Semantic Mediation between Loosely Coupled Information Models in Service-Oriented Architectures

vorgelegt von
Diplom-Informatiker
Nils Barnickel
aus Berlin

Von der Fakultät IV – Elektrotechnik und Informatik –
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
– Dr.-Ing. –

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Thomas Magedanz
Berichter:	Prof. Dr. Dr. h.c. Radu Popescu-Zeletin
Berichter:	Prof. Dr. Bernd Mahr
Berichter:	Prof. Dr. Mathias Weske

Tag der wissenschaftlichen Aussprache: 11. Mai 2011

Berlin 2011

D 83

Abstract

The last two decades have shown a major shift from stand-alone to networked information technology (IT) systems. Consequently, the effective and efficient achievement of interoperability is a key factor to enable seamless business process chains and networks across intra- and inter-organizational boundaries. Thereby, interoperability can be understood along three dimensions: technical, semantic and organizational interoperability.

While the concept of service-oriented architectures (SOA) and widely accepted Web service standards have benefited technical interoperability in recent years substantially, managing and integrating semantic differences in heterogeneous distributed environments remains critical and cost intensive. In order to preserve the precise meaning as data is moved from one IT system to another, explicit formal information models in terms of ontologies have evolved as the concept of choice from academia to first industry adoption. However, it has been recognized that the dominant approach of developing a common, globally shared ontology as an information model standard has turned out to be limited in real world cross-domain environments. Organizational boundaries with regard to consensus degree and the complexity deriving from inherent domain-specific differences in requirements force a coexistence of independently managed but however semantic interoperable information models.

In order to address this challenge, the guiding idea of this work is to transfer the principle of loose coupling to the semantic level. In particular, the goal of this thesis is to contribute to the reduction of complexity in semantic system integration by developing an effective and efficient approach for semantic interoperability in large-scale SOA landscapes based on semantic mediation between loosely coupled information models. Moreover, this work shows how emerging semantic technologies can contribute to the instantiation of this concept exploiting their capabilities to explicitly express semantics. The main contributions of this work are:

- A conceptual framework for semantic interoperability in SOA, which is mapped to an overview and evaluation of existing academic and industry-driven approaches pointing out shortcomings and fields for further advancements.
- A concept of semantic mediation between loosely coupled information models in SOA, which describes an information architecture design pattern that provides an optimized balance within the identified inherent trade-off between effectiveness and efficiency in achieving semantic interoperability in SOA. It includes a specification of loosely coupled information models in terms of key characteristics derived from the principle of loose coupling such as autonomy, flexible binding and encapsulation.
- An instantiating semantic mediation mechanism by means of description logic rule-based semantic bridges and self-contained domain ontologies exploiting capabilities such as polymorphism, facet analysis classification and declarative entity manipulation.
- A semantic mediation methodology and prototypical toolkit, which maps the developed concept and mechanism to the SOA life-cycle ranging from business process modeling, over service composition to runtime process execution, in order to provide a proof of concept.

The developed approach is evaluated based on a case study of an exemplary distributed organization. It is shown how the approach of semantic mediation between loosely coupled information models can be applied in practice and which benefits can be generated with regard to achieving effective and efficient semantic interoperability in large-scale SOA landscapes.

Zusammenfassung

Die Informationstechnologie (IT) der letzten zwei Jahrzehnte war durch eine zunehmende Entwicklung weg von eigenständigen hin zu vernetzten IT-Systemen geprägt. Vor diesem Hintergrund ergibt sich die Herausforderung, Interoperabilität möglichst effektiv und effizient zu erreichen, um nahtlose Geschäftsprozesse innerhalb und über Organisationsgrenzen hinweg zu ermöglichen. Interoperabilität kann dabei entlang von drei Dimensionen verstanden werden: technische, semantische und organisatorische Interoperabilität.

Während das Konzept der Service-orientierten Architekturen (SOA) und weit etablierte Web Service-Standards in den letzten Jahren wesentlich zum Erreichen von technischer Interoperabilität beigetragen haben, ist die semantische Integration in heterogenen verteilten Umgebungen weiterhin schwierig und kostenintensiv. Für den bedeutungskonsistenten Datenaustausch zwischen IT-Systemen haben sich explizite formale Informationsmodelle in Form von Ontologien als erfolgversprechendes Konzept in akademischen und ersten industriellen Bereichen herausgestellt. Allerdings hat sich gezeigt, dass der dominierende Ansatz basierend auf einer umfassenden gemeinsam zu nutzenden Ontologie als standardisiertes Informationsmodell in organisationsübergreifenden Szenarien nur begrenzt praktikabel ist. Organisatorische Grenzen mit Hinsicht auf Konsensfähigkeit und die Komplexität, die aus unterschiedlichen domänenspezifischen Anforderungen hervorgeht, erfordern eine Koexistenz von unabhängig zu verwaltenden jedoch semantisch interoperablen Informationsmodellen.

Um dieser Herausforderung zu begegnen, ist der Leitgedanke der vorliegenden Arbeit, das Prinzip der losen Kopplung auf die semantische Ebene zu übertragen. Dabei verfolgt die Arbeit das Ziel, einen Beitrag zur Verringerung der Komplexität bei der semantischen Systemintegration zu leisten. Im Zentrum steht die Entwicklung eines effektiven und effizienten Ansatzes für die semantische Interoperabilität in groß angelegten SOA-Landschaften mittels semantischer Mediation zwischen lose gekoppelten Informationsmodellen. Darüber hinaus zeigt die Arbeit, wie neuartige semantische Technologien verwendet werden können, um das entworfene Konzept zu instanziieren. Die wichtigsten Beiträge dieser Arbeit sind:

- Ein konzeptioneller Rahmen der semantischen Interoperabilität in SOA, der abgebildet wird auf einen Überblick existierender akademischer und industrieller Ansätze, mit dem Ziel Handlungsfelder und Entwicklungsbedarfe aufzuzeigen.
- Ein Konzept der semantischen Mediation zwischen lose gekoppelten Informationsmodellen in SOA als Entwurfsmuster für Informationsarchitekturen. Es beinhaltet eine Spezifikation auf Basis von wesentlichen Merkmalen des Prinzips der losen Kopplung wie Autonomie, flexible Bindung und Kapselung.
- Ein semantischer Mediationsmechanismus basierend auf regelbasierten semantischen Brücken und unabhängiger Ontologien unter Nutzung von Eigenschaften wie Polymorphismus, Facetten-basierte Klassifizierung und deklarativer Entitätenmanipulation.
- Ein Machbarkeitsnachweis auf Basis einer Methodik und prototypischer Werkzeuge zur semantischen Mediation, welche das entwickelte Konzept auf den SOA-Lebenszyklus abbilden und instanziieren mit dem Fokus auf der Geschäftsprozessmodellierung, der Servicekomposition und der laufzeitorientierten Prozessausführung.

Der entwickelte Ansatz wird anhand einer Fallstudie einer beispielhaften verteilten Organisation evaluiert. Es wird gezeigt, wie der Ansatz in der Praxis angewendet werden kann und welche Vorteile sich daraus für die effektive und effiziente Erreichung der semantischen Interoperabilität in groß angelegten SOA-Landschaften ergeben.

Preface

After finishing my studies, I made an internship at the United Nations Headquarters, where I attended a conference called Web for development. A marketing vice president from a large IT company gave a presentation on how service-oriented architectures (SOA) can accelerate development. After the talk a question came from the audience asking to further elaborate on how SOA can foster development in Africa. This misunderstanding has shown me that semantic interoperability - or the absence of it – is not only an abstract concept but can be found all around us even though often not visible and identified as such. Another example was the organization-wide knowledge management system, which could not be adopted in the eGovernance department I was working for, because the general terms and categories did not match the required differentiation and perspective for this practice area.

These practical experiences and the unexploited potentials for organizational synergies through seamless IT integration have motivated me to undertake my research on semantic mediation, when I started to work at the eGovernment competence center of the Fraunhofer Institute for Open Communication Systems (FOKUS).

Doing a PhD is an endeavor with many challenges. Especially in a dynamic environment driven by client-orientation, it is sometimes hard to find the time besides all the daily project work. However, it is just this combination at FOKUS covering theoretical research and real-world client projects, which provides a unique opportunity to understand the multiple dimensions and challenges of IT integration in cross-organizational contexts, for which I am very grateful.

In particular, I want to thank my two PhD supervisors: Prof. Dr. Radu Popescu-Zeletin for the discussions, his encouragement and practical advice at key points of my PhD project and Prof. Dr. Bernd Mahr for his conceptual advice and motivating feedback. I also express my gratitude to Prof. Dr. Mathias Weske for serving as the external reviewer of my dissertation and for his helpful comments. Furthermore, I want to thank my department head Gerd Schürmann for giving me the freedom of a home office Friday in the second phase of the PhD. A big thanks goes to my colleague and office mate Dr. Matthias Flügge for sharing his experiences and for the discussions and feedbacks especially during paper publication. I also want to thank Prof. Dr. Adrian Paschke for the joint work we have done for the iSemantics and European Semantic Web Conference in 2010. Additionally, I thank my students, especially Ralf Weinand, Elena Antonenko and Johannes Böttcher for supporting the development of the semantic mediation toolkit.

To close the cycle in this personal preface I am referring back to my internship with the United Nations. The second part of it brought me to Dakar in Senegal, West Africa to support the launch of a Web community platform for eGovernance practitioners in the region. During that time I met my wonderful wife Anta. The internship was over and I was working on semantic mediation of IT systems. But the same time I found myself heavily involved in semantic mediation with her, her family and relatives trying to bridge continents, languages and different cultures. Finally, I am happy and proud that I can write about this twofold semantic mediation success story and herewith dedicate this work to my wife Anta and our five month old son Junus.

Berlin, May 2011

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

The last two decades have shown a major shift from stand-alone to networked information technology (IT) systems. Today, networked IT systems based on the infrastructure of the World Wide Web provide the technological backbone of enterprise ecosystems enabling various business process chains and networks within and across organizational borders. Consequently, the effectiveness and efficiency of integration of independent and distributed IT systems is of great practical importance, which can already be seen by the estimation that up to 40% of companies' IT budgets are spent on integration issues [1]. Considering historically grown and heterogeneous IT landscapes, the ability of organizations and their IT systems to work together – namely by ensuring interoperability – is the key factor for achieving seamless business processes. Consequently, cross-organizational interoperation of IT systems becomes a critical business success factor [2].

Interoperability can be understood along three dimensions: technical, semantic and organizational interoperability [3]. Although the concept of service-oriented architectures (SOA) [4] and widely accepted Web service standards [19] have benefited technical interoperability in recent years substantially, managing and integrating semantic differences in heterogeneous distributed environments remains critical and cost intensive [5]. In fact, case studies have shown that 60-80% of the resources of integration projects are spent on reconciling semantic heterogeneities [6].

To provide an example, a distributed organization such as the German Chamber of Industry and Commerce with 80 decentralized sites can be considered. The sites are operated by four different IT service providers resulting in a heterogeneous IT landscape. In order to establish organization-wide business processes, in particular existing historically grown information models of different providers need to be integrated. The semantic integration challenge further increases taking into account the various external business partners to be integrated in cross-organizational business processes.

In order to preserve the precise meaning as data is moved from one IT system to another, ontologies have evolved as the concept of choice from academia to first industry adoption. Ontologies provide the means for generating explicit formal information models of a domain that can be shared between applications. The description logic-based expressiveness of ontologies not only enables humans to develop, discuss and agree on shared conceptualizations but also enables machines to interpret these information models in a meaningful manner across different IT systems.

However, the dominant approach of developing one common ontology-based standard for information exchange, which has to be globally shared by all actors in a distributed IT ecosystem, has turned out to be limited in real world cross-organizational contexts. In practice, organizational boundaries and the complexity deriving from different requirements on information models hinder the overall commitment to one common conceptualization. Thus, ontology-based standards could only alleviate the problem of semantic heterogeneity and a mapping between ontologies originating from different contexts is needed [7]. Consequently, diverse SOA landscapes covering multiple independent organizations require a more flexible information architecture to achieve semantic consistency while allowing for and accepting different conceptualizations.

1.2 Overall Goals and Scope

The core concept of SOA is the decomposition of complex business processes into a composition of loosely coupled independently managed services providing distinct business functionalities. The guiding idea of this work is that the same principle of loosely coupled units can be applied to information models, in order to capture the complexity of semantics in distributed IT ecosystems.

According to this principle, the overall goal of this thesis is to contribute to the reduction of complexity in semantic system integration by analyzing, designing, instantiating and evaluating an information architecture pattern for large-scale SOA landscapes based on semantic mediation between loosely coupled information models.

The concept should take into account the realistic perspective of different conceptualizations and resulting information representations that need to evolve independently from each other to serve best for their domain**.** Therefore, the concept should allow for autonomous management of self-contained information models of independent business domains. Furthermore, the concept should target semantic interoperability on the level of domain models rather than addressing it recursively during process integration on the application level. In order to facilitate semantic consistency in cross-organizational SOA scenarios, these heterogeneous domain-specific information models should be interlinked in a loosely coupled manner by means of an effective and efficient semantic mediation mechanism. The mechanism has to provide a high level of expressiveness that enables to reconcile complex semantic heterogeneities between information representations from different domain models. And at the same time the mechanism should be easy to handle. Thus, declarative approaches should be favored in contrast to procedural ones in order to assure efficient maintainability.

Moreover, a technology instantiating the concept of semantic mediation should be developed. To reap the benefits of explicit semantic formalizations, it should be based on emerging Semantic Web technologies. In particular domain ontologies and description logic rules should be exploited to describe ontology mappings in terms of so called semantic bridges [8]. Semantic bridges provide declarative reasoning-based means which can be integrated in SOA scenarios for aligning heterogeneous information models and thus ensure semantic interoperability by remaining organizational independence. However, taking into account technological path dependency in SOA, already existing traditional XML-based Web service technology should be respected and therefore the approach should be realized as an additional semantic layer on top of existing technology.

Given the horizontal nature of semantic interoperability, implications of the approach of semantic mediation to the SOA life-cycle should be derived with a focus on cross-organizational aspects. Consequently, the approach should be applied to key steps of the SOA life-cycle from conceptual business process modeling, over service composition to runtime process execution. The therefore required technologies should be bundled in a semantic mediation toolkit, which finally should be evaluated in terms of a case study of an exemplarily distributed organization.

To summarize, the objectives of this work are to:

- provide problem awareness in terms of a framework of semantic interoperability in SOA used to analyze the state-of-the-art and outline open challenges;
- develop a concept for semantic mediation between loosely coupled information models in SOA;
- design a semantic mediation methodology that applies the approach to the SOA life-cycle;
- instantiate key steps of the methodology in terms of a semantic mediation toolkit;
- and evaluate the semantic mediation approach in terms of a case study of an exemplary distributed organization.

The identified overall goals and objectives are further refined in the following section covering the methodology of this work and its research hypothesis.

1.3 Methodology

1.3.1 Scientific Hypothesis and its Confirmation

Based on the above presented overall goals and objectives the scientific hypothesis of this work can be formulated as follows:

> *In order to effectively and efficiently achieve semantic interoperability in large-scale cross-organizational service-oriented architectures, the principle of loose coupling can be applied to information models based on a flexible semantic mediation mechanism using Semantic Web technology for autonomous management and integration of domain-specific information models in terms of self-contained ontologies.*

To confirm the hypothesis a systematic approach is followed. The research methodology is aligned to the approach of design research in information systems. Design research has its origin in engineering and sciences of the artificial [9]. The approach is motivated by improving the state-of-the-art in terms of solving practical problems, whereby the utility of the solutions is focused. In the context of the design paradigm, understanding and knowledge of the problem domain and its solution are achieved by construction and application of designed artifacts. Information systems artifacts are defined as constructs (vocabularies and symbols), models (abstractions and representations), methods (sequence of activities) and instantiations (implemented and prototype systems) [10]. The results of design research in information systems are useful artifacts built to address an organizational problem.

Corresponding to the basic steps in design research [11], the confirmation of the hypothesis is structured in five consecutive parts. For each general step in design research the concrete artifact produced in this work is further specified:

1. **Awareness of a problem – Framework of Semantic Interoperability in SOA and State-of-the-Art**

   This work addresses the problem of achieving semantic interoperability in large-scale cross-organizational service-oriented architectures. Therefore, based on literature review, definitions and models of semantic interoperability are analyzed including their context to other dimensions of interoperability such as technical and organizational interoperability. An aggregation of conceptual models for semantic interoperability is further specified to the focused domain of SOA. Consequently, a conceptual framework of semantic interoperability in SOA has to be derived to deepen the understanding and providing a consistent conceptualization of the problem area. The framework then is utilized as a reference point for comparison in the following chapters of this work. Furthermore, an analysis is given discussing advantages and limitations of state-of-the-art approaches and technologies for achieving semantic interoperability in SOA, whereas it is referred to the previously developed framework.

2. **Suggestion – Concept of Semantic Mediation between Loosely Coupled Information Models**

   The guiding idea of this work is to transfer the concept of loose coupling to the semantic level. In contrast to limitations of state-of-the-art approaches based on one common information model to be globally-shared as a lingua franca, this work develops a concept based on multiple coexisting information models. It aims to provide a flexible information architecture pattern that allows for autonomous management of distinct information models, whereas a semantic mediation mechanism provides loose coupling between them to ensure semantic interoperability. In particular, the claimed effectiveness and efficiency of the developed approach is addressed by a comparative analysis. The concept is further concretized by relating it to formal ontologies representing information models of specific domains. Furthermore, the concept introduces a description logic rules-based ontology mapping approach, in order to realize the semantic mediation between the heterogeneous domain ontologies.

3. **Development - Semantic Mediation Methodology for SOA Life-Cycle and Semantic Mediation Toolkit**

   By means of a connecting step between theory and experiment, the theoretical concept is mapped to the concrete application domain of SOA. A specific semantic mediation methodology is developed that determines the basic steps relevant for the application of the concept of semantic mediation to the SOA life-cycle. In order to instantiate key steps of the methodology and to provide an experimental confirmation, a prototypical toolkit based on Semantic Web technologies is designed and developed. The toolkit integrates existing components and services and is extended with key tools required for semantic mediation in the SOA life-cycle. In particular, the semantic mediation toolkit addresses the design of semantic bridges in terms of ontology mapping rules, their systematic testing, their integration into business process modeling, as well as into service composition and finally into runtime execution infrastructures.

4. **Evaluation - Case Study of an Exemplary Distributed Organization**

   The evaluation of the developed approach of semantic mediation is addressed from a practical perspective investigating its effectiveness and efficiency in comparison to state-of-the-art approaches. For this purpose the developed methodology and the toolkit is mapped to an exemplary distributed organization in terms of a case study. Thus, the potential of the semantic mediation concept is demonstrated. Finally, the originally set goals and the derived research hypothesis are recalled and discussed, in order to access how and to which extent they could be covered and whether the claims of the research hypothesis could be confirmed.

5. **Conclusion**

   The conclusion summarizes the before described steps and points out the main conceptual conclusions and scientific contributions in a condensed manner. Furthermore, remaining open issues are discussed and potential extensions and future work is outlined.

1.3.2 Research Questions and Technical Challenges

In the above outlined multi-step process for the confirmation of the hypothesis various challenges have to be overcome. In the following, the central research questions and technical challenges are outlined.

Challenges in Step 1: Framework of Semantic Interoperability in SOA and State-of-the-Art

- Semantic interoperability is an abstract concept, which frameworks about interoperability often do not clearly distinguish from related aspects such a syntactical or structural interoperability originating from a more technical perspective or with pragmatic interoperability leading to a more organizational perspective. The framework to be developed should differentiate between these aspects and define the scope of semantic interoperability as it is addressed in this work.
- The framework of semantic interoperability should be expressive enough to compare various approaches possibly following opposing concepts. The range should cover industry-based state-of-the-art approaches to academic-driven ontology-based ones on the one hand and as well approaches based on shared homogeneous information models following a semantic standardization approach to approaches accepting and focusing on heterogeneous conceptualizations on the other hand.

Challenges in Step 2: Concept of Semantic Mediation between Loosely Coupled Information Models

- It should be investigated why on the one hand, the success of widely accepted Web service standards for SOA has benefited technical interoperability substantially in recent years, but on the other hand, standardization on the semantic level has turned out to be limited in the cross-organizational context. Therefore, analogies from other fields of standardization should be derived, in order to examine the relation between consensus degree and adequate scope of standards and its implication for the semantic level.
- Furthermore, it should be investigated how context dependency of information models influences heterogeneous conceptualizations and how this relates to limiting factors for their monolithic alignment. Therefore, a model theoretic approach should be mapped to information models.
- The transfer of the concept of loose coupling to information models implies that the central principles of loose coupling are addressed. Therefore, the question should be addressed how principles such as autonomy, encapsulation and flexible binding can be applied to the

semantic level and how these characteristics can be interpreted to provide a specification of loosely coupled information models and a corresponding semantic mediation mechanism.

