronic Notes in Theoretical Computer Science*

WRS 2001, 1st International Workshop on Reduction Strategies in Rewriting and Programming, 57, 109-143.

Vitvar, T., Kopecky, J., & Fensel, D. (2008). *WSMO-Lite: Lightweight Semantic Descriptions for Services on the Web*.

Vogel, J., Schnabel, F., Mehandjiev, N., Lombardi, J.-P., Lecue, F., & Wajid, U. (2009). *D7.2 Scenario Definition*.

Vogel, J., Tsvetkova, I., Meyer, S., Un, P., Pariente Lobo, T., Lucea, J., et al. (2010). *D7.6 End User Service Annotation and Context Description*.

W3C. (1999). *XML Path Language (XPath)*: W3C.

W3C. (2004a). *OWL Web Ontology Language*: W3C.

W3C. (2004b). *OWL-S: Semantic Markup for Web Services*.

W3C. (2004c). *RDF Primer*: W3C.

W3C. (2004d). *RDF Semantics*: W3C.

W3C. (2004e). *XML Schema Part 0: Primer Second Edition*.

W3C. (2007a). *Semantic Annotations for WSDL and XML Schema*: W3C.

W3C. (2007b). *XSL Transformations (XSLT) Version 2.0*.

W3C. (2008). *SPARQL Query Language for RDF*.

W3C. (2010). *XQery 1.0: An XML Query Language (Second Edition)*.

Wajid, U., Namoune, A., & Mehandjiev, N. (2010). *A Comparison of Three Service Composition Approaches for End Users.* Paper presented at the EUD4Services Workshop in conjunction with AVI 2010, Rome.

Wand, Y., & Weber, R. (1989). An ontological evaluation of systems analysis and design methods. In E. D. Falkenberg & P. Lindgreen (Eds.), *Information System Concepts: An Indepth Analysis* (pp. 79-107).

Wand, Y., & Weber, R. (1993). On the ontological expressiveness of information systems analysis and design grammars. *Information Systems, 3*(4), 217-237.

Weber, B., Reichert, M., & Rinderle-Ma, S. (2008). Change patterns and change support features - Enhancing flexibility in process-aware information systems. *Data Knowl. Eng., 66*(3), 438-466.

Wielinga, B., & Schreiber, G. (1997). Configuration-Design Problem Solving. *IEEE Expert: Intelligent Systems and Their Applications, 12*(2), 49-56.

Wielinga, B. J., Akkermans, J. M., & Schreiber, A. T. (1995). *A Formal Analysis of Parametric Design Problem Solving.* Paper presented at the 9th Banff Knowledge Acquisition Workshop (KAW-95).

Wiig, K. M. (2004). *People-focused knowledge management: how effective decision making leads to corporate success*: Elsevier Butterworth-Heinemann.

Williams, S. (1967). Business Process Modeling improves Administrative Control. In *Automation* (pp. 44-50).

Winter, R. (2003). Modelle, Techniken und Werkzeuge im Business Engineering. In *in: Business Engineering, Hubert Österle* (Vol. Auflage 2., pp. pp. 3- 18): Springer Verlag.

Wohed, P., van der Aalst, W. M. P., Dumas, M., & ter Hofstede, A. H. M. (2003). Analysis of Web Services Composition Languages: The Case of BPEL4WS. In *Proceedings of Conceptual Modelling*: Springer.

WSMO-Working-Group. (2008). *D16.1 v1.0 WSML Language Reference*: WSMO Working Group.

Wulf, V., & Jarke, M. (2004). The economics of end-user development. *Commun. ACM, 47*(9), 41-42.

Zairi, M. (1997). Business process management: a boundaryless approach to modern competitiveness. *Business Process Management Journal, 3*(1), 64-80.

Zaslavsky, A. (2002). *Adaptability and Interfaces: Key to efficient pervasive computing*. Providence.

Zelewski, S. (1996). Eignung von Petrinetzen für die Modellierung komplexer Realsysteme - Beurteilungskriterien. *Wirtschaftsinformatik, 38*(4), 369-381.

Ziemann, J., & Mendling, J. (2005). *EPC-Based Modelling of BPEL Processes: a Pragmatic Transformation Approach.* Paper presented at the International Conference on Modern Information Technology in the Innovation Processes of the Industrial Enterprises, Genova.

Zimmermann, O., Krogdahl, P., & Gee, C. (2004). *Elements of Service-oriented Analysis and Design*: IBM Developernetwork.

zur Muehlen, M. (2004). *Workflow-based Process Controlling: Foundation, Design and Application of workflow-driven Process Information Systems*. Berlin: Logos-Verlag.

zur Muehlen, M., & Recker, J. C. (2008). *How Much Language is Enough? Theoretical and Practical Use of the Business Process Modeling Notation*: Springer.

Curriculum Vitae

Persönliche Daten

Geburtsort	Schorndorf (Deutschland)
Nationalität	Deutsch

Arbeitserfahrung

Seit Aug 2010	Berater im Management Consulting bei der Swisscom ITS AG im Bereich Finance (ehemals COMIT AG).
Feb 2007 – Jul 2010	Forschungsassistent an der Universität St. Gallen (HSG), Institut für Medien- und Kommunikationsmanagement (=mcminstitute) und SAP Schweiz AG, SAP Research CEC St. Gallen
Apr 2006 – Dez 2006	Werksstudent an der Universität St. Gallen (HSG), Institut für Wirtschaftsinformatik.
Okt 2005 – Mär 2006	Praktikum bei der IBM Deutschland GmbH im Bereich Kundenzufriedenheitsanalyse.
Mär 2003 – Apr 2003	Werksstudent bei der Daimler AG im Bereich Geschäftsprozessmodellierung.
Aug 2002 – Okt 2002	Werksstudent bei der Daimler AG im Bereich Geschäftsprozessmodellierung.
Jul 2001 – Aug 2001	Einzelhandelsassistent im Kaufland Schorndorf.
Jul 2000 – Mai 2001	Zivildienst am Marienstift Schorndorf.

Ausbildung

Feb 2007 – Jul 2010	Doktorat an der Universität St. Gallen (HSG)
Sep 2004 – Jun 2005	Studium der Informatik am ENSIMAG (Grenoble, Frankreich) und Studium der Volkswirtschaft an der Université Pierre Mendes (Grenoble, Frankreich)
Sep 2001 – Apr 2007	Studium der Informationswirtschaft am Karlsruhe Institute of Technology (ehemals Universität Karlsruhe (TH)). Abschluss: Diplom-Informationswirt
1991 – 2000	Schulausbildung am Burggymnasium Schorndorf