- Having identified the practical limitations of semantic standardization across organizational boundaries, a trade-off between effectiveness and efficiency for achieving semantic interoperability becomes apparent. On the one hand, N actors have to develop an agreement in terms of a community process about one common standardized information model. However, with regard to cross-organizational and heterogeneous IT landscapes with a large number of actors N with possibly divergent business requirements, a high coordination complexity appears hindering an effective solution. On the other hand, aiming at an approach which is based on direct mediation between each two independent information models just requires coordination efforts for two actors, which results in lower complexity. However, this effort could be potentially become necessary $N x N$ times to map between each two information models. Thus, both general approaches do not provide a sufficient solution regarding effectiveness on the one side and efficiency on the other side. Therefore, an adequate solution within this trade-off needs to be addressed by the developed concept of semantic mediation.
- As the developed concept for semantic mediation is designed to be based on Semantic Web concepts and technologies, it needs to be pointed out which specific features of Semantic Web languages and meta-models are the beneficial and enabling factors for the semantic mediation approach compared to other technologies.

Challenges in Step 3: Semantic Mediation Methodology for the SOA Life-Cycle and Semantic Mediation Toolkit

In order to develop the methodology and toolkit, the relevant phases of the SOA life-cycle where mediation between heterogeneous information models is required need to be identified and the afore-developed conceptual solution needs to be applied.

- The SOA life-cycle starts from the business perspective on how processes can be supported by IT systems. Therefore, with regard to semantic mediation the modeling of cross-organizational business processes should be covered, whereas the modeling of information flow across heterogeneous conceptualizations is of particular concern. In order to ease the modeling of business processes and reduce technical complexity, the heterogeneity between different information models should be transparent for the user and its resolution should be handled automatically based on underlying semantic bridges. This implies that required information models and semantic bridges are already in place. Furthermore, coming from the perspective of agile development and continuous maintenance, information models need to evolve over time and correspondingly semantic bridges between them. According to process-orientation, the requirements for the evolution should be derived from business processes. Consequently, specific features for requirement engineering of information models and semantic bridges should be supported during cross-organizational business process modeling.
- The identified requirements provide a foundation for the development and testing of semantic bridges. As first prototypical tools for the development of semantic mappings are already available, they can be exploited in an adequate manner, in order to define semantic bridges according to the requirements of the developed semantic mediation mechanism. Furthermore, taking into account that semantic bridge developers and users such as process experts or Web service composers are divided into different roles and may originate from different organizational contexts, the consideration of trust in the quality of the underlying semantic mappings is essential. Therefore, an approach and tool for testing of semantic

bridges should be provided. The focus should be put on how to apply concepts from software testing to testing of ontology mappings.

- Having all required assets such as business process models, information models and quality assessed semantic bridges at hand; the consequent next step of the SOA life-cycle is the composition of services to instantiate the business process. Therein, the explicit semantic description of information models and formalized semantic bridges between the involved heterogeneous information models should be exploited for seamless information flow design between the services to be composed. One particular challenge lies in the consideration of technological path dependency. On the one hand, the dominant instantiation of SOA is based on Web service technology, which relies on the XML and XML schema meta-data model. On the other hand, the meta-data model applied for the semantic mediation approach is based on ontology concepts and description logic based rules. Thus, a challenge is to integrate as well an appropriate mapping mechanism between these two meta-data models and realize the solution as an additional layer on top of existing technology.
- After design time, the runtime execution of Web service compositions takes the focus in the SOA life-cycle. Again, well established industry standards should be considered. On this regard especially the industry standard BPEL [12] should be addressed, which relies on the XML meta-data model, too. Therefore, components providing Semantic Web technology have to be incorporated into BPEL-based process integration middleware and the different meta-data models need to be reflected on the runtime level. Another challenge thereby lies in ensuring a reasonable performance during the rule-based inferencing process, which still often remains a bottleneck of Semantic Web technology.

Challenges in Step 4: Case Study of an Exemplary Distributed Organization

- The evaluation needs to address how the potential of the developed methodology and toolkit for loosely coupled domain-specific ontologies can be qualitatively analyzed and demonstrated. Therefore, a case study is carried out in context of a research transfer project with the German Chambers of Industry and Commerce. The Fraunhofer Institute for Open Communication Systems (FOKUS) supports the introduction of an SOA-based IT integration infrastructure to the German Chambers of Industry and Commerce consisting of 80 decentralized sites, which are operated by four different IT service providers. In particular, the activities of the data conference working group targeting the development and alignment of organization-wide information models and semantic integration with external business process partners in the larger eGovernment context are subject to the evaluation. In this process, shortcomings of applied state-of-the-art practices and technologies need to be pointed out and compared to the potential provided by the developed semantic mediation approach.

1.4 Outline of the Thesis

After having discussed the objectives and the methodology of the work, this section outlines the structure of the thesis. The thesis is organized in 8 chapters, which are derived straightforward from the applied methodology of design research as illustrated in the following figure:

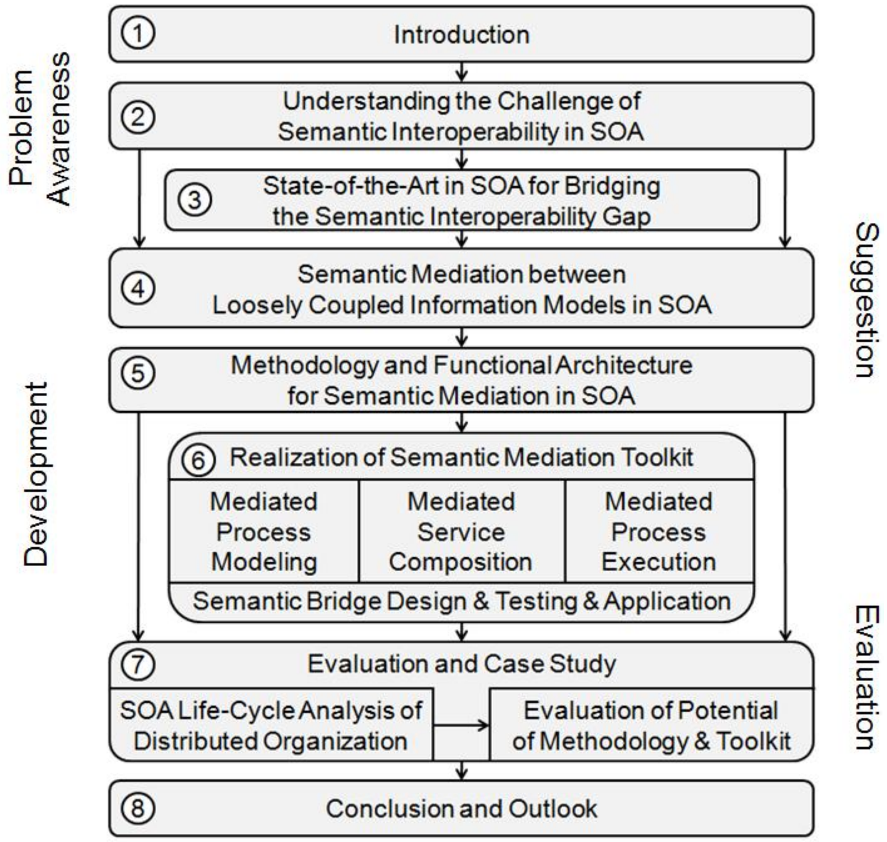


Figure 1-1 Thesis Structure

Chapter 1 gives the motivation and background of this work, its goals and scope and the research hypothesis and methodology structuring this thesis.

Chapter 2 provides an understanding of the challenge to achieve semantic interoperability in cross-organizational service-oriented architectures. Finally, a conceptual framework of semantic interoperability in SOA is elaborated, in order to provide a foundation for comparison in the following chapters.

Chapter 3 then performs a systematic state-of-the-art analysis of existing approaches. Conceptual ideas, technologies and standards for achieving semantic interoperability in SOA originating from different backgrounds ranging from industry to academia are presented and are evaluated against the before developed framework.

Chapter 4 presents the core part of this work namely the concept of semantic mediation between loosely coupled information models in SOA. The transfer of the principle of loose coupling to the semantic level is discussed based on a conceptual argumentation ranging from the limitations of semantic standardization to context dependency of information models and its consequences for semantic interoperability in cross-organizational SOA. Finally, a specification of loosely coupled information models in SOA is provided including a developed conceptual approach for a corresponding semantic mediation mechanism.

Chapter 5 provides an intermediate step between the developed theory and its application and presents a derived semantic mediation methodology. It maps the concept to the SOA life-cycle ranging from business process modeling, over service composition to runtime process execution. It describes how the semantic mediation approach can be integrated within these phases, in order to improve effectiveness and efficiency in achieving semantic interoperability.

Chapter 6 presents the developed semantic mediation toolkit, which instantiates key steps of the before developed methodology. It includes prototypical tools for mediated business process modeling, semantic bridge testing, mediated service composition and mediated process execution. Its realization by means of a combination of state-of-the-art Web service technologies and emerging description logic based Semantic Web technologies is described with regard to design and development aspects.

Chapter 7 evaluates the developed approach for semantic mediation. Based on a case study of an exemplarily distributed organization, namely the German Chambers of Industry and Commerce, the semantic mediation methodology and the potential of the developed toolkit are assessed. On the basis of this analysis, the coverage of the originally set conceptual goals and the confirmation of the research hypothesis are discussed.

The final Chapter 8 provides a conclusion of the thesis and recalls the fundamental concepts and ideas of the proposed approach for semantic mediation between loosely coupled information models in SOA. Moreover, remaining open issues and potential advancements are discussed. Finally, an outlook on future developments and priorities in this area is outlined.

Chapter 2

Understanding the Challenge of Semantic Interoperability in SOA

2.1 Overview

The introduction has already outlined that interoperability is the enabling factor to achieve IT-supported business processes across intra- and inter-organizational boundaries. The mentioned estimation that enterprises and organizations today spend up to 40% of their IT budget on integration projects [1] further points out the relevance and shows that interoperability has become a crucial competitive factor. Taking into account this background, it becomes comprehensible that the promise to advance interoperability has been a central success factor for the adoption of SOA. However, focusing on a particular dimension of interoperability, namely semantic interoperability, still substantive limitations have to be overcome as managing and integrating semantic differences in heterogeneous distributed environments remains critical and cost intensive [5]. As this work aims at advancing the way semantic interoperability in cross-organizational SOA is achieved, firstly the problem area of this particular interoperability dimension is analyzed.

This chapter begins by setting the scope of the addressed problem and putting semantic interoperability into the context of related dimensions of interoperability. Then, existing conceptual models dedicated to semantic interoperability are reviewed and interpreted from the perspective of SOA. Furthermore, in order to develop a framework for comparison of existing and emerging approaches, an aggregation of the analyzed conceptual models is derived. The derived framework should deepen the understanding and provide a consistent conceptualization of the problem area to be utilized as a reference point in the following chapters. Thereby, the framework is limited to a descriptive scope with a clear distinction to the description of a solution as targeted in the conceptual part of this work.

2.2 Interoperability Dimensions

In the context of the European Union's Information Society activities, interoperability is defined as the ability of information and communication technology systems and of the business processes they support to exchange data and to enable the sharing of information and knowledge [3]. In order to understand the nature of interoperability, it is important to note that interoperability is not a static property, which is provided or not, but rather a continuous degree which can be achieved to a lower or higher extent. According to this conception, a further definition states that interoperability is the ongoing process of ensuring that the systems, procedures and cultures of an organization are managed in such a way as to maximize opportunities for exchange and re-use of information, whether internally or externally [13].

Given such a wide scope within the suggested definitions, it becomes useful to further subdivide the notion of interoperability. The European interoperability framework distinguishes between three main interoperability dimensions [3]:

- technical interoperability, which is concerned with the technical issues of linking up computer systems, the definition of open interfaces and telecommunication protocols
- semantic interoperability, which is concerned with ensuring that the precise meaning of exchanged information is understandable by any other application not initially developed for this purpose
- organizational interoperability, which is concerned with modeling business processes, aligning information architectures with organizational goals and helping business processes to co-operate

Other frameworks about interoperability introduce further dimensions to be considered, such as the political context including cultural aspects or legal interoperability dealing with the alignment of heterogeneous legal environments that my hinder integration [14].

However, as this work addresses the dimension of semantic interoperability, deeper analysis should be given with a finer granularity on its context within the three interoperability dimensions presented above.

2.2.1 The Context of Semantic Interoperability

In [15] a changing focus on interoperability of information systems is discussed: from system, over syntax and structure to semantics. Thereby, several types of heterogeneity are identified with according types of interoperability:

- System - differences between hardware, operating systems, protocols etc.;
- Syntactic - incompatibilities in encodings and formats;
- Structural - differences in representation and schemata;
- Semantic - inconsistencies in terminology and meanings.

In this sense, system interoperability corresponds to the common understanding of technical interoperability as outlined above in context of the European interoperability framework. However, the above presented perspective identifies further aspects between technical and semantic interoperability. On the one hand, syntactic interoperability can be referred to the ability of different systems to interpret the syntax of data the same way, i.e. to share common rules how parts of data can be arranged together. In particular, this deals with technical aspects such as the alignment of common APIs, interchange formats and messaging standards. And on the other hand, structural interoperability can be identified, which refers to the ability to align different data representations based on differently structured schemata. Taking into account that schemata relate to specific domains or applications, it points out that the structure of data in terms of schemata captures partly – in terms of a limited view – the aspect of meaning. Thus, it can be stated that structural interoperability is closer related to semantic interoperability than syntactical interoperability, as syntax is generally more independent and generic from the specific domain or application context of IT systems.

However, semantic interoperability covers further aspects than discussed in context of structural interoperability. Data representation in terms of schemata cannot capture the entire meaning as they lack the description of context of data and explicit description of relations between data, which constitute fundamental aspects of meaning [16]. From the research area of linguistics a further aspect comes into the play and the before discussed field is broken down into the three branches [17]:

- Syntactics - relation of signs to each other in formal structures;
- Semantics - relation between signs and the things they refer to;
- Pragmatics - relation of signs to their impacts on those who use them.

With regard to interoperability, the aspect of pragmatics is reflected and referred to the pragmatic interoperability problem, which arises when the sender's intended effect of a message differs from the actual effect of the message performed by the receiver [18]. In SOA, this is the case when there is insufficient insight in the interworking of services and their interdependent behavior [283]. This problem can be overcome by means of languages that define so called service choreographies [284]. In [285] choreographies are defined as complex interactions with behavioral dependencies between the contained interactions. Consequently, this aspect on how information is processed depending on its dynamic or behavioral context, points out the relation to the dimension of organizational interoperability with its particular focus on business process alignment between different actors as identified in the European interoperability framework.

This section has analyzed the different dimensions of interoperability and positioned semantic interoperability within these dimensions. With regard to technical interoperability, the aspect of structural interoperability in terms of mismatching schemata has been identified as partly overlapping with semantic interoperability. On the other hand, the aspect of pragmatic interoperability has been identified as bridging the gap between semantic and organizational interoperability.

While the requirement for interoperability in all three dimensions seems obvious, it is a fact that IT systems today are not interoperable in the way that seamless process integration can be realized to its full potential. Only with the ubiquity of internet technologies based on open standards and specifications namely TCP/IP, HTTP and SMTP etc., it has been possible to achieve a high degree of technical interoperability. In this context, the recent developments of Web service standards [19] along with the advent of the SOA paradigm, which are discussed in more detail in the next chapter, have to be highlighted as well. However, in order to enable IT systems to exchange and combine information and accordingly process it in a meaningful manner, it requires agreement on more complex issues, such as the relation to the context within information is created and used and consensus on how to represent meaning of data in principle.

As this work focuses on semantic interoperability in SOA, the following section focuses particularly on the semantic dimension of interoperability. The scope for semantic interoperability as targeted in this work is pointed out taking into account the overlapping aspects identified above. Furthermore, the dimension of semantic interoperability is broken down to the targeted domain of SOA.

2.3 Semantic Interoperability

The quest for meaning in language has a history that is almost as old as language itself [20]. Accordingly, the challenge to achieve semantic interoperability of IT systems is an ongoing effort since the advent of distributed IT environments. In this process, is has turned out that semantic interoperability requires much more than a simple agreement concerning the isolated meaning of a term but rather depends on the individual context [20].

2.3.1 Terms as Representation of Meaning

To further understand semantic interoperability, an analysis of the words *meaning* and *term* is required. In linguistics, meaning is considered as a human artifact [21]. In this sense, terms as well as things to which terms refer have no meaning per se. The meaning is assigned to them by human beings. In order to exchange the meaning, which is subject of semantic interoperability, it has to be encoded by utilizing terms[1], whereas inherently never all facets of meaning can be represented but restrictions have to be made according to an individual context. Thus, a first distinction can be made between the meaning as a human artifact or conceptual idea on the one hand and the term which represents the meaning on the other hand.

To further analyze the semantic interoperability problem, which occurs if the exchanged terms do not refer to the intended meaning, the characteristics of terms as a representation of meaning in context of IT systems should be elaborated. According to system design in informatics, terms that represent meaning refer to information models which can be distinguished along different abstraction levels. This distinction between different abstraction levels for representation of meaning provides the starting point and foundation of the envisaged conceptual framework for semantic interoperability in SOA, which is presented in the following section.

2.3.2 Abstraction Levels for Representation of Meaning

Following the paradigm of separation of concerns, each different abstraction level for the representation of meaning is used for a specific purpose. The starting point in IT system design are highly abstract modeling languages, which should be closely related to the actual meaning or conceptual idea in the mind of human beings. Such highly abstract modeling languages are e.g. the Unified Modeling Language (UML) [22] or the Entity-Relationship Model (ER) [23]. In order to be used in a concrete application context, these information models need to be broken down to a lower application specific level. Considering the common database design methodology of different abstraction levels, it can be distinguished between:

(1) the Conceptual (2) the Logical and (3) the Physical Data Model.

The conceptual data model is used for the abstract modeling of an information space as already mentioned. The logical data model provides a more concrete view on the information space to be used in application development. In the context of database systems, this means to map an ER-model to tables, columns and rows, the relational model. The physical model is private to the actual system processing and storing the data.

In order to get a consistent picture and integrate the above described analysis regarding terms as representation of meaning, a further abstraction level can be added on top of the presented levels. Accordingly, in the following the conceptual idea or the meaning in the human mind is considered as the initial model of a thing or information. Consequently, it can be distinguished between the following abstraction levels:

(0) the Conceptual Idea (1) the Conceptual Data Model (2) the Logical Data Model and (3) the Physical Data Model

---

[1] derived from terminus, *lat.:* terminus = border, border stone, identifier, denotation

A further reference to these basic abstraction levels for information models can be found in the context of model driven architecture (MDA) [24] for modeling software systems from the Object Management Group (OMG) [25]. MDA focuses on functionality and dynamic behavior, however the static data-oriented part can be related to the identified abstraction levels. The MDA viewpoints distinguish between:

(1) Computation Independent Model (CIM) - The computation independent model focuses on the environment of the system and the requirements for the system. A CIM does not show details of the structure of systems. It is sometimes called a domain model, whereas a vocabulary that is familiar to the practitioners of the domain in question is used in its specification. Accordingly, the information model in CIM corresponds to (1) the conceptual data model described above.

(2) Platform Independent Model (PIM) - The platform independent model focuses on the operation of a system while hiding the details necessary for a particular platform. A platform independent view shows that part of the complete specification that does not change from one platform to another. Accordingly, the information model in PIM corresponds to (2) the logical data model described above.

(3) Platform Specific Model (PSM) - The platform specific model combines the platform independent viewpoint with an additional focus on the detail of the use of a specific platform by a system and the underlying implementation. Accordingly, the information model in PSM corresponds to (3) the physical data model described above.

The identified abstraction levels have further differentiated the characteristics of terms as a representation of meaning in context of IT systems and can be taken as a reference framework to further analyze semantic interoperability problems. In the following, the terminology is aligned to the identified abstraction levels in context of database system design including the initial level of the conceptual idea. However, instead of *data model* or *information model* just the term *model* is used. The conception of a message as data or as information depends on the user's or receiver's ability to interpret the data according to a certain context. In a first step, this differentiation should be out of scope while the focus is put on the identification of the different abstraction levels for representing meaning by utilizing terms. In a following second step (cf. Section 2.3.3), the usage of these abstraction levels for the exchange of meaning is analyzed and then the differentiation between data and information becomes relevant.

To summarize, it can be distinguished between the following abstraction levels for representation of meaning, which provide the starting point to the framework to further analyze semantic interoperability:

(0) Conceptual Idea (1) Conceptual Model (2) Logical Model (3) Physical Model

2.3.3 Semantic Interoperability Gap

The goal of semantic interoperability is to ensure that the meaning of exchanged information is preserved in different application context in a distributed IT system. However, as the conceptual idea cannot be directly and fully formalized in an IT system and therefore not exchanged directly, the conceptual idea has to be represented by means of terms. As described above, this representation can be distinguished into the four different abstraction levels. Thereby, it is important to note that the meaning of a term is inherently dependent on the context. The expressiveness of context description thereby differs between the different abstraction levels.

The conceptual idea of a thing captures the full domain context, as it represents the initial model and constitutes a kind of master model or reference model for capturing the meaning per definition. Following the path down to the lesser abstract levels, certain context information gets reduced while the same time application specific information gets concretized. The conceptual model reduces the potentially holistic conceptual graph from the conceptual idea to the focused application domain and refines the conceptual structure such as generalization and composition of classes and its attributes. In a further step, the logical model reduces explicit context description between concepts, in order to map the representation to an application specific level. Thus, logical operations can be well defined on a sufficiently concrete level to enable machine interpretation and processing. On the physical level, context is only encoded implicitly on a technical level specific to the IT system performing the application.

Regarding information exchange, this explicit context description or lack of it becomes important for semantic interoperability. The following Figure 2-1 illustrates a typical information exchange scenario between two IT systems based on service-oriented exchange of messages. The fundamentals for their interoperation are overlapping conceptual ideas about a domain model which describes a certain information space. The domain context may be different between the two IT systems. However, the designers share or refer to the same understanding of specific concepts, which can be located in the overlapping part of the two conceptual ideas.

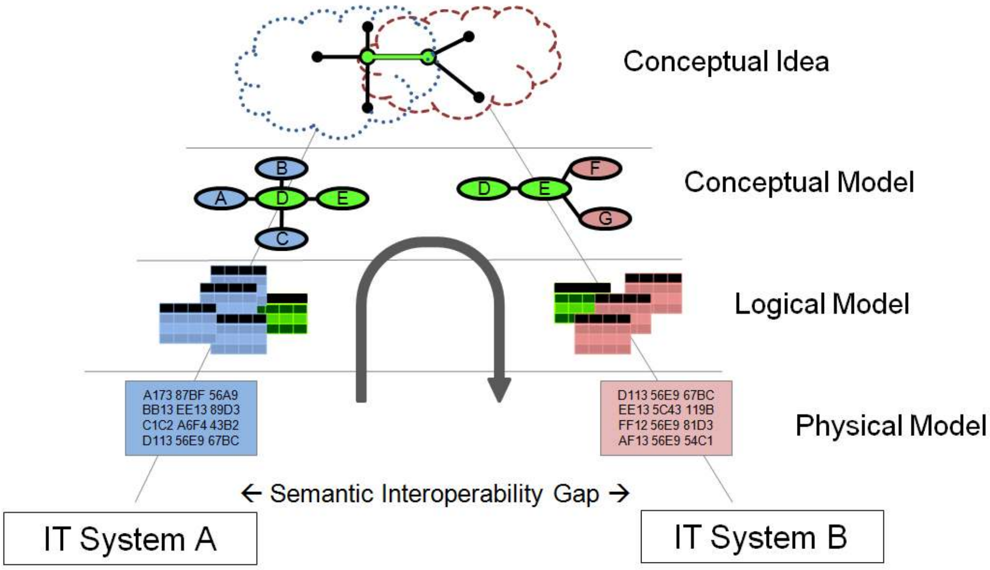


Figure 2-1 Semantic Interoperability Gap

Due to the different domain context the corresponding conceptual models are different. They may also exhibit overlapping parts which are modeled consistently but they may also differ completely and thus only with the reference to the conceptual idea the corresponding parts can be identified. On the abstraction level of the logical model, additional differences in information representation can be identified as the logical model maps the representation to an application specific level, which further differs between IT System A and B. Thus, it can be stated that the semantic interoperability gap grows with each lower abstraction level as the differences between the representations and the conceptual idea increase. Consequently, the largest semantic interoperability gap is given if information is exchanged on the physical level as the concepts including their context relation are only encoded implicitly, which may differ completely between the two IT systems.

In order to overcome the semantic interoperability gap, the information representation needs to be interpreted – in an automated, machine-supported or manual manner – according to the next higher abstraction level. Thus, the representation gets linked to the shared understanding in the two shared conceptual ideas and semantic interoperability is ensured. Finally, in order finish the round-trip across the semantic interoperability gap and to ensure that the information gets processed by the receiving system, the information needs to be represented according to the system’s technical representation on the logical or physical level.

Moreover, it is important to note that even if in a concrete scenario a direct transformation between two different physical models or logical models is applied, the above described steps have to be performed logically and are included within the transformation as virtual steps. However, this logical analysis points out the complexity which these transformations contain.

Another, conclusion can be drawn from the above developed model for semantic interoperability: If two systems are equally designed with regard to their utilized information models on the different abstraction levels, the logical steps to bridge the semantic interoperability gap have to be performed as well. However, the actual round-trip can be shortened, if two models are equal on the same abstraction level. The above described round-trip just has to be performed up to the respective abstraction level where the models are equal. Accordingly, the semantic interoperability gap on this level is not present and therefore no reference to the upper abstraction level is required. This explains why semantic interoperability can be achieved by standardizing the representation models of information – whenever this is possible. This topic is detailed later on in Chapter 4.

Referring again to the aspects identified in the context of semantic interoperability as described in Section 2.2.1, they can be mapped to the above introduced conceptual framework: Pragmatic heterogeneity can be located at the highest level, as this level deals with the idea of a concept in the human mind including the context about the intension of sending or receiving such information. The identified aspect referred to as structural heterogeneity can be located at the logical level, as it deals with heterogeneous schemata. Syntactic heterogeneity, which addresses incompatibilities in encodings and formats, can be mapped to the physical level. Finally, what is referred to semantic heterogeneity in general in Section 2.1.1 as addressing the inconsistencies in terminology and meanings can be mapped to the level of conceptual ideas with regard to meaning and to the conceptual, logical and physical model with regard to terminology each representing the meaning on a different abstraction level. Thus, it can be stated that the developed model describing the problem of semantic interoperability captures consistently the different semantic interoperability aspects identified in current state of research (cf. Section 2.2.1).

As the aim of this chapter is to develop a framework for semantic interoperability in SOA to provide a systematic understanding of the problem addressed in this work, the above developed model describing the semantic interoperability gap needs to be mapped to the domain of SOA. Therefore, as an intermediate step, the concepts and approaches of SOA are introduced in the following section.

2.4 Service-Oriented Architecture

In [26] several definitions of SOA are introduced and compared, whereas the following unified definition is provided:

„*A service-oriented architecture is a framework for integrating business processes and supporting IT infrastructure as secure, standardized services that can be reused and combined to address changing business priorities*.“

This definition outlines two major goals of SOA, namely flexibility and reusability of IT systems. These two characteristics become particularly relevant as more and more business processes are spanning multiple organizational and administrative domains. Changing external business partners need to be flexibly integrated into IT-supported processes while reuse of IT infrastructure needs to ensure that this flexibility can be realized with regard to optimized resource spending within economic constraints.

To address these challenges, the core concept of SOA is the decomposition of complex business processes into a composition of loosely coupled independently managed services providing distinct business functionalities. IT systems supporting business processes are split of into a set of loosely coupled reusable services, whereas each service realizes one modular unit of business logic.

The architectural model of SOA is based on fundamental principles such as modularization, encapsulation and platform-independence. These principles are incorporated from prior approaches, mainly from component-based middleware platforms. Thus, SOA is no revolutionary new development but rather it is based on various known concepts and methods. However, the component-based approach implemented in platforms such as CORBA [27] or EJB [28] require the business partners to adopt a specific object model that might not be suitable for all collaborating parties [29]. Considering that the lifecycle of a component has to be managed by its consumer, the coupling between provision and usage of functionality must be still regarded as tight.

Addressing this shortcoming and extending the ability of loose coupling, the central novelty of the architectural model of SOA relies on the strict focus on the service concept. The service concept represents a further step up in abstraction for distributed IT system design [30]. Consequently, not the component capturing the business functionality takes the center stage but just the service which the component provides replaces the focal point. Thereby, the conceptual model of a service can be defined as follows [31]:

i. A service establishes an agreed relationship between partners in two distinguished roles: a service provider and a service user.

ii. A service is meant to produce benefit of a definite type to the service user and to meet the user’s needs.

iii. A service is the result generated by processes at the interface between the provider and the user and by processes internal to the provider and internal to the user.

The service model can be further refined focusing on the interaction patterns between the above identified roles of the service provider and service user. Accordingly, the service interaction model often is equated with the “find-bind-execute” paradigm. As illustrated in Figure 2-2, firstly a service provider registers a service at a registry which includes a description of the service and the business context relevant for the usage. Then, a service user interacts with the registry for finding service descriptions which fulfill certain criteria, e.g. regarding service class or non-functional properties. This refers to the user’s needs and the aimed produced benefit of a definite type (cf. ii. above). Then, the service user utilizes the service description to bind to a service provider and to invoke the provided service. Thus, the relation between service user and service provider gets established.

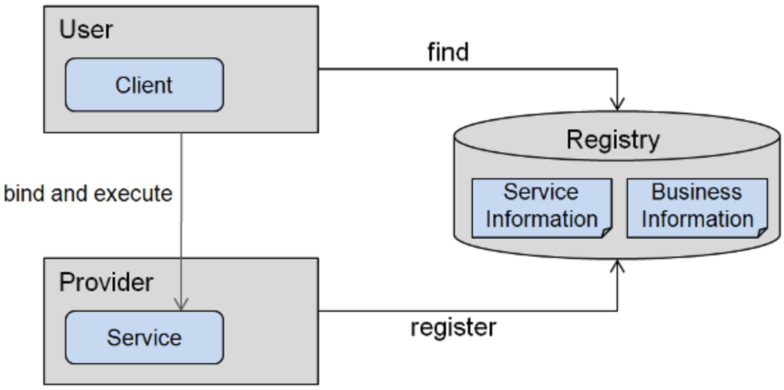


Figure 2-2 Service Interaction Model

In order to enable this so called find-bind-execute paradigm, the SOA concept needs to be instantiated with an appropriate technology. In recent years, fostered by broad standardization initiatives and wide industry adoption, Web services have taken the lead role as the dominant realization approach to implement SOA. The main technologies behind Web services such as XML, WSDL, SOAP, UDDI and BPEL are analyzed in Chapter 3 with specific focus on how they address the problem of semantic interoperability in SOA.

In order to outline how the principles of the SOA model are reflected within a typical enterprise IT architecture, the following Figure 2-3 illustrates the decomposition of complex business processes into a composition of loosely coupled Web services along the so called SOA layers:

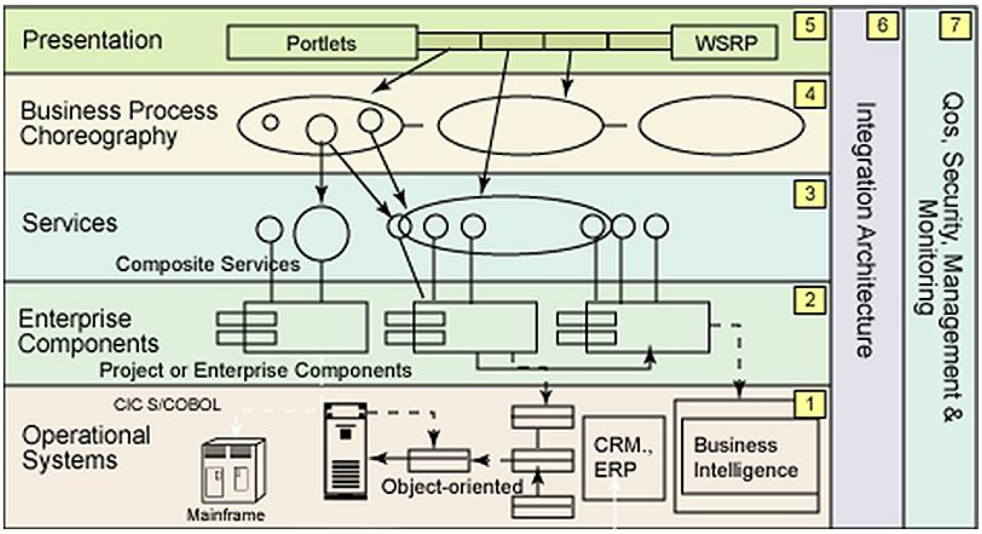


Figure 2-3 Enterprise SOA Layers [32]

- The bottom layer (layer 1) contains existing business applications, which may originate from different organizational domains including e.g. customer relationship management (CRM) and enterprise resource planning (ERP) systems, legacy applications or specifically designed object-oriented systems as well as business-intelligence applications.
- The component layer (layer 2) uses typical container-based technologies and component implementation models. It enables distribution of functional components within the enterprise.

- Layer 3 provides the mechanism to make enterprise-scale components, business unit-specific components and in some cases project-specific components available as services. The interfaces are exported by means of standard service descriptions. Moreover, this layer comprises the service infrastructure (e.g. service registries).
- Layer 4 combines services and other composite services to orchestrations or choreographies which implement enterprise-wide or even cross-enterprise business processes. Visual process modeling and process execution engines are used for this purpose.
- The presentation layer (layer 5) is usually out of the scope of the actual SOA. It is important to note that generally in SOA the user interfaces are decoupled from the services. However, it is part of the figure because recent standards such as Web Services for remote portlets (WSRP) may indeed carry service functionalities directly to the application interface or presentation level.
- Layer 6 (orthogonal) enables the integration of services through the introduction of reliable and intelligent routing, protocol mediation and other transformation mechanisms, often described as the enterprise service bus.
- Layer 7 (orthogonal) ensures quality of service through sense- and respond mechanisms and tools that monitor the state of SOA applications.

Besides the runtime perspective focused in the above SOA layers, as well the design time of SOA needs to be taken into account. Such a holistic perspective is provided in terms of a so called service life-cycle or SOA life-cycle focusing less on any particular service but rather on the entire set of service from design over implementation to usage and monitoring. Even there exists no well-established definition of the term service life-cycle or SOA life-cycle, there is a common understanding about the main phases to be covered in it. Nevertheless, the phases are clustered on different granularity levels and different aspects are more or less highlighted in the various available definitions. In [33] an overview of popular definitions of the service life-cycle model is provided. In order to stick to a consistent perspective within this work, the following refers to a definition provided by IBM [34]. Accordingly, the service or SOA life-cycle can be distinguished into the following phases:

- Model – This phase is about capturing the business requirements and translating them into business process models refined by service identification and service specification.
- Assemble – This phase is about developing reusable services and composing them into service orchestration plans which instantiate the modeled business process.
- Deploy – In this phase the developed services and service compositions are tested and deployed to a runtime infrastructure.
- Manage – The last phase is about maintenance, measurement and optimization of service operations from a technical and as well from a business perspective.

In order to reflect the discussed SOA concepts within the framework of semantic interoperability in SOA described in the next section, the following Figure 2-4 presents a condensed SOA layer model focusing on the conceptually most relevant parts. Furthermore, to stress that the business processes may consist of services and underlying components from different organizational domains the corresponding layers are split as well into different domains.

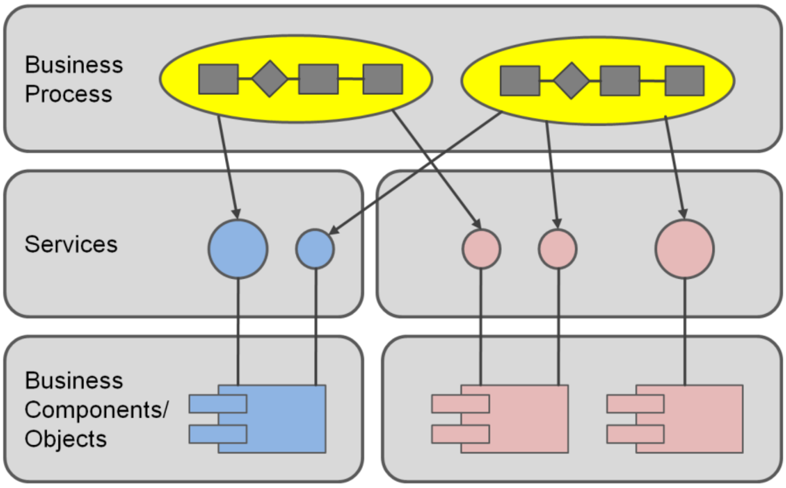


Figure 2-4 SOA Layer Model

- The business process layer describes the cross-organizational business process as a composition of services (from an abstract perspective in terms of business process models and from a concrete perspective in terms of a service orchestration plan).
- The service layer describes the (heterogeneous) services which provide the distinct business functionalities.
- The business components or objects layer describes the underlying components which realize the services implementations for the middle layer.

In the following this condensed SOA layer model is incorporated as an integral part into the framework of semantic interoperability in SOA, which is presented in the next section.

2.5 Framework of Semantic Interoperability in SOA

Before relating the concept of the semantic interoperability gap introduced in Section 2.3.3 to the above described SOA layer model, a connecting step in terms of further analysis of the service artifact presented in the middle layer is elaborated. As the concept of the semantic interoperability gap has focused on information models represented on different abstraction levels, these representations need to be related to the service descriptions.

According to the Web Ontology Language for Services [35] which aims at providing a specification of a service in terms of a formal ontology, the following conceptual characteristics of a service can be distinguished:

- Inputs
- Outputs
- Preconditions
- Postconditions or Effects

In this sense, the specification of a service can be related to the mathematical concept of a function as an abstract entity that associates an input to a corresponding output according to some specific rule [36].

In the service context the input describes information which needs to be provided by the service user necessary to invoke the service and perform the provider's internal processes in order to deliver the service. The output describes information which is generated as a result of the provider's internal processes and delivered to the service user. The preconditions specify the state of the information space of the service before its execution. Moreover, preconditions can be represented as expressions that are required to be true before an operation can be successfully invoked. Vice versa postconditions describe the state of the information space of the service after the execution of the service. Postconditions can be represented as expressions that must be true after the service has been invoked and its operation completes its execution. In the following Figure 2-5 the service model as described above is illustrated:

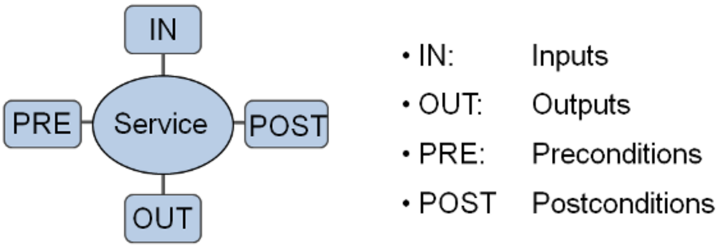


Figure 2-5 Service Model

Considering instantiations of this service model in concrete IT systems, these four service characteristics have an information representation in terms of the different abstraction levels for information models as introduced in Section 2.3.2:

Firstly, a conceptual idea about inputs, outputs, preconditions and postconditions is existent in the mind of an IT system designer. Later in the design process, domain specific representations of these service characteristics can be derived and captured in a conceptual model. With regard to preconditions and postconditions most state-of-the-art approaches limit the representation to textual description addressing the human reader. This is due to the fact that a fully-formal specified conceptual model that represents the preconditions and postconditions is difficult to define on a sufficient level that enables machine interpretation. Thus, further specification and concretization of the information model on lower abstraction levels such as the logical or physical model is limited, too. However, some approaches in the research field of Semantic Web services also target this aspect as further analyzed in Chapter 3.

With regard to input and output parameters, which are subject to information flow in a service composition scenario instantiating the business process model, they can be represented in a conceptual model addressing the domain context by relating the input and output parameters to other domain concepts relevant in the business process. The corresponding information model can be further specified on the logical level representing the abstraction level that is utilized when information flow is specified in a concrete application context. In order to realize the business functionality which is provided by the services, the business components or objects get involved. These components process the input and output parameters and thus need to represent the information according to their specific technical environment. Accordingly, the physical level of the information model can be located on the business components or objects layer.

Taking into account the analysis above, the service model including inputs, outputs, preconditions and postconditions provide a further detailed description of services in the SOA layer model. Moreover, the service characteristics can be represented on the different abstraction levels for describing their information models. As the SOA layer model has distinguished between heterogeneous services originating from different organizational domains, these heterogeneous services can be directly related to the semantic interoperability gap describing the heterogeneous information model representations on the different abstraction levels. Consequently, a unified model illustrated in Figure 2-6 can be derived combining the introduced SOA layer model together with the refined service model and the model describing the semantic interoperability gap.

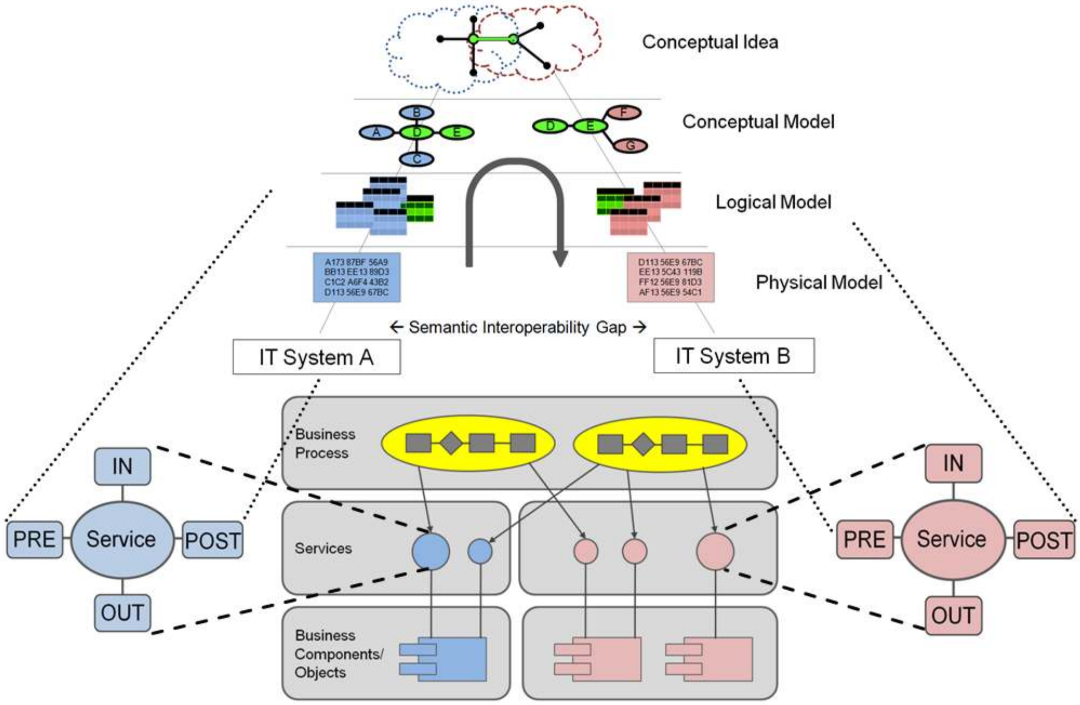


Figure 2-6 Framework of Semantic Interoperability in SOA

Hence, the derived perspective on the three models constitutes the framework of semantic interoperability in SOA as a reference point and problem description. In the following, it provides a common ground for comparison. Consequently, approaches and technologies presented in the state-of-the-art analysis in Chapter 3 as well as the concept for semantic mediation between loosely coupled information models in SOA developed in Chapter 4 refer back to this framework.

2.6 Summary and Reflection

This work addresses the problem of achieving semantic interoperability in cross-organizational service-oriented architectures. Therefore, definitions and conceptual models covering the different aspects of semantic interoperability have been analyzed in a first step. In order to define the problem scope, semantic interoperability has been put into context with related interoperability dimensions, namely technical interoperability and organizational interoperability and overlapping issues have been identified.

An aggregation of conceptual models for semantic interoperability has been developed based on the finding that different abstraction levels for representing information models are fundamental for the understanding of semantic interoperability. Consequently, a model describing the semantic interoperability gap has been derived, that demonstrates how heterogeneous IT systems differ in their information models along different abstraction levels. Starting from the conceptual idea of information in the human mind, to the conceptual model formalizing the domain context, over the logical model to the physical model representing the concrete information model on the technical level, the semantic interoperability gap between heterogeneous IT systems continuously increases. Furthermore, different fundamental approaches for achieving semantic interoperability such as alignment of terminology or transformation between heterogeneous representations could be mapped to the derived model of the semantic interoperability gap.

In order to address the targeted domain of semantic interoperability in SOA, the architectural model and basic concepts of SOA have been presented. A layer model has been elaborated capturing the central approach of SOA that lies in the decomposition of complex business processes into a composition of loosely coupled independently managed services providing distinct business functionalities. The service concept has been further analyzed to link the information model used in a service interface description to the different abstraction levels analyzed in the model of the semantic interoperability gap.

Finally, a unified model has been derived from the analysis above describing the semantic interoperability problem in the context of SOA. Consequently, the unified model forms the reference framework that provides a common ground for comparison between approaches and technologies for achieving semantic interoperability in SOA. In the following, it is referred to this framework including Chapter 3 analyzing the state-of-the-art and Chapter 4 presenting the concept for semantic mediation between loosely coupled information models in SOA.

Chapter 3

State-of-the-Art in SOA for Bridging the Semantic Interoperability Gap

3.1 Overview

Having set the scope for the addressed problem of achieving semantic interoperability in heterogeneous IT systems with particular focus on SOA in the previous chapter, the next step is to examine state-of-the-art approaches and technologies that aim at tackling the identified challenges. Even the selected approaches and technologies are often embedded into a broader context; the analysis tries to focus on the aspects particularly relevant for the topic of semantic interoperability in SOA and to limit the general aspects to the basic essentials.

In a first step, Section 3.2 reviews the idea and concepts of traditional Web services along with its existing technology stack. Furthermore, an evaluation is performed describing the capabilities and limitations of traditional Web services in context of the previously developed framework of semantic interoperability in SOA (cf. Section 2.5).

After outlining the need for formally defined semantics of Web service descriptions, an intermediate step introducing the core concepts and technologies of the Semantic Web initiative are described in Section 3.3. The following Section 3.4 then describes how these concepts can be applied to Web services in terms of so called Semantic Web services. Again, the previously developed reference framework describing the semantic interoperability gap is utilized, in order to provide an evaluation outlining the advantages, limitations and open issues of this technology.

Additionally, a survey is carried out on relevant related areas such as semantic information integration in distributed database systems and distributed object-oriented systems.

Finally, these traditional approaches are related to a detailed analysis of ontology-based strategies for semantic integration including approaches where multiple ontologies are involved. In this context, as well a number of ontology mapping approaches and exemplary tools are investigated.

3.2 Web Services

Whenever the realization of an SOA with state-of-the-art technology is discussed, the term Web service takes an important role as Web services represent the dominant technology for the instantiation of an SOA. In the last decade, Web services have gained considerable popularity. Many software vendors have adopted Web service initiatives and correspondingly provide extensive product portfolios. A large consulting market for advisory services regarding Web service technology has emerged. Furthermore, there are many organizations which are involved in the refinement of Web service standards. However, driven by marketing campaigns and the related ongoing SOA and Web service hype, often the problem remains that few people seem to actually agree on what a Web Service is [37]. The introduction in Chapter 1 has already briefly outlined the idea behind Web services and their capabilities for ensuring semantic interoperability. This section aims to provide a further detailed analysis and starts with clarifying what Web services are and how they are used to build an SOA. Furthermore, this section explains the core Web service concepts and related technologies. Finally, an analysis about the shortcomings with regard to semantic interoperability is provided.

3.2.1 Definition and Concepts

The World Wide Web consortium defines Web services as programmatic interfaces for application to application communication over the World Wide Web [38]. This definition states one major aspect that Web service interaction is usually machine to machine. In the same sense another definition states that the easiest way to describe a Web service is to say that it is done on the Internet, using Web protocols, and it does not involve a live user operating a Web browser [39].

The World Wide Web consortium furthermore highlights the importance of XML by defining a Web service as a software application identified by a URI [40], whose interfaces and bindings are capable of being defined, described, and discovered as XML artifacts. Moreover, a Web service supports direct interactions with other software agents using XML based messages exchanged via Internet protocols [41].

Even though there exists no uniform, consistent, standardized and official terminology for Web services, it can be stated that a common understanding about fundamental Web service characteristics is shared among the various actors. In the following the fundamental characteristics that are featured by Web services are listed and briefly described [42]:

- Programmable - Web services are accessible by programmable interfaces. Web services are used for application communication and not for human information processing. Web services do not have a user interface.
- Self-descriptive - Web services include meta-data which can be processed during runtime, e.g. name, description, version, quality of service etc.
- Encapsulated - Web services encapsulate independent and discrete functionalities that perform a particular task.
- Loosely coupled - Web services communicate over messages, implementation details are hidden to Web service providers and Web service consumers.
- Location transparent - Web services are location independent and can be accessed from anywhere at any time only dependent on access rights of applications that consume the Web services.
- Protocol transparent – Web services are based on the Internet protocol stack. Operations and messages can support multiple, such as Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP)

- Reusable and composable - Web services can be divided into further finer grained Web services or multiple reusable basic Web services can be composed to a new Web service.

The here presented characteristics are not unique to Web services but rather reflect principles that have been adopted from previous middleware approaches. Therefore, in the following the factors and conditions that have shaped the development of Web services as well as fundamental Web service concepts are presented and discussed.

Evolution of Integration Middleware

In a dynamically changing and more and more global economy companies and organizations are continuously seeking for new means to cope with competitive pressure. The need to shorten production and development cycles, to reduce costs and time-to-market, to increase customer satisfaction, and to rapidly adapt to market changes has historically led companies to collaborate and to distribute their business processes.

In order to automate business processes spanning multiple administrative and organizational domains the existing stand-alone applications had to be opened and interoperability mechanisms had to be established. Distributed object-oriented technologies and middleware platforms, such as the Distributed Component Object Model (DCOM), Java 2 Platform Enterprise Edition (J2EE) or the Common Object Request Broker Architecture (CORBA), are powerful means for the integration of applications within companies or organizations.

However, as argued already briefly in section 2.4 about SOA, for integrating systems across organizational domains these technologies are only suitable up to a limited extent. This is due to the highly heterogeneous environments in which prescription or shared agreement of common object models and corresponding programming languages is not appropriate and feasible. Even the communication between organizations that are using compatible middleware technologies is not necessarily practicable since the underlying data transport may be blocked by security facilities such as firewalls. Moreover, not just the transport of data but also its shared interpretation has to be taken into account.

Taking into account the conceptual similarity to component-based approaches, which become obvious by comparing the main characteristics of Web services, it can be stated that the basic idea behind Web Services is not new. However, reflecting the analysis above it becomes comprehensible that with the emergence of XML as the dominant standard exchange format as well Web services based on XML message exchange formats and XML based interface descriptions have taken the lead role in building applications from reusable building blocks.

Web Service Scenario

The following figure illustrates the idea and a communication scenario of Web services across organizational domains:

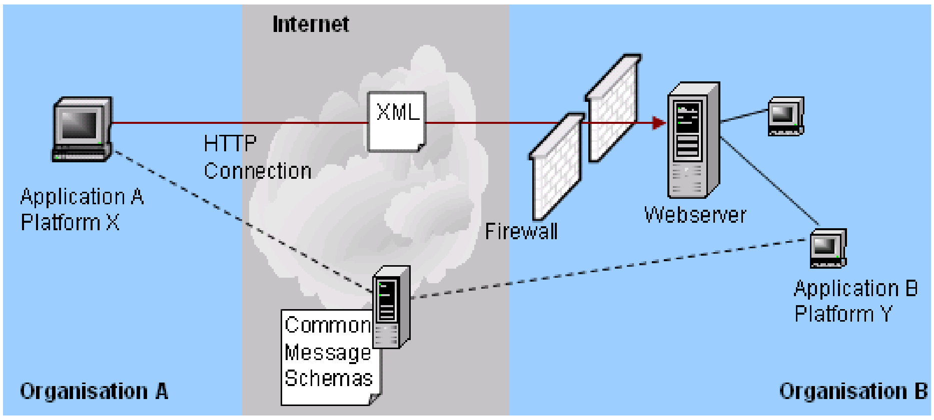


Figure 3-1 Cross-Organizational Communication using HTTP and XML [37]

The success of the World Wide Web (WWW) as the ubiquitous infrastructure for information exchange has brought the idea of using the WWW also as a medium for communication between applications based on standard Web protocols, such as the Hypertext Transfer Protocol (HTTP) or the Simple Mail Transfer Protocol (SMTP). As most organizations are using a Web or mail server the data transport can be handled on existing infrastructure. Moreover, in many cases this is the only communication channel which is permitted by security policies such as firewall configurations. On top of these transport protocols messages are defined in terms of the Extensible Mark-up Language (XML). In order to be able to process the message content, application A and application B share common message schemas that are e.g. based on the XML Schema Definition Language (XSD). Based on XSD the definition of customized mark-up language for a specific business context is possible providing a representation of structured data in a human- and as well machine-readable manner. The transformation of XML messages to specific programming languages and object instantiations and vice versa has to be performed by the processing applications corresponding to their underlying platform.

Web Service Interaction Model

However, Web services are not monolithic and have to be regarded in context of a distributed architecture. Based on the general service interaction model presented in Section 2.4 further refinements with regard to the concrete Web service technology can be made. The following roles for the interaction of Web services can be identified [42], whereas the concrete XML standards used in the role descriptions are further described in the following Section 3.2.2:

- User: The user consumes the Web services based on service descriptions defined in the Web Service Description Language (WSDL).
- Provider: The provider provides services and ensures that the services are accessible over programmatic interfaces described in declarative Web service descriptions (WSDL).
- Registry: The registry contains declarative Web service descriptions of various Web service providers and their access points. It provides registry services based on standards such as UDDI or ebXML.

It is important to note, that the roles user and provider are exchangeable. A user can act as a retailer combining several Web services according to a business process using the Business process Execution Language (BPEL), which then is provided as an upper level Web service in the provider role. The following figure illustrates the Web service interaction model:

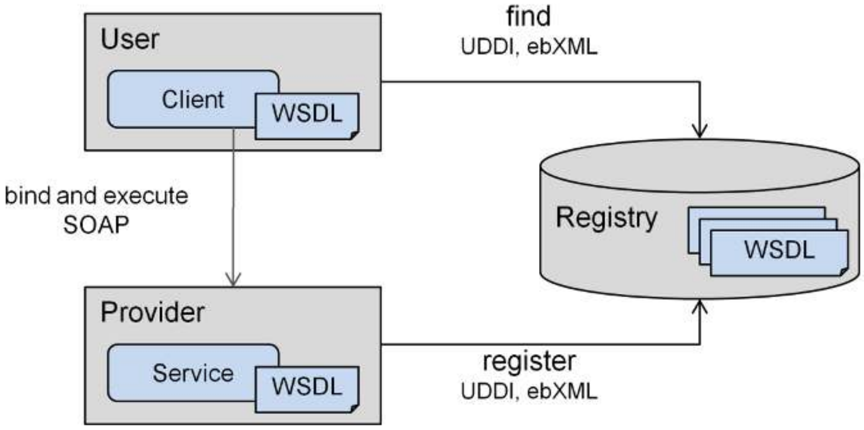


Figure 3-2 Web Service Interaction Model

In an exemplary simple interaction pattern the life-cycle of a Web service can be described as follows:

1. The service provider provides a Web service and publishes its declarative description (WSDL) into the Web service registry using standardized registry services described in e.g. UDDI or ebXML.
2. The potential user of a Web service sends a search query to the Web service registry utilizing further standardized registry services described in e.g. UDDI or ebXML.
3. The Web service registry contains a categorized collection of registered, trustful Web services which are each described in declarative Web service descriptions (WSDL).
4. After discovering the desired Web service in the Web service registry further details about message formats and transport protocols can be gathered.
5. Based on the service description a binding to message formats and transport protocols can be performed by the user. Then the user can communicate with the provider over XML-based message exchange format and protocol (SOAP) and consume the desired Web service.

Web Service Composition

As already mentioned above Web services are mostly applied as an instantiation of an SOA. Recalling the basic idea of SOA in the context of Web services, IT systems supporting business processes have to be split of into a set of loosely coupled reusable Web services, where each Web service realizes one modular unit of business logic. Consequently, mechanisms are required for the consistent and meaningful integration of Web services. This integration is called Web service composition. In order to obtain meaningful composition results, Web services need to be invoked in a well-defined order and they have to exchange data. The following Figure 3-3 illustrates control and data flows in the composition of services.

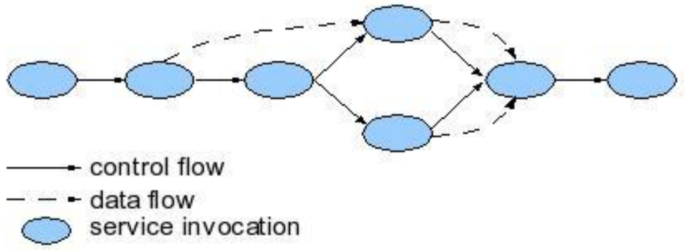


Figure 3-3 Flow-based Web Service Composition [43]

Two different composition models can be distinguished: orchestration and choreography. There is no well agreed common sense regarding these two definitions. Nevertheless, mostly it is considered that in an orchestration all interactions that are part of a business process (including the sequence of activities in particular Web service calls, conditional events such as loops etc.) must be described like in a traditional workflow system. This description is then executed by an adequate engine which has control of the overall Web service composition. On the other hand, choreography is more collaborative and less centralized in nature. Only the public message exchanges are considered relevant and moreover, each Web service only knows about its own interactions and behavior. In contrast to orchestration, there is not an entity that has a global view or control of the process.

3.2.2 Technologies and Standards

Accompanied by the success story of SOA in recent years a couple of standards for Web services have emerged, which either have become an official standard or at least have the status of a widely used de-facto standard. Aggregating the technologies into an overall picture the following Web service stack can be derived:

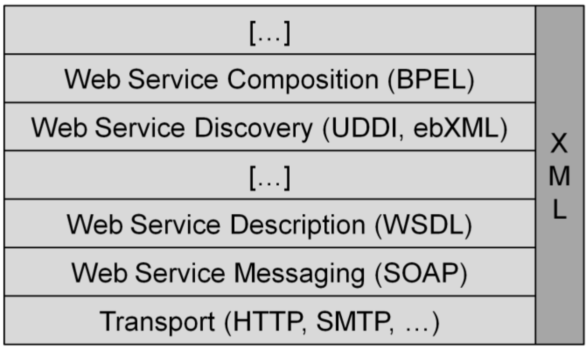


Figure 3-4 Web Service Stack

As indicated by the free spaces there are still more elements that are part of the Web service stack, such as technologies related to quality of service (Web service transactions support [44] or security and reliability by means of encryption [45]) or service management. In the following the most important standards are described in more detail [37]:

- **Simple Object Access Protocol (SOAP)** - With the Simple Object Access Protocol (SOAP) a lightweight format and protocol for the exchange of XML messages in a request/response-manner has been developed. SOAP holds the status of a W3C recommendation [46]. SOAP defines a convention that can be used to represent remote procedure calls (RPC). In the case of using HTTP as the protocol binding, an RPC call maps naturally to an HTTP request and an RPC response maps to an HTTP response.

Although, SOAP was intended to provide networked applications with RPC services in XML, the interaction with a Web service is not necessarily RPC-centric but may also be document-centric. In the former case a service is seen as a set of methods to be invoked remotely and the messages are serializations of business objects. With document-centric communication, however, the documents themselves are the main purpose of the distributed computation and the services are considered as components that read, store and produce documents.

- **Web Service Description Language (WSDL) -** Communication mechanisms and message representations are not sufficient to create services. One of the most important characteristics of a service is that it exposes a well-defined interface that describes its functionality. This includes the description of a set of messages that the service receives and sends, a set of named operations and, if the service is deployed, a binding to a documented network address. The binding mechanism defines services as collections of network endpoints or ports. A port is defined by associating a network address with a binding. Finally, a collection of ports define a service. For describing the interface of a Web service a specific XML language, the Web Service Description Language (WSDL), has been developed and holds the status of a W3C recommendation [47]. A Web service description contains definitions (data types and messages), operations and service bindings, thus providing all necessary information for a client to interact with a Web service.
- **Universal Description, Discovery and Integration (UDDI) and Electronic Business XML (ebXML) -** There are several, mostly industrial driven, registry initiatives for Web services, among them Electronic Business XML (ebXML) and Universal Description, Discovery and Integration (UDDI). UDDI and ebXML provide a mechanism for clients to find Web services. A UDDI registry contains categorized information about services, about the businesses that offer services and about the interfaces and communication standards that are used for conducting transactions. Requestors can search a UDDI registry, find services based on certain matchmaking criteria and retrieve service details, such as links to the service description (WSDL) and the invocation address. It is important to note that UDDI does not define a specific registry implementation but the interfaces and data structures that are used for storing and finding services and businesses. Similarly to UDDI an ebXML registry allows businesses to find partners, to define trading-agreements, and to exchange messages in support of business operations.
- **Business Process Execution Language (BPEL) –** BPEL provides a mechanism for the composition of Web services. The design process of such service compositions is also called programming in the large. In order to keep the composition independent from the underlying IT infrastructure, the exact data flow and control flow is provided in a composition language, which can be interpreted by workflow execution engines. Different approaches for such languages have arisen, e.g. WSFL [48] or XLANG [49]. However, BPEL, which is based on the before mentioned Web service specifications, has been the most successful and holds the status of an OASIS standard [50]. BPEL defines a business process as an XML-serialized description of data flow and control flow between participating Web services and allows to run the process in a long-running asynchronous manner. Data flow and manipulation can be expressed in XML-related languages such as XPath [51] and XSLT [52]. In order to ease the design of service compositions in BPEL, vendors offer a range of graphical integrated development environments, e.g. the Oracle BPEL Process Manager as illustrated in Figure 3-5Figure 3-5 Development Environment for Process Design:

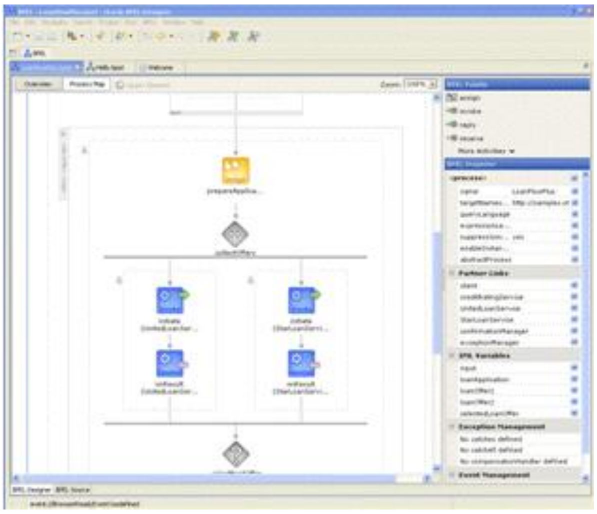


Figure 3-5 Development Environment for Process Design

However, the above technologies still exhibit fundamental limitations when it comes to automation in the Web service life-cycle and further tool support especially with regard to Web service composition. The composition design is still complex and time-consuming, which is due to the shortcomings regarding semantic interoperability of conventional Web service technology. The following section deeper elaborates on these shortcomings based on the analysis of Web service technology above and the reference framework of semantic interoperability in SOA developed in Chapter 2.

3.2.3 Evaluation

The fundamental characteristics of Web services have been analyzed in section 3.2.1. However, it can be stated that the current Web service technology stack has only partially kept the promise of enabling Web services which are truly self-descriptive, encapsulated and loosely coupled.

Limited Web Service Characteristics

Self-description of Web services is limited. Having a look at the meta-data provided by Web services, they just allow for processing during runtime to some extent. Web service descriptions defined in terms of the XML-based Web Service Description Language (WSDL) describe the operations, parameters and Internet address of a Web service rather in syntactical and structured manner. However, they are lacking any context information required for advanced automated processing. For machines or software applications which act behind Web services the information and description of a Web service is barely interpretable because the underlying mark-up language XML lacks an expressive semantic background and WSDL does not define any further semantics. Thus, the limitations of XML encoded information just allows Web services to parse each other input and output messages and verify whether it adheres to the expected formats, and eventually locate each piece of information within the message parameters. But unfortunately, the cooperating Web services do not have any means to decode the meaning of the messages on the conceptual level, in order to extract the information they contain. Referring back to the perspective of the service model presented in the above developed Framework of Semantic Interoperability in SOA (cf. Section 2.5), further shortcomings adhered to the WSDL approach can be identified. Furthermore, additionally to the limitations regarding self-description of input and output parameters WSDL-based Web service descriptions lack any information about the preconditions or postconditions of the Web service.

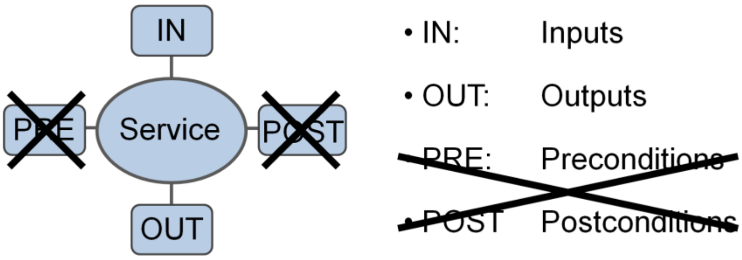


Figure 3-6 WSDL-based Web Service Model

Thus, cooperating Web services understand the structure of each other messages but do not understand the content of such messages [53]. Taking this into account, the semantics of Web service operations and data structures in corresponding messages can only be interpreted by humans. As a consequence human interaction is necessary in order to understand what a service does and how it can be invoked.

Therefore, encapsulation is limited, too. As the semantics cannot be exposed to the externally available meta-data further internal information about Web service semantics is required. Thus, the core principle of encapsulation, the hiding of internal information, cannot be assured to the full extent as further insight into the Web service is necessary for the user in order to combine them and create reasonable Web service compositions.

With regard to loose coupling of Web services it can also be stated that this goal has only be achieved to a limited extent. As Web services are just enabled to parse each other's messages and process their structure cooperating Web services have to rely on strictly agreed message schemas in order to ensure sound exchange of message content. Thus, it can be stated that on the semantic level a strong coupling is still necessary. In order to overcome this situation, scenario specific adapters and transformations have to be integrated in the Web service interaction by means of human intervention.

Human Intervention in Web Service based SOA-Life-Cycle

As a consequence of the analysis above it can be stated that for many tasks of the Web service based SOA life-cycle manual efforts in terms of human interaction and collaboration is necessary, which is time consuming, costly and error-prone. The following figure illustrates the fields of human interaction according to the SOA layer model presented in Section 2.4:

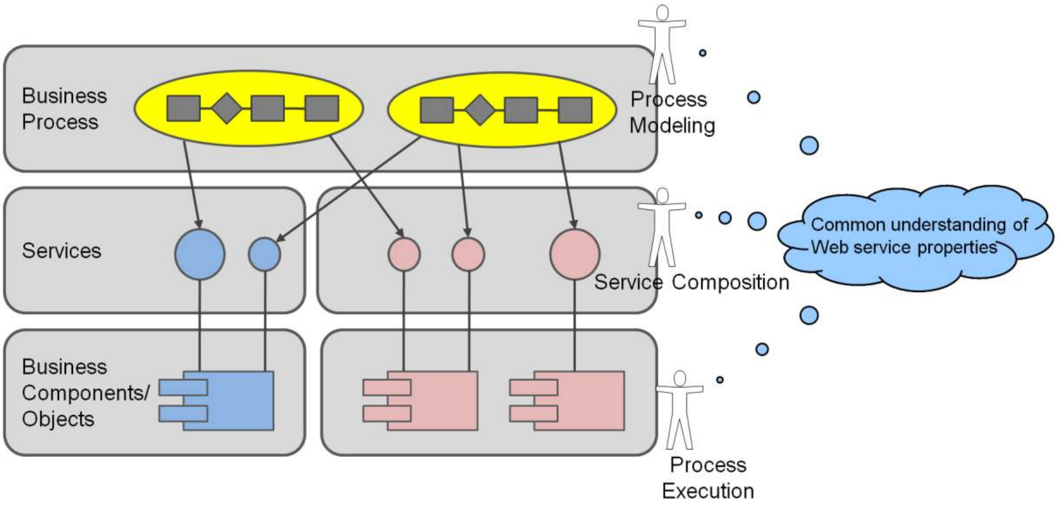


Figure 3-7 Human Interaction in Web service technology based SOA-Life-Cycle

During business process analysis and modeling process experts need to understand the business context in which the IT-supported business process takes place. This includes a detailed requirement analysis for the business process and specification on an abstract level of which IT services are needed in order to fulfill the requirements. Also the information flow between the building blocks of a business process has to be specified and modeled according to the domain context. Based on this input created by process experts shared operation patterns and message structures of Web services can be derived, which are fundamental requirements for their interoperation. Due to the limited expressiveness of Web service meta-data the presented Web service technology does not provide the means to handle the heterogeneity of Web service properties automatically. Human intervention in terms of meetings, documents, etc. is needed to define and agree on a common understanding of Web service properties which then can be reflected on the technical level by cooperating service providers and service users.

On the technical level Web services need to be discovered and composed in order to realize the desired business process. Caused by the limitations of self-description, discovery approaches applied in UDDI and ebXML categorize Web services using external flat service classifications in terms of so called *tModels* that represent taxonomies, identifier systems, etc. However, Web service discovery is only keywords-based. As a result, this leads to low quality of the retrieved results as keywords are often not unique and contextual information is not considered. An analysis of different Web service discovery approaches and their limitations in quality have been discussed in [54]. As a consequence, in practice as well for Web service discovery human intervention is needed. Automatic discovery mechanisms just provide a first step in the process. Additionally, human experts are involved to select or eventually further discover provided Web services based on a shared understanding of Web service properties required by the user as well as Web service properties of the provider.

Regarding Web service composition, BPEL as the de-facto standard allows for the design of abstract processes. However, the activities within a process are still bound to fixed XML-based interfaces which consequently include fixed operation patterns and message structures. In a highly heterogeneous business environment this approach still lacks flexibility, since all collaborating actors and corresponding systems need to adhere strictly to a previously defined common message schema in order to ensure semantic interoperability. Composition specific adapters and transformations have to be integrated in the Web service interaction. Data flow and data manipulation is expressed in XML-based languages, such as XPath and XSLT. According to the hierarchically tree-structured data model of XML the approach behind traditional WSDL and BPEL-based Web service composition is mainly syntactical. Consequently, the implicit semantics of services can only be understood by a human composer and the whole range of composition tasks, including the selection of matching services, the control flow and the data flow design, is a manual and recurring effort. Thus, the composition design still remains complex, time-consuming and error-prone. The lack of explicit semantics in Web service descriptions is an obstacle in increasing automation and further tool support in the process of composition design [55].

Looking at Web service execution it can be stated that compared to the previous SOA life-cycle phases the degree of automation is much higher. This can be related to the fact that during the phases of business process modeling and Web service composition the context dependencies and heterogeneities of the distributed business process have been already anticipated and broken down to a concrete technical level. Hence, the execution is limited to a purely technical task processing the instructions defined in the control flow, data flow and transformation design. However, this also implicates that in case of even small changes in the business process the existing execution plan becomes obsolete. As no mechanisms are incorporated on the execution level these changes accompanied by additional heterogeneity cannot be handled on the fly and in a preferable transparent way. Consequently, the top down approach starting from business process modeling phase followed by Web service composition and discovery has to be iterated again according to the SOA life-cycle.

Limitations of Underlying XML Data-Model

The above described problems leading to shortcomings regarding the automation potential in the Web service technology based SOA life-cycle originates from the limited expressiveness of the underlying XML languages. In Section 3.2.2 it has been pointed out that the whole Web service technology stack is based on the markup language XML. XML provides a meta-language to syntactically describe the structure of documents. In fact, the XML syntax is designed for representing an encoded serialization of documents. Thus, XML has a very limited range of expression for modeling complex information entity semantics with context awareness in terms of constrained relationships and properties. Consequently, it can be stated that XML is a poor language for data modeling if the goal is to represent information entities in the problem domain such that they correspond explicitly to the user's or process expert's conceptual model of information entities in this domain [56]. The principal constructs available in XML for expressing relationships are limited to "containment" (hierarchy), "adjacency" (A 'followed by' B), "co-occurrence" (if A then [also/not] B), "attribute", and "opaque reference". These constructs are indeed useful for serialization, but are not optimal for modeling information entities of a problem domain with sufficient expressiveness. All conceptual and relational semantics must be mapped into syntactic structures leading to limitations in processing and consequently the XML processor is not able to recognize their significance.

XML in the Semantic Interoperability Gap

As discussed above, XML schemas are a means to provide integrity constraints to information sources, either documents or semi-structured data. XML schemas provide a basic vocabulary and mechanisms for structuring information hierarchically. The tree structure of tags as represented in e.g. the W3C Document Object Model (DOM) [57] reflects the view used for application development. Taking the perspective of different abstraction levels for information representation as analyzed in the model of the semantic interoperability gap in Section 2.3.3, the abstraction level of the logical model provides a concrete view on the information space to be used in application development. It refers to the approach of structuring information e.g. in terms of tables, columns and rows according to the relational model known from database design. Thus, taking into account the goal of XML to structure information hierarchically and the platform independent usage of XML in concrete application scenarios, it can be stated that the abstraction level for the information model of XML can be located at the layer of the logical model. The following Figure 3-8 illustrates the placement of XML and its schema definition language in the model describing the semantic interoperability gap:

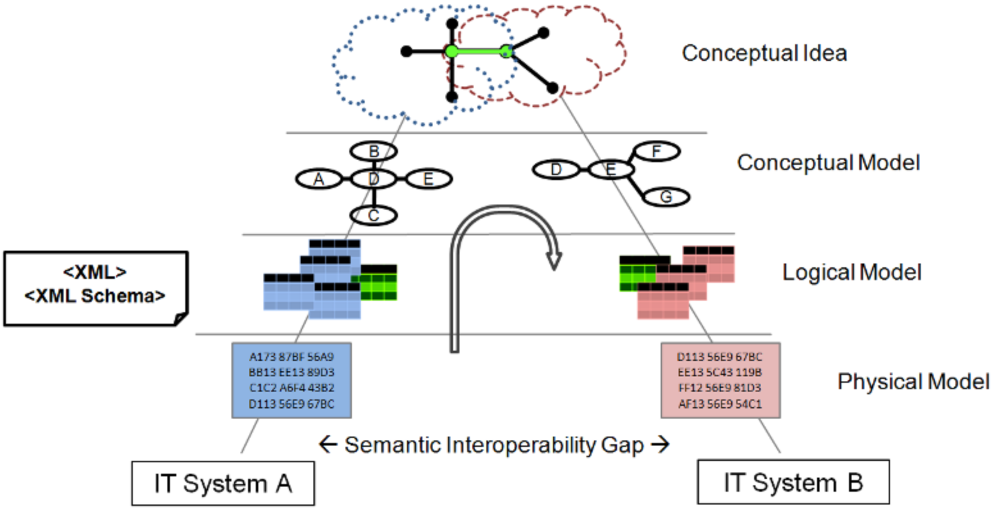


Figure 3-8 Placement of XML in the Semantic Interoperability Gap

Having identified where XML can be located in the abstraction level stack the above described limitations of traditional Web service technology regarding semantic interoperability become evident. The identified necessity of human intervention to handle the heterogeneities of Web service properties can be directly linked to the required round-trip bridging the semantic interoperability gap described in Section 2.3.3. As the underlying semantic differences are only contained implicitly with the XML schemata describing the WSDL-based Web services, business experts need to manually interpret the message formats to the conceptual level. Linked to their shared understanding of conceptual ideas achieved in meetings, documents, etc. semantic interoperability is ensured. Furthermore, in order finish the round-trip across the semantic interoperability gap the information needs to get represented according to the XML schemata of the receiving system, i.e. the Web service user. Finally, this round-trip needs to be expressed in a direct transformation, e.g. in terms of XSL transformation or XPath-based BPEL data flow manipulations, in order to be automatically executable during runtime.

With regard to transformation development and recurring manual efforts the above described situation results in low efficiency of the SOA life-cycle and thus significantly jeopardizes the fundamental goal of SOA to quickly and easily respond to business process changes. In order to overcome this unsatisfying situation the conceptual expressiveness of Web service descriptions has to be brought from an implicit to an explicit level. This means that technologies are needed that allow for the formal definition and for the standardized exchange of conceptual descriptions of Web services. Moreover, frameworks and tools are needed that are capable of reasoning about the formally defined semantics.

The Semantic Web initiative defines descriptive languages for representing machine-interpretable metadata and provides technologies with the aforementioned capabilities. As an intermediate step, before describing how these concepts can be applied to Web services in terms of so called Semantic Web services, firstly the core concepts and technologies of the Semantic Web are described in the next section.

3.3 Semantic Web

The shortcomings regarding automatic information processing caused by the limited expressiveness of meta-data does not only target Web services. Moreover, it is a general problem of the World Wide Web. The World Wide Web consists of billions of Web pages and is often described as a huge distributed knowledge base. To a large extend the information is stored in the form of static HTML pages or in the form of HTML pages dynamically generated from contents of databases upon request of a client. HTML, if rendered and displayed in a Web browser, is very suitable in case the information is consumed by a human being. However, for a computer the meaning of the data is not processable due to the lack of explicit semantics [58]. In order to address these shortcomings, the Semantic Web initiative has evolved as a collaborative effort led by the W3C with participation from a large number of academic institutions and industrial research partners. The goal of the Semantic Web is to make the content of Web pages machine-understandable and processable, in order to enable computers to perform tasks, which require interpretation of the meaning of Web resources [59]. Thus, computers are the primary user addressed. However, finally, by easing Web-based application development focusing on the integration of heterogeneous information resources, the human user shall be provided with more sophisticated means for the usage of the World Wide Web.

3.3.1 Definition and Concepts

Tim Berners-Lee, one of the main initiators of the Semantic Web vision, defines the Semantic Web as an extension of the current Web, in which information is given well-defined meaning, better enabling computers and people to work in cooperation [60]. This definition points out that the Semantic Web should not be regarded as a contrary development that is detached from the current Web but rather as complementary. Furthermore, the definition in [60] states that the Semantic Web will bring structure to the meaningful content of Web pages, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users.

Aligned to the above definition the W3C provides a generalized definition by stating that the Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries [61]. However, the definition further includes, that the Semantic Web is based on the Resource Description Framework (RDF) [62] and thus refers to a concrete standardized language for expressing semantics on the World Wide Web. That points out the strong commitment to a standards-based approach to enable the vision of the Semantic Web.

Before further technologies and standards for the Semantic Web are presented, firstly the core concepts are introduced in more detail in the following section.

Ontologies

As already mentioned the primary goal of the Semantic Web is to provide an extension to the current World Wide Web by enriching its content with machine processable meaning or semantics. Providing the means for processing the semantics of Web content to a certain extend could enable tasks that are currently difficult to do, such as locating content, collating and crossrelating content or drawing conclusions from information found in two or more separate sources [63].

In order to enable machines to process Web content with regard to its meaning, the content needs to be expressed in machine understandable ontologies. In this sense, ontologies can be defined as formal and explicit specification of a shared conceptualization of a domain [64]. This means that an ontology defines a conceptual model of a domain that ideally represents an agreed consensus among involved actors. On the one hand, the formal and explicit manner ensures that the so modeled meaning can be processed by machines. On the other hand, the shared aspect ensures a commonly accepted understanding based on consensual terminologies, so that the modeled meaning is processed the same way anywhere. Thus, ontologies interweave human understanding of symbols with their machine processability [65] and consequently enable to bridge the gap between the real world and IT systems [66]. In more detail ontologies consists of the following elements:

- individuals or instances - are the base components of ontologies and represent concrete or abstract objects (also referred to as A-Box containing assertional knowledge)
- classes or concepts - represent sets of individuals and can be considered as types (also referred to as T-Box containing terminological knowledge)
- properties - represent characteristics of individuals and concepts
- relations - individuals, concepts, and properties can be related to each other expressed by properties
- rules - formulate statements about individuals, concepts, properties, and relations dependent on other statements

Some traditional approaches regard rules separate from ontologies. But since the mathematical formalization of ontologies in terms of description logics, this distinction becomes obsolete and rules, as well based on description logics, become an essential part of domain conceptualizations. The here referred description logics are a subset of predicate logic. I.e. the ontology elements discussed before are represented as predicates and logic operators within formulas, e.g. unary predicates for atomic concepts and binary predicates for atomic relations [67]. Description logics are aimed at being tractable on the one hand but keeping a high degree of semantic expressiveness on the other hand. Therefore, description logics are designed to be decidable in contrast to their superset predicate logic, which is undecidable [68].

Thus, knowledge modeling gets a solid mathematical foundation. Subsequently, this formalism enables machines to interpret or reason over knowledge representations. However, there is a trade-off between semantic expressiveness and computational complexity of reasoning and thus many different variants of description logics have emerged [69].

From Knowledge Based Systems to Semantic Web

Having modeled content in that manner, knowledge based systems using inference engines and reasoners, as illustrated in Figure 3-9, can query and process the content as a knowledge base.

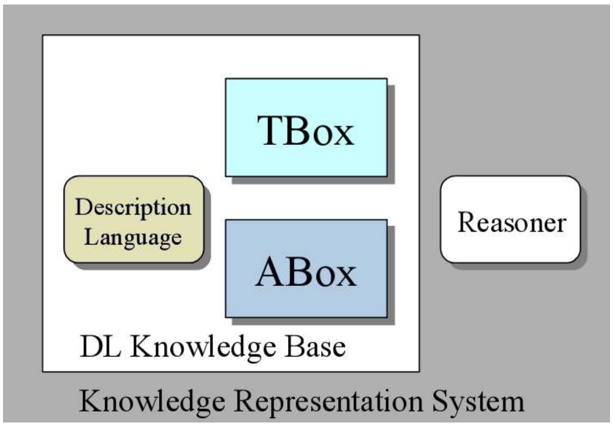


Figure 3-9 Typical Knowledge Representation System based on Description Logics [70]

The vision of the Semantic Web is about applying these concepts to the World Wide Web and using it as a huge knowledge base enabling powerful knowledge-based applications. Consequently, reasoning has to be realized on a partial and incomplete knowledge base. This background has yield to the concept of open-world semantics in contrast to closed-world semantics used in the context of knowledge-based systems, which perform reasoning on a complete and closed knowledge base. The concept of open-world semantics assumes that the absence of information about a fact does not indicate that this fact is false. Hence, it is possible to reason over a dynamic knowledge base without generating contradictions.

This idea of the Semantic Web has received high interest in academia and industry, which has led to the formation of a steadily growing, international research community. Consequently, a wealth of work has been produced that mainly covers [71]:

- formal ontology languages (*e.g.* [72]) and efficient reasoning techniques (e.g. [73], [74])
- ontology management technologies (e.g. [75]), which cover methodologies and tools for ontology engineering (e.g. [76]), scalable ontology repositories (e.g. [77]), and techniques for ontology versioning and evolution support (e.g. [78]);
- ontology-based data integration (e.g. [79], [80]);
- several applications for Semantic Web technologies (e.g. [81])

The latter two are further discussed as they target the specific context of this work. Furthermore, Semantic Web services as an application of Semantic Web technologies are described in Section 3.4 and ontology-based data integration is presented in Section 3.6. Common to all areas of Semantic Web research and related industry activities is their commitment to a strong standards-based approach led by the W3C. The corresponding technologies and standards are described in the following section.

3.3.2 Technologies and Standards

The W3C has released several standards to realize the Semantic Web vision as illustrated in Figure 3-10. In the following the Semantic Web stack which is also referred to as the Semantic Web layer cake including the most important standards and basic technologies is described in more detail:

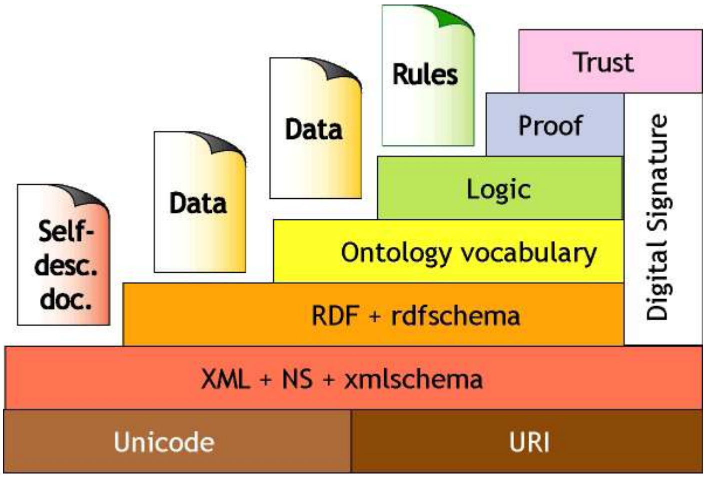


Figure 3-10 Semantic Web Stack

- **Unicode and URI** - The foundation of the Semantic Web stack is built by means of a standardized encoding of data (Unicode), which joins different character sets to one international character set together with the Unified Resource Identifier (URI) standard, which allows the identification of any resource in the Semantic Web. A Uniform Resource Locator (URL) is a specific type of a URI and identifies a resource that is retrievable over a network. However, it is important to note that URIs are not only used for identifying resources that are retrievable on the Web, but they can also reference any other resource whether it is a concrete object such as Web page or any abstract concept.
- **XML, Namespaces and XML Schema** - XML is used as a universal format for message exchanges (cf. Section 3.2.2). XML enables the structuring of data through opening and closing tags, which eases the structured processing by parsers. Tag names can be specified in different namespaces (NS) to avoid name collisions. The underlying structural model of XML is hierarchical and thus an XML instance can be regarded as a tree. XML schema allows to specify grammars to define how the different tags can be structured.
- **RDF** - The Resource Description Framework (RDF) provides a mechanism to make statements about data. These statements - also called triples - consist of subject, predicate, and object. The subject is the resource described, the predicate is a property of the subject, and the object is the value of the property. A set of statements spans a unidirectional graph, which is also referred to as the RDF-Graph. Conceptually, RDF is based on the semantic relational data model [82]. For example, it can be stated that *"Nils is the author of this thesis"*. As RDF, among other notations, can be serialized in XML, the statement above could then be represented as the following:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
                xmlns:ex="http://exampleVocabulary.org/1.0/">
  <rdf:Description rdf:about="http://www.ThesisURI.net/">
    <ex:author>
      <rdf:Description rdf:about="http://www.NilsURI.net/">
        ...
      </rdf:Description>
    </ex:author>
  </rdf:Description>
</rdf:RDF>
```

Figure 3-11 XML Serialization of RDF

- **RDF Schema** - All important resources in the Web should be identified by an URI, so that statements can refer to it. Objects can be resources as well as empty nodes or literals. Empty nodes represent anonymous resources that do not have a URI and are just used to connect other parts of the graph, e.g. sub-properties of an aggregated concept. In order to define vocabularies for these statements, it is possible to define classes of resources and properties that the class members share. Furthermore, these predicates defined in vocabularies can be referred to by namespaces. These vocabularies are specified in RDF Schema, which is similar to an ontology definition language but less expressive and less formalized. RDF Schema builds on top of RDF, i.e. it is based on the RDF data model and a set of standard properties and resources to create simple domain-specific vocabularies. For example, RDF Schema allows to define classes in terms of using inheritance and properties specified with their domain and range. The detailed specification of RDF Schema can be found in [83]. Basically, the RDF Schema meta-data model is very similar to the meta-data model of object oriented programming languages. Accordingly, it is possible to define classes, i.e. sets of individuals that have shared characteristics. However, the meta-models are distinct with regard to a significant difference. In object orientation a class is defined listing the properties that the instances of this class share. In contrast, in RDF Schema the properties take the central role. Properties are independent concepts as well defined outside of class definitions, which are described in terms of the classes which they apply to.

- **OWL** - The ontology layer is based on top of the RDF layer. While RDF Schema provides efficient reasoning complexity, its semantic expressiveness is relatively limited. In order to model complex ontologies which go beyond simple classifications or typed hierarchies, more expressive language features such as additional relations between classes (e.g. equality or disjointness), enumerated classes and cardinality constraints are needed. With the Web Ontology Language (OWL) the W3C has specified a widely accepted language to define more expressive ontologies that keep the trade-off between rich expressiveness and computational efficiency. OWL is based on former ontology projects namely the DARPA Agent Markup Language [84] and the Ontology Inference Layer [85] and uses the RDF syntax as well as most of the RDF Schema constructs. However, in contrast to RDF Schema, OWL allows for defining cardinality constraints on properties. Moreover, boolean combinations of class expressions, such as intersections, unions and universal quantifiers as well as existential quantifiers can be used to define restrictions on how properties are used by instances of a class. In order to address the trade-off between expressiveness and reasoning efficiency, OWL contains three decreasingly-expressive sublanguages: OWL Full, OWL DL and OWL Light. With regard to the concrete application context of the semantic mediation toolkit developed in this work (cf. Section 6.2.2) the usage and adequacy of the three sub-languages is further discussed. Together with RDF and RDF Schema OWL belongs to the key standards of the current Semantic Web.

The layers discussed so far represent the current state of the Semantic Web research that has reached a clear conceptualization, whether the upper layers represent the future requirements for the Semantic Web and are much more under construction.

- **Logic** – The logic layer covers technologies and methods for inferring facts that are not explicitly stated. Currently, the main debate is focused on the logic layer and how to integrate rules to make OWL more expressive. Several candidates for rule extension of OWL have been submitted to the W3C, whereas the Semantic Web Rule Language (SWRL) [86] and the Rule Interchange Format (RIF) [87] has received much attention recently [88]. In that context a debate has arisen, whether this extension by rules should be realized by means of splitting the Semantic Web stack with rules and OWL ontologies sitting side by side on the same level on top of an extra intermediate layer. The purpose is to allow closed-world semantics as an alternative to open-world semantics exposed by OWL. With respect to compatibility to the existing languages namely RDF and OWL, [89] presents an approach that allows for forms of closed-world assumption by remaining the stack architecture.
- **Proof and Trust** - Besides the logic layer, the far reaching vision is to enable heuristic engines which can proof whether a statement is correct or wrong based on ontologies and rules queried from the Semantic Web. The proof layer should enable tracing and explaining the logical reasoning steps, i.e. explaining why a particular conclusion has been reached. Furthermore, it is aspired to create a so called Semantic Web of Trust by utilizing authentication mechanisms including signing assertions based on digital signatures. Thus, a given statement can be referred to a specific person or author. Moreover, trust relations including transitive relations over multiple actors can be reasoned, in order to transfer trust characteristics and achieve network effects.

3.3.3 Evaluation

The Semantic Web is still under construction and an ongoing process. As discussed above the Semantic Web stack still has to be fully realized. Furthermore, limitations of the core Semantic Web languages RDF and OWL have been investigated theoretically and through study of ontology applications as they are being created and used in practice. A wide variety of issues have been identified ranging from required enhancements to expressiveness including e.g. extended data types and qualified number restrictions to major additions such as the incorporation of temporal concepts [90], semantics of geospatial data [91] or the reflection of uncertainty in terms of fuzzy logic related aspects [92]. Some of the most common limitations have already been addressed in implemented systems with language extensions where these extensions are well understood, and efforts have already begun with a view to standardization of so called low-hanging fruits [93]. In this context as well a new version of OWL namely OWL2 has been developed [94].

One particular relevant conceptual problem with regard to this work concerns the Semantic Web inherent notion of open world semantics. Open world semantics are reflected in standardized Semantic Web languages such as OWL and implemented in corresponding inference engines. The underlying concept of the open-world assumption is designed for reasoning on a partial and incomplete knowledge base. This notion is well suited for the general idea of the Semantic Web. However, when applying it to the context of SOA, as targeted in this work, the open world assumption might not be the optimal choice. Even, the addressed information space in SOA landscapes is heterogeneous and distributed still in most scenarios the set of involved services is self-contained and hence the inferencing process is performed on a stable and complete knowledge base. Thus, with regard to SOA closed world semantics would be a more suitable approach. This fundamental difference points out that the applicability of the current Semantic Web standards as a general technology for information integration might lead to some drawbacks in a particular context such as SOA. As presented above (cf. Section 3.3.2) the support of closed world semantics besides open world semantics is subject to the ongoing discussing regarding the further realization of the Semantic Web stack.

Besides these outlined conceptual shortcomings practical limitations can be identified with regard to performance of inference engines and a general immaturity of the required technical infrastructure. RDF and OWL stores are still slower than optimized relational databases, in particularly with growing difficulties when the technology has to scale up. However, the performance is improved steadily [95]. Thus, it can be stated that gaps in standards and implementations still exist and adoption is limited by typical problems with early technologies. This includes the requirement for a critical mass of practitioners and corresponding running applications. Semantically annotated information sources applying RDF or OWL are still rare in the WWW. Moreover, different domain fields have developed and are using ontologies of their own. Ontologies are typically developed by domain specific expert groups without much systematic collaboration with other fields. When using such isolated ontologies in cross-domain environments still semantic interoperability problems arise. As a result, there is the danger that the global Semantic Web will not emerge but rather a set of isolated mutually incompatible Semantic Web islands may arise [96]. This issue targets the problem of ontology integration and in particular the fields of ontology matching, ontology alignment and ontology mapping, which are further discussed in Section 3.6. However, anticipating such ontology integration opportunities points out the potential that lies in the evolving Semantic Web islands and hence when linked together they can contribute to a step by step realization of the Semantic Web vision.

Having outlined the basic concepts and standards of the Semantic Web initiative the following section will describe how these concepts are applied to Web services in terms of so called Semantic Web services.

3.4 Semantic Web Services

In consideration of the shortcomings of traditional Web services discussed in Section 3.2.3, the idea of bringing implicit service semantics to an explicit level has arisen. By providing machine understandable Web service descriptions with formally defined semantics powerful inference engines and matchmaking mechanisms can be enabled, in order to automate the whole composition process including discovery, composition, execution and interoperation of Web services. After providing a definition of what Semantic Web services are in terms putting them into the context of Web services and Semantic Web technologies, the basic ideas and concepts of Semantic Web services are introduced. Furthermore, the most dominant Semantic Web service technologies and standards are outlined. Moreover, the placement of these technologies and standards in the framework of semantic interoperability in SOA (cf. Section 2.5) focusing particularly on the semantic interoperability gap will be analyzed with regard to advantages and shortcomings.

3.4.1 Definition and Concepts

There exists no universally established definition for Semantic Web services. However, there seems to be a general shared understanding of the basic concepts and characteristics of Semantic Web services. For example in the context of the W3C Semantic Web service interest group it is simply stated that the integration of Semantic Web technologies to Web services constitutes what Semantic Web services are [97]. Other definitions further include the purpose of Semantic Web services and state that they are extended Web service descriptions with rich semantic annotations and upon this, provide inference-based techniques for automating the usage of Web services [98]. Hence, it can be summarized that the basic idea of Semantic Web services is about applying Semantic Web technology to Web services in order to combine flexibility, reusability, and universal access of Web services with the power of semantic markup and reasoning [99]. Considering Web services as the dynamic part of the World Wide Web and the Semantic Web as an extension of the current static Web, consequently Semantic Web services can be defined as the combined evolution of these two dimensions. The corresponding classification of Semantic Web service concepts is illustrated in the following Figure 3-12.

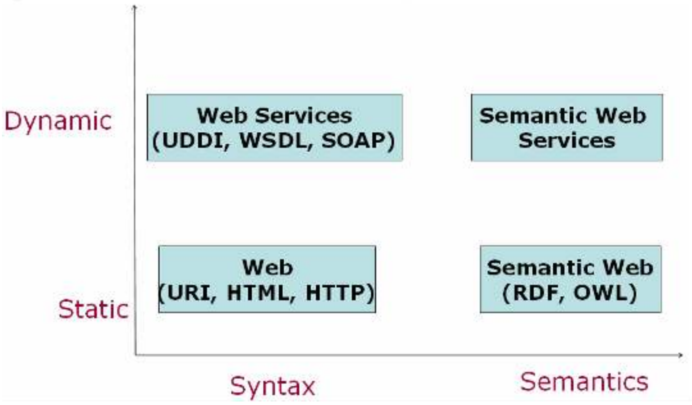


Figure 3-12 Classification of Semantic Web Service Concept [99]

The long term vision behind Semantic Web services is to enable dynamic goal-based service composition and to use powerful inference engines and matchmaking mechanisms, in order to automate the whole Web service life-cycle including discovery, composition, execution and interoperation of Web services. On the one hand, the research background comes from the Semantic Web community (cf. Section 3.3). And on the other hand, there is also an influence from the research field of dynamic planning in artificial intelligence research, in particular with regard to goal-based discovery and composition with is further discussed in Section 3.4.3.

Generic Web Service Ontologies and Domain Ontologies

The basic idea behind Semantic Web services is to use ontology languages such as OWL in order to create machine-understandable Web service descriptions. This can be achieved by enriching Web service descriptions with so called upper ontologies in terms of specific Semantic Web service ontologies. Such ontologies define generic concepts for the description of Web services as illustrated in Figure 3-13:

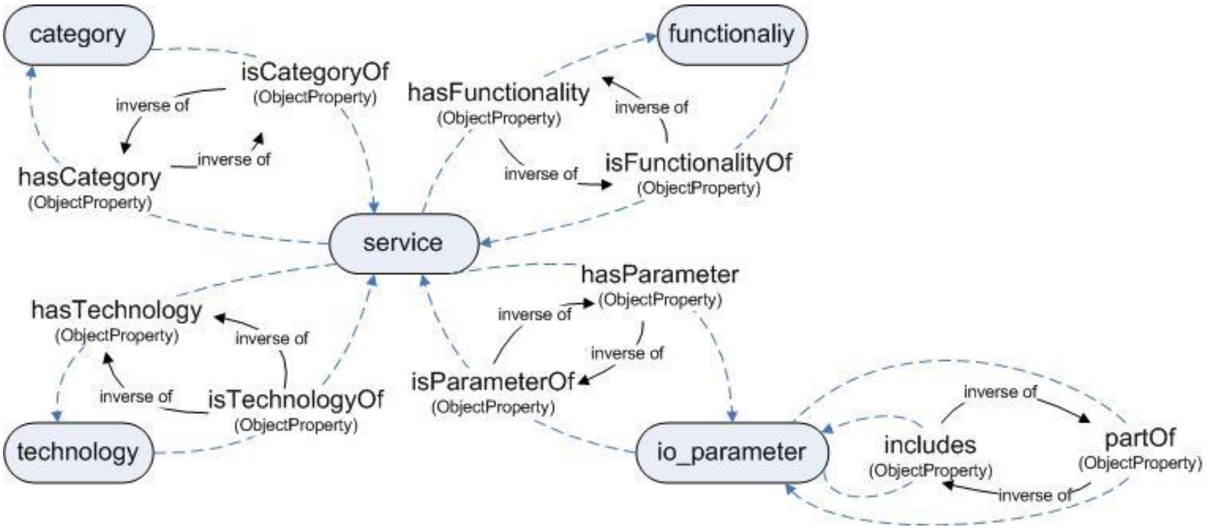


Figure 3-13 Exemplary Web Service Ontology [100]

Such generic Web service concepts then can be further refined in terms of domain specific ontologies in order to describe a concrete Web service in a particular application domain.

Building Blocks of Semantic Web Services

In this sense, the following aspects regarding the formalization of Web service descriptions can be distinguished [101]:

- Semantic Model - A semantic model is a set of machine-interpretable representations used to model an area of knowledge or some part of the world. As outlined above the semantic model consists of a generic or upper semantic model on the one hand and of an application domain specific semantic model on the other hand. Such semantic models are represented in terms of ontologies that embody some community agreement regarding terminology and conceptualization. They are combined with a formal representation based on description logics that consequently allow for advanced information processing.

- Concept – A concept is an element of a semantic model. Accordingly, the concepts are the building-blocks for the Semantic Web service description. They cover functional as well as non-functional service aspects including the description of message parameters as well as references to underlying technological bindings of abstract Semantic Web service descriptions or functional descriptions in terms of specifications of pre- and postconditions.

- Semantic Annotation - A semantic annotation can be contained in a Web service description or provided in an additional document that relates to or defines the concepts of a semantic model that are used to describe the Web service. Accordingly, a semantic annotation can be regarded as the framework for the above mentioned aspects. In contrast to traditional Web service descriptions which rely on XML, ontologies are used as the meta-data model for describing Semantic Web services. Furthermore, semantic annotations provide references to the underlying technological binding, i.e. how the Web service is realized and how to access it during runtime. Often this can be a traditional Web service that exhibits XML entities for describing the Web service.

Additional Semantic Layer on top of Traditional Web Services

Semantic Web services cannot be regarded as a technology without any path dependency. The success story of SOA with its dominant instantiation in terms of XML-based Web services requires the concept of Semantic Web services to reflect this layer of existing technology. Therefore, the explicit semantic annotation of Web services has to be provided in terms of an additional semantic layer on top of traditional XML-based Web service technology. Accordingly, the so called grounding of Semantic Web services by traditional Web services ensures the reuse of existing technology instead of requiring a re-implementation. The following figure illustrates the generic grounding idea. A more detailed and technical analysis with regard to different realizations is given in Section 3.4.2.

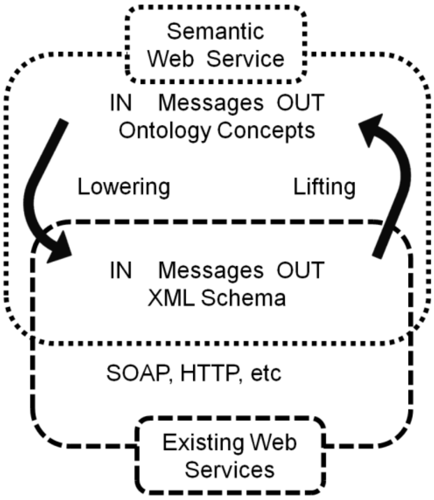


Figure 3-14 Generic Semantic Web Service Grounding

On the one hand, the service descriptions are lifted on the ontology level, which provides the foundation for machine interpretation and service life-cycle automation. On the other hand, the semantic annotations provide a bidirectional mapping that includes also the lowering between the OWL meta-data model and the XML Schema meta-data model exhibited by the underlying existing pure XML-based Web services. This description of the mappings between the meta-data models allows a Semantic Web service consumer to derive the corresponding technical representation for the Web service message parameters during runtime. Such a bidirectional mapping is not a trivial task. As outlined above this mapping has to bridge the gap between two different abstraction levels regarding meta-data models. Taking also into account that the different abstraction levels furthermore feature different expressiveness, for example with regard to polymorphism, it requires to ensure that the advantages introduced by the semantic layer are retained. This aspect is further discussed in context of the realization of the approach presented in this work that applies Semantic Web service technology on top of existing XML-based technologies to address the semantic interoperability challenge in SOA (cf. Section 6.4).

Based on the conceptual approaches presented above the goal of Semantic Web services, i.e. the provision of machine-understandable Web service descriptions can be achieved. Thus, it can be stated that Semantic Web services fulfill the fundamental SOA promise of providing self-descriptive services from the machine perspective. The following Figure 3-15 illustrates the machine-based interpretation of Web services.

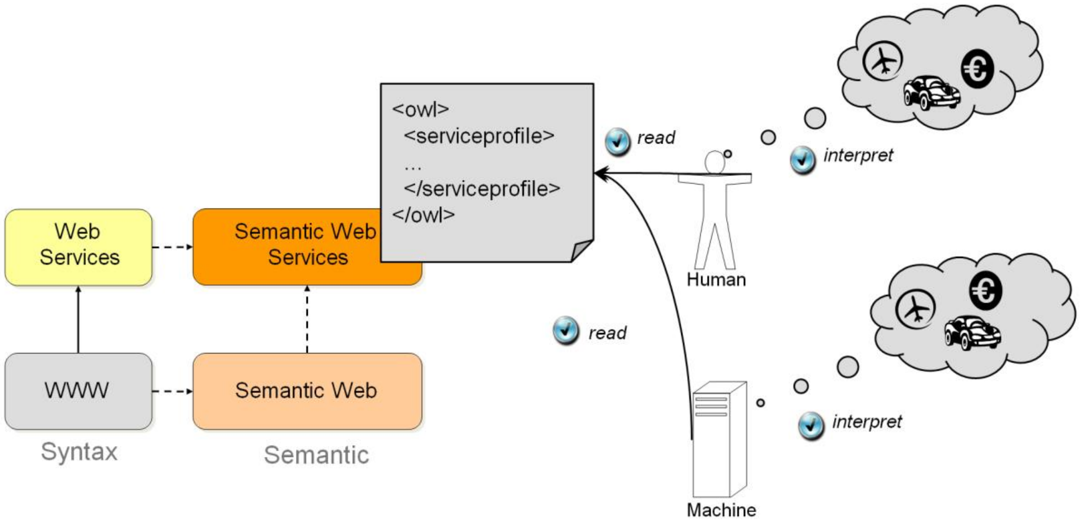


Figure 3-15 Machine-based Interpretation of Web Services

On the one hand, the advanced expressiveness of Web service descriptions eases the SOA life-cycle tasks for process experts. The formal character of conceptualizations facilitates a consistent interpretation of service specifications and thus promotes a direct transfer of formalized requirements to application development and Web service implementation. On the other hand, the mathematical foundation of description logic-based Web service not only enables machines to read service description as known from traditional Web services but moreover machines are provided with the means to interpret the content according to the formalized conceptualization. This facilitates the automation of the SOA life-cycle and further tool support for Web service usage especially with regard to formal business process modeling, Web service composition, service enactment and semantic integration of Web services.

Semantic Integration with Semantic Web Services

As already discussed data or more precisely information integration is not only a structural but mainly a semantic problem (cf. Section 2.3). Defining explicitly and coherently the semantics of information entities is crucial for meaningful integration results. This applies either to humans with regard to agreement processes as well as for machine based processing. Accordingly, the central idea of utilizing the Semantic Web service approach for semantic integration in SOA lies in the mapping of different heterogeneous Web service descriptions to one coherent shared formal conceptualization in terms of an ontology. The following figure demonstrates how Semantic Web services are applied in a typical semantic integration scenario based on two different and heterogeneous traditional XML-based Web services.

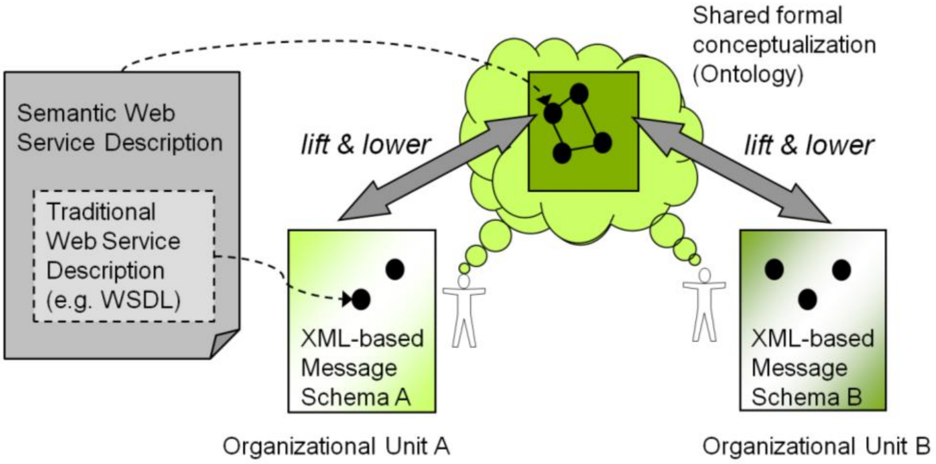


Figure 3-16 Semantic Integration with Semantic Web Services

Firstly, domain experts of the two organizations develop a shared formal conceptualization that captures the requirements for information models of both organizations. The shared conceptualization is then formalized in an ontology language such as OWL. Existing WSDL-based Web services, which describe their parameters and message parts in terms of XML schemas, are mapped to the shared ontology by bidirectional lifting and lowering transformations between the meta-data models. Moreover, the shared ontology can be used to derive message schemas for new Web services.

Before the above presented approach is put in context to the model describing the semantic interoperability gap, it is further analyzed how different technologies and standards instantiate the concept of Semantic Web services. Then, from a combined perspective including technical and conceptual issues, the identified advantages and shortcomings for bridging the semantic interoperability gap in SOA are discussed.

3.4.2 Technologies and Standards

Many approaches for semantic enrichment of service descriptions have arisen with sometimes overlapping or opposing concepts. In the following, existing frameworks that instantiate the outlined approach above and define comprehensive specifications for semantically describing Web services are examined. The aim is to give an overview of the technologies and standards that most of the research in the field of Semantic Web services is based upon. The most relevant approaches have been submitted to standardization bodies. In the following five different specifications that have been published by the W3C in recent years are described:

- Web Ontology Language for Services (OWL-S) [102]
- Web Service Modeling Ontology (WSMO) [103]
- Semantic Web Services Framework (SWSF) [104]
- Web Service Semantics (WSDL-S) [105]
- Semantic Annotations for WSDL and XML Schema (SAWSDL) [106]

Web Ontology Language for Services (OWL-S)

OWL-S is an OWL-based Web service ontology, which supplies a core set of markup- language constructs for describing the properties and capabilities of Web services in a machine-interpretable form. OWL-S markup of Web services facilitates a higher degree of automation of Web service tasks, such as Web service composition, execution and interoperation [107]. It can be stated that OWL-S is the most mature and most widely deployed comprehensive Semantic Web service technology [108], which is reflected by the availability of a large number and variety of tools. As OWL-S is applied in the prototypical toolkit of this work, consequently OWL-S is described in more detail.

In particular, OWL-S is an upper ontology for services. It is structured into three complementary parts, which are further illustrated in the following figure:

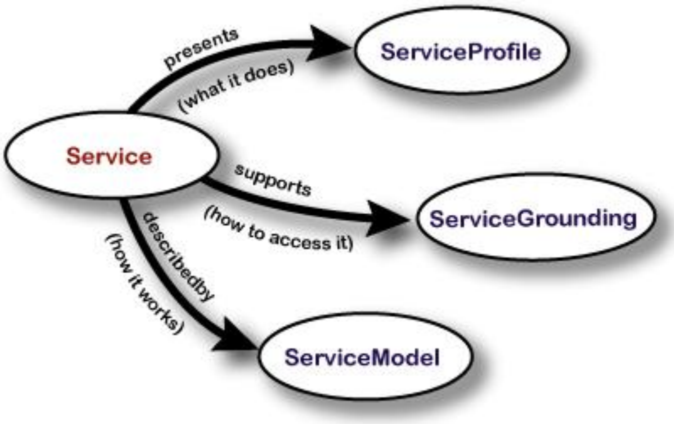


Figure 3-17 Top Level of OWL-S Service Ontology [107]

OWL-S specifies that a service can have multiple service profiles that are used for advertising and discovering services. Furthermore, a service can be described by at most one service model that provides information about service operation and their dependencies. Finally, a grounding has to be specified which describes how to access and invoke the service and its binding to a traditional Web service.

More detailed the service profile describes what the service does to be used by service requesters for discovering or directories to categorize advertised services. The service profile consists of three pieces of information. The first is about the name of the Web service, its provider including contact information and a natural language description. Furthermore and most important, the service profile includes the functional description of a service. It consists of a description of input and output parameters by means of relating them to OWL concepts from domain ontologies. Additionally, it describes preconditions required by the service and its expected effects according to the Web service model presented in Section 2.5. The conditions are represented by logical formulas specified in terms of OWL-based concepts that model expressions. These specific concepts are defined in the OWL-S ontology and further include constructs of a rule language, namely the Semantic Web Rule Language (SWRL) [109], as OWL itself does not completely provide the necessary language constructs to model logical expressions. Finally, it is possible to describe various non-functional properties, e.g. quality-of-service ratings or response time information in terms of OWL concepts.

Once a service has been discovered and selected the service profile is not used anymore. Subsequently, the service model is processed. It specifies how to interact with the service by presenting the possible service interactions and their dependencies to be conceived as a process.

The service model can either consist of an atomic process or a composite process. An atomic process expects one message and produces one message, whereas a composite process builds upon several atomic processes that can expect different messages over time depending on before received messages. Thus, by describing the service model in terms of a composite process a stateful service is described. The different dependencies can be expressed by various control constructs which specify the message flow. In order to make it possible for the service client to interact properly with the service, the service model also presents input, output, precondition, and effect descriptions (IOPE) for each atomic process as specified in the profile.

Finally, the grounding of a service specifies how to access the service in terms of protocol, addressing and message formats. Furthermore, the service grounding needs to deal with the mapping of abstract input and output parameters of atomic processes to concrete messages processed by a concrete Web service realization. The default mapping is the WSDL grounding mechanism. However, different mappings are possible. An OWL-S service can be bound to a concrete WSDL-based Web service by means of mapping OWL-S atomic processes to WSDL operations and OWL-S input and output parameters to WSDL messages. However, as message parts in WSDL are specified using XML Schema by default and parameters in OWL-S are expressed in terms of OWL classes, this mapping task becomes complex because XML Schema cannot express the description logic based semantics of OWL classes (cf. Section 2.3.2). Therefore, an OWL-S service grounding provides an XSLT-based mapping mechanism. The grounding transforms OWL instances serialized in RDF/XML into corresponding XML instances that are structured according to given XML Schema types. This transformation has to be performed for service inputs and vice versa respectively for service outputs. But similar to XML Schema, XSLT is based on XPath and therefore conceptualized on a completely different abstraction level. Accordingly, it cannot capture the semantics of OWL and has to handle the OWL individuals on a syntactical level. Thus, the successful mapping demands for complicated XSLT scripts specific for each RDF/XML serialization as different types exist. This can be seen as a shortcoming of the grounding mechanism or a lack of appropriate transformation languages which are able to capture both the meta-data model of tree-based syntactical XML entities and the meta-data model of OWL ontologies. However, to sum up, by providing the three ontology parts for specifying a service profile, service model, and service grounding OWL-S enables explicit semantic enrichment of traditional existing WSDL-based Web services without any impact or necessary changes to the underlying implementation.

Web Service Modeling Ontology (WSMO)

WSMO shares the vision with OWL-S, but it differs much in the approach for achieving it. WSMO is an alternative approach, which is not built on the W3C standard OWL. Furthermore, in contrast to OWL-S it does not define explicit service ontologies, but it provides a conceptual framework where ontologies can be specified in. One possible instantiation for the WSMO framework is given by a corresponding specification language called WSML [110]. Moreover, under the WSMO umbrella a Web Service Execution Environment named WSMX [111] has been developed as a reference architecture and implementation. In general the WSMO framework defines four top-level notions as illustrated in the following Figure 3-18:

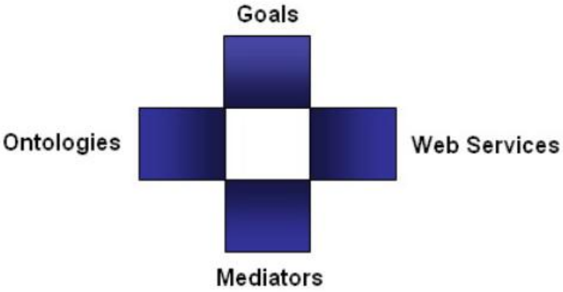


Figure 3-18 WSMO Top Level Notions [103]

- Ontologies define formally specified domain knowledge and terminology, whereas other WSMO elements are making use of these domain specific ontologies.
- Goals describe objectives that clients want to achieve by using Web services.
- Web services are defined in terms of semantic descriptions about functional capabilities and the usage in terms of an interface.
- Mediators provide means for resolving potentially occurring heterogeneities.

In contrast to OWL-S, the WSMO approach does not purely focus on semantic annotations of Web services in terms of providing ontology concepts for functional and non-functional service properties but it propagates a goal-based approach for the usage of Semantic Web services. The idea is that clients formulate requests in terms of goals, which formally describe the objective to be achieved while abstracting from technical details. The system then automatically detects, eventually composes and executes the suitable Web services in order to solve the goal [112]. This goal-based approach has its research background in the field of artificial intelligence and planning algorithms in particular. The problem domain is represented as states, whereas states can be expressed in terms of logical axioms. Then Web services can be regarded as state transition operators. Based on conditions, effects and goal planning algorithms a path can be derived from the initial state to the goal state based on backtracking or forward chaining expression provers known from declarative programming [113]. The applicability of this goal-based planning approach in SOA is discussed in the next Section 3.4.3 about the evaluation of Semantic Web services approaches.

Additionally, WSMO includes a mediator concept to deal with the interoperation problems between Semantic Web services. WSMO defines specific mediator services which perform translations between ontologies that describe input and output parameters as well as goals. Accordingly, the idea of mediators is to overcome heterogeneous resource descriptions by resolving incompatibilities on the data level on the one hand and on the process level on the other hand. Mediation on the data level mainly covers the integration of different terminologies. On the process level mediation is performed by aligning heterogeneous interaction patterns between different Web services, e.g. by splitting or grouping messages or by changing their order. WSMO foresees four kinds of mediators [114]:

- OO Mediators for ontology mediation,
- GG Mediators for linking goals,
- WG Mediators for linking Web services to goals and
- WW Mediators for enabling interoperability between two heterogeneous Web services.

The idea of the mediator approach is as follows: Mediator services transform instances of service parameters or (sub) goals from a source to a target ontology. Then, the mediator services are integrated as common services into the planning process, whereas the planning algorithm automatically ensures that mediator services are selected between the interoperation of heterogeneous Web services including mediation between (sub) goals. However, the main purpose of mediator services is to reconcile the differences between goals of Web services and their usage is inherently dependent on the integration into a planning process. Thus, it can be stated that it may be difficult to map this approach to non-planning oriented problems that purely focus on Web service interoperation, e.g. in process modeling, service composition and invocation [115].

Semantic Web Services Framework (SWSF)

SWSF is a further alternative approach likewise not build on OWL. The major contribution of SWSF is a rich behavioral process model based on the Process Specification Language (PSL) [116]. It aims at being a comprehensive framework that spans the full range of Semantic Web service related issues including orchestration and mediation. However, the design of the orchestration concept focuses on automated planning as well as the mediation concept, which therefore readopts the goal-based approach similar to the mediator concept in WSMO.

Web Service Semantics (WSDL-S)

In contrast, WSDL-S is a light-weight approach for Semantic Web service description. Instead of defining a comprehensive framework for Semantic Web services, WSDL-S defines inline extensions to WSDL in order to semantically annotate XML data types as well as messages and operations in WSDL descriptions. It externalizes the ontology language representation in terms of specific tags that refer to the semantic annotations and thus allows the binding to OWL. In particular, WSDL-S defines three types of annotations [117]:

- WSDL types, i.e. WSDL entities specified in terms of XSD types, are referenced to concepts of a domain ontology, including a mapping description between XSD types and the corresponding semantic model concepts.
- WSDL operations can be described by preconditions and effects in terms of referencing to respective expressions.
- Categorization information about Web services can be defined on the basis of an ontology.

Semantic Annotations for WSDL and XML Schema (SAWSDL)

SAWSDL follows the pragmatic approach of WSDL-S and provides simple semantic annotations within traditional WSDL-based Web service descriptions. While the above presented approaches have been published as W3C member submissions, SAWSDL is the only official W3C technology recommendation for Semantic Web services. The following figure illustrates the annotation of WSDL documents with additional tags that reference to a domain ontology:

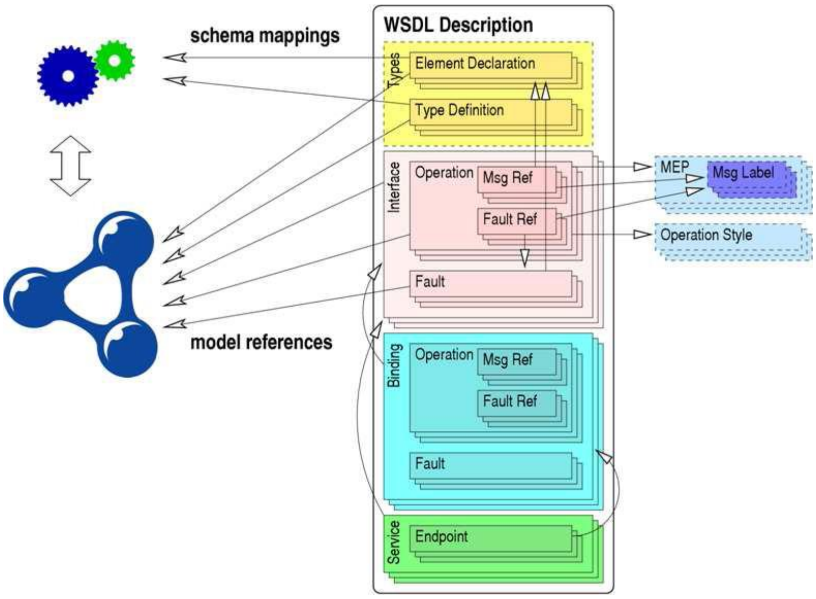


Figure 3-19 SAWSDL Overview [118]

According to the general Semantic Web Service grounding approach as well SAWSDL consists of two parts:

- Schema mappings between XSD typed XML instances and domain ontology individuals.
- Model references that point to a concept in a semantic model covering interface categorization, operation functionality, fault meaning and data type or element correspondence in an ontology.

Thereby, it is important to note that SAWSDL limits the annotation by references to only ontology concepts, so that the definition of preconditions and effects in terms of logical expression is not supported. This underlines the light-weight approach of SAWSDL in contrast to goal-based approaches such as WSMO or SWSF where specification of logical expressions for preconditions and effects is essential. Furthermore, this limitation demonstrates that the standardization at the W3C follows a non-planning approach to Semantic Web services focusing on pragmatic approaches.

3.4.3 Evaluation

A comparison of the presented Semantic Web service technologies and frameworks reveals the following commonalities and differences: The first approach, OWL-S defines a description model for Web services that includes a formal description of interfaces regarding input and output parameters as well as formalization of preconditions and effects. It uses OWL as the specification language and hence it is compliant with the W3C standards for the Semantic Web. In contrast WSMO and SWSF are not built upon OWL and apply specific ontology languages that aim at more sophisticated service interaction descriptions. In particular, WSMO and SWSF propagate a goal-based approach along with artificial intelligence-based planning algorithms, which goes beyond the basic idea of annotating Web services. WSDL-S and its successor SAWSDL take a step back and focus again just on semantic annotation of Web services and thus can be regarded as light-weight approaches. Furthermore, WSDL-S and SAWSDL strongly rely on existing Web service standards, namely WSDL, and rather focus on its extension than providing an alternative conceptual framework. Moreover, SAWSDL is the only approach standardized as a recommendation by the W3C for the semantic annotation of Web services.

Suitability of Goal-based Planning in Service-Oriented Architectures

Regarding the concept of goal-based planning applied to the Web service life-cycle in particular in the context of discovery, selection and composition its practical limitations have already briefly been discussed above. In general, the user of a Web service might have an overall goal he wants to achieve by using a Web service or a composition of Web services. In order to formalize the overall goal as required for the goal-based planning algorithms as well the problem domain needs to be formalized. This does not only comprise an ontology conceptualizing the domain but as well a formalization of the current state of the problem domain and all possible states it may take. Then a decomposition of the overall goal into a combination of formalized sub-goals referring to the corresponding states of the problem domain is required. Based on this formalization of the goal a reasoner then can be enabled to derive the corresponding selection and composition of Web services that can be invoked in order to achieve the desired overall goal. However, if the user has to perform this goal decomposition on a relatively fine-granular level to apply goal-based inferencing techniques for planning, it remains unclear whether this approach provides any advantages compared to manual plan creation performed directly by the user. Furthermore, the additional efforts for the identification and formalization of possible states of the problem domain have to be considered as well in this comparison. Moreover, classical planning problems assume complete knowledge about the problem domain including narrow and deep formalization of possible states. In contrast, distributed and heterogeneous SOA landscapes cover a broad problem space and do not allow for deep formalization of all possible states including non-deterministic side-effects [119]. Thus, it can be stated that although Web service discovery, selection and composition problems resemble planning problems, it does not seem suitable to apply goal-based inferencing techniques known from artificial intelligence research to them [120]. Accordingly, M. Stollberg concludes that the employment of the goal-based Semantic Web service approach requires a comprehensive re-design of an SOA system [121]. Moreover, J. Hendler states that even the best works in this area assume non-realistic simplifications that generally twist Web services into a planning framework [122].

Semantic Web Services for Bridging the Semantic Interoperability Gap

Having outlined the limitations of goal-based planning in SOA also the proposed concept of WSMO mediators has to be analyzed in this context. Mediator services as proposed in the WSMO framework directly address the heterogeneity of metadata formalization and thus the problem of semantic interoperability as targeted in this work. As described above they are integrated as common services into the planning process, whereas the planning algorithm automatically ensures that mediator services are placed between the interoperation of heterogeneous Web services. However, taken such mediator services out of the context of goal-based planning they just provide transformation services between parameters described by a source ontology to parameters described by a target ontology. Accordingly, in the absence of automated planning they might only be usable as basic data transformation services during runtime of Web service compositions, which furthermore have to be manually integrated and do not provide adequate support or added value at design time. However, as described in Section 2.4 the fundamental first phases in the SOA life-cycle begin with business process modeling and the design of service compositions. In particular, these phases are relevant for the alignment or mediation between heterogeneous metadata formalizations as they deal with the conceptual abstraction level of data or more precisely of information. Consequently, it can be stated that the discussed mediator concept, if taken out of the goal-based planning context, does not provide a sufficient solution to achieve semantic interoperability in SOA.

Leaving the goal-based planning approach including the notion of integrated mediators and taking a wider perspective on how Semantic Web services can contribute on bridging the semantic interoperability gap in SOA, nevertheless a significant added value compared to traditional Web service technology can be identified. Formalization and representation of Web services parameters are expressed in context of domain models in terms of ontology concepts based on description logics. Recalling again the different abstraction levels for the representation of meaning introduced in Section 2.3.2 ontologies as used in Semantic Web services provide the following characteristics: On the one hand, the triple structure of the ontology meta-data model can be regarded as well as a logical data model analogical to XML, as it provides the view applied in application development. However, on the other hand it can additionally be considered as a conceptual model, as concepts, properties and relations in an ontology allow for expressive modeling similar to languages such as the Unified Modeling Language (UML) or the Entity Relationship model (ER). Thus, it can be concluded that the ontology meta-data model is located on a higher abstraction level compared to the XML meta-data model used in traditional Web service technology. In the following Figure 3-20 this comparison is illustrated in context of the model describing the semantic interoperability in SOA:

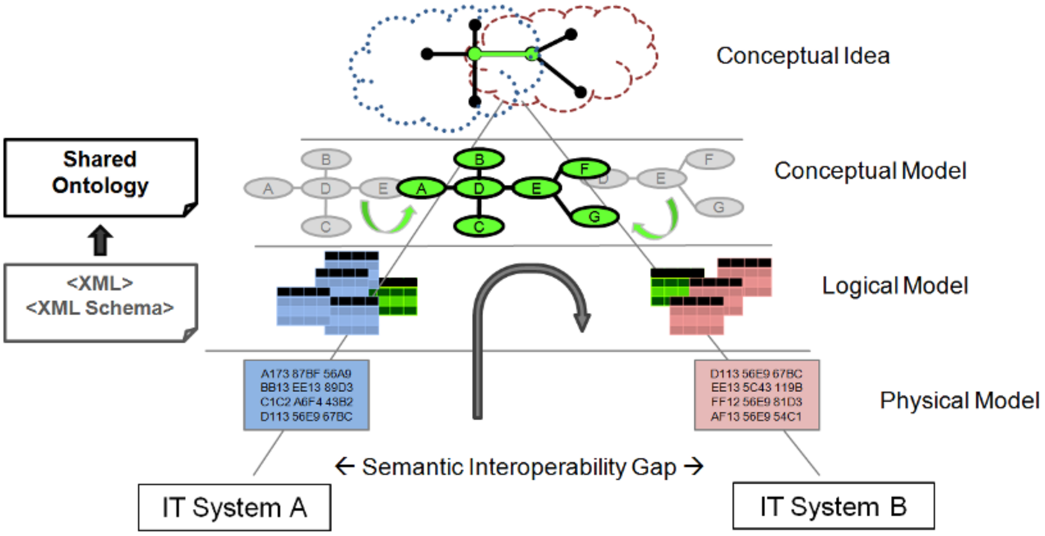


Figure 3-20 Shift of Abstraction Level using Semantic Web Services

Semantic Web service parameters are annotated by utilizing a shared ontology. Supported by a concrete Semantic Web service framework this conceptual model is then mapped to an XML Schema-based logical model by means of an integrated grounding mechanism on top of traditional XML-based Web services. Furthermore, this figure demonstrates that the general notion of Semantic Web services, even if they are provided by different IT systems, considers that a shared ontology is used for the annotation of Web service parameters. Consequently, even if the conceptual ideas of the different human designers have been diverse, on the conceptual level there are no more two different and heterogeneous conceptions but one aligned shared conceptual model. This fact provides a major advantage of Semantic Web services for bridging the semantic interoperability gap. As the lifting and lowering mechanism included in the grounding of Semantic Web services provides already the mapping between the different abstraction levels the round-trip across the semantic interoperability gap is already closed due to the shared conceptual model. Consequently, no more additional technical transformations between representation formats are necessary, which provides a significant more efficient solution compared to the traditional alternative (cf. Section 3.2.3) applying XML-based Web services including extensive technical transformation code.

However, the underlying assumption of this general Semantic Web service approach is that a shared domain ontology exists or can be developed collaboratively between the actors providing and using Semantic Web services. Unfortunately, in real world cross-domain context as given in the targeted large-scale SOA landscapes this approach of developing a globally shared ontology-based standard for information models as a kind of lingua franca has turned out limited. In particularly, organizational boundaries in community processes for the development of shared conceptualizations covering multiple domains can be identified. While once so called Enterprise Data Models have been a well-established approach, they have not yielded the results expected or required including a large base of failed projects. Empirical case studies have shown that data or information modeling cannot create new organizations or new businesses in its own image. Conceptualizations must reflect the business [123]. Therefore, due to the complexity deriving from inherent domain-specific differences in requirements, collective agreement on a shared conceptualization is often only feasible under significant limitations. These circumstances lead to the requirement for coexistence of multiple independent but however conceptually overlapping information models. Accordingly, moving back to Semantic Web services, it might be necessary to refer to multiple ontologies as well. Hence, it can be stated that in the context of large-scale SOA landscapes the dominant semantic interoperability approach of Semantic Web services is limited. In fact, it can be concluded that the ontology-based approach for Web service description only alleviates the problem in terms of lifting the abstraction level of semantic heterogeneity. However a mapping between ontologies originating from different contexts is still needed [124].

To sum up the evaluation, it can be stated that the approach of Semantic Web services contributes in achieving semantic interoperability by lifting the abstraction level of information models and thus narrows the distance for bridging the semantic interoperability gap (cf. Figure 3-20). However, the general approach assumes the usage of a globally shared ontology with the discussed limitations. Further analysis of the identified organizational boundaries and the fundamental requirement for coexistence of independently managed conceptualizations deeper elaborated in Section 4.5 which discusses context dependency of information models leading to the proposed model of semantic mediation between loosely coupled information models in SOA.

3.5 Semantic Information Integration in Related Areas

Having discussed how traditional Web service technology and Semantic Web service technology can contribute to semantic interoperability and having identified the requirement for mappings between heterogeneous conceptualizations in terms of ontologies, the next step is to further analyze the state-of-the-art in this field. As ontology mapping is a relatively young discipline and has been strongly influenced and based on previous work on semantic information integration such related areas are firstly analyzed as an intermediate step before coming back to ontology mapping.

Generally, semantic information integration, often also referred to as enterprise information integration, is required in distributed IT systems when information from disparate sources with different conceptualizations needs to be processed uniformly. Two prominent areas where this problem is targeted in particularly are distributed database systems and distributed object-oriented or component-based systems. As mentioned above before referring back to ontologies this section highlights traditional approaches in these fields: Firstly the focus is put on data integration in database systems and secondly on how semantic information integration is targeted in the open distributed processing reference model (RM-ODP) [125].

3.5.1 Semantic Information Integration in Database Systems

In database systems the semantic interoperability thematic occurs in the context of data integration between distributed databases. Two different approaches for data integration can be distinguished. Firstly, data integration can be realized by so called materialized integration, where data from different sources gets extracted, transformed and loaded (ETL) into one single data store for uniformed processing [126]. This approach is also called data warehousing and is used for data analysis in the context of enabling processes across different departments or supporting business decision making tasks. The weakness of this approach is the lack of data coherence when the original sources are updated but the single data store still contains the old data. Consequently in the case of update, ETL processing needs to be done again. Alternatively, data can just be integrated virtually by loosely coupling the different sources. This avoids the repeated ETL process but increases complexity. Instead of integrating the data physically a mediator with an integrated query interface is provided which transforms the queries to the virtual integrated database into specific queries to each original source. Considering that data is represented differently in the underlying database schemas of the original sources the different source schemas need to be mapped to a so called global schema of the virtual or materialized database. This is where semantic information integration takes place.

To define an appropriate global schema is a challenging task. The global schema needs to express the overlapping concepts from different source schemas in a uniform manner. This task is mainly done manually. However, various approaches have been developed for (semi-) automatic schema matching [127]. Such a matching can be used to define the global schema by means of matching two sources to extract the overlapping part. The matching does only cover the design time task of semantic information integration. During runtime the global queries need to be translated into queries for the local sources. For this process a mapping between the schemas needs to be defined. Ideally, a mapping is the output of an automatic schema matching.

Global-as-View vs. Local-as-View

Such mappings can be expressed by making use of so called views. Views are read only virtual tables of a data base schema composed of the result set of a query[128]. The main approaches for schema mapping are the following [129]:

- Global-as-View (GAV) requires that the global schema is expressed in terms of local data sources. The local data is stored physically according to different local schemas, whereby mappings to the global schema are provided. More precisely, to every element of the global schema, a view over the data sources is associated, so that its meaning is specified in terms of the data residing at the sources. The following figure illustrates the global-as-view approach:

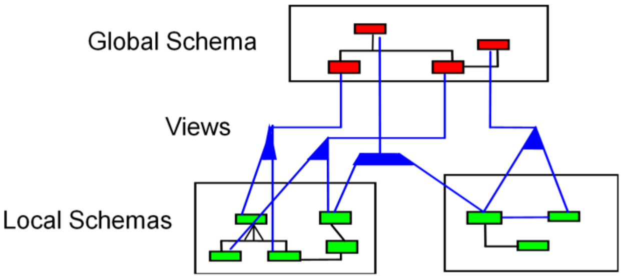


Figure 3-21 Global-as-View [130]

- Local-as-View (LAV) requires the global schema to be specified independently from the sources. In turn, the sources are defined as views over the global schema. The relationships between the global schema and the sources are thus established by specifying the data of every source in terms of a view over the global schema. The following figure illustrates the local-as-view approach:

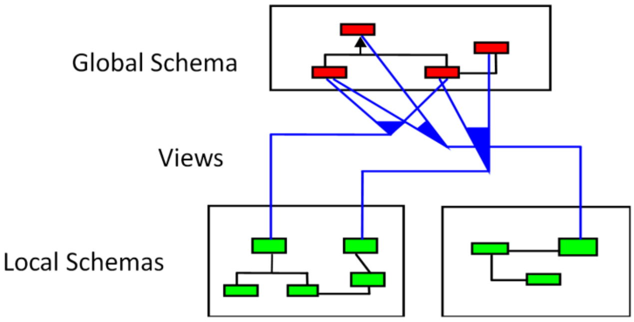


Figure 3-22 Local-as-View [130]

In the GAV approach the views need to be updated whenever a source changes or a new one is added, which is inflexible in a dynamic environment. In this regard LAV is more appropriate as the global schema remains unchanged even when sources are changed or added. However, in GAV the query reformulation task for the mediator can be performed straight forward as queries for the sources are already defined in the views. In contrast query reformulation in LAV is more complicated. Queries need to be constructed in terms of analyzing the views over the global schema, whereas the relation between entities in the global schema and entities in the local schema is only given inverse. Section 3.6 refers back to these presented approaches in database schema mapping and outlines the relation to integration approaches with ontologies.

3.5.2 Semantic Information Integration in RM-ODP

The reference model for open distributed processing (RM-ODP) [125] is a joint standard of the International Standards Organization (ISO) and the International Telecommunications Union (ITU). RM-ODP offers a conceptual framework and an architecture that integrates aspects related to distribution, interoperation and portability of software systems, in such way that hardware heterogeneity, operating systems, networks, programming languages, databases and management systems are transparent to the user. In this sense, RM-ODP manages complexity through *a separation of concerns*, addressing specific problems from different points of view [131]. It targets a comprehensive approach and aims at being a coordinating framework for any current and future standards in the field of open distributed systems. As stated in Section 2.4 as well various fundamental concepts of SOA are already mentioned in RM-ODP.

However, one of ODP's fundamental concepts is the use of a common object model, thus following the object-oriented paradigm [132]. Software components are modeled as objects that interact via interfaces with other objects. These objects can be remote objects and run each on different machines. Therefore, interactions are realized in terms of remote procedure calls. This context might explain why RM-ODP has received much attention in the context of object-oriented distributed systems and especially the various standards related to the Common Object Request Broker Architecture (CORBA). In contrast to the context of recent Web service developments, in which RM-ODP is rarely mentioned, although most fundamental conceptual approaches of Web services have already been identified in RM-ODP [133].

Information Viewpoint and Trading Function

Another fundamental concept of RM-ODP is the specification of a distributed system in terms of viewpoints. Besides the enterprise, the computational, the engineering and the technology viewpoint, RM-ODP defines the information viewpoint, which focuses on the semantics of information and their processing. It describes the information managed by the system and the structure and content type of the supporting data [134]. One of the common functions on which RM-ODP gives outline definitions is the trading function, which targets in particular the information viewpoint. In general, the trading function provides a centralized service for discovery, binding and interaction between different objects by making use of attribute-based descriptions, e.g. security policies or service advertisements. The foreseen usage of the trading function in the ODP framework is to support inter-object communication via interrogations and announcements. The communication methods require, besides the trading function, also the support of a type repository, channel creation (binding) and service invocation functions. The type repository function stores information about abstract types and their concrete representation forms, in order to support translations between the service representations. The binding function sets up a channel between the objects. It requires information from the trading function and the type repository. When the binding has been established, the invocation controller can activate a service request on behalf of the client. The communication scheme is illustrated in the following figure:

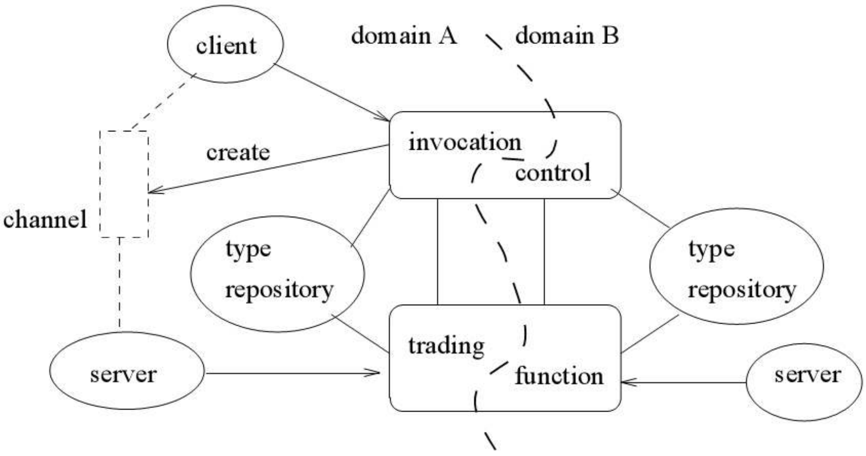


Figure 3-23 RM-ODP Inter-Domain Communication Architecture [134]

Hence, during discovery and interaction, the trading function transparently integrates the information from type repositories for the mediation between concrete representation forms in different domains. Besides others the type repository therefore offers operations for [135]:

- Publishing realizations of abstract types,
- Checking whether two type realizations are conformant and interchangeable,
- Retrieving subtypes or supertypes of a type realization and
- Translating one type realization to another.

The RM-ODP does not provide a concrete specification on how the actual translation of type realizations should be performed. However, it defines the architecture how the trading function and the type repository interact for mediation. Similar to the mediator approach in data integration (cf. Section 3.5.1) a specialized service between source and requestor, here the trading function together with the type repositories, performs the mediation. RM-ODP also describes architectures where each domain uses a distinct trader also referred to as multiple trading domains. In such architectures designed for heterogeneous platform integration, traders interact with one another by instantiating proxies to services of another trader. The proxy needs to be supervised by a so called interceptor [136]. The interceptor interprets the routed data and performs the translation of representation formats by interaction with type repositories as discussed before.

In contrast to data integration in context of database systems there is no distinction between a global and local perspective of metadata. Rather type repositories which store metadata are considered as equal each dedicated to its corresponding domain. However, it is possible to regard one specific type repository as global compared to other local type repositories. Hence, the integration from many local to one global perspective can be modeled within RM-ODP. Furthermore, it can be stated that the concept of semantic mapping between heterogeneous type systems is already identified within RM-ODP. However, the approach of the trading function focuses on the runtime level and does not explicitly include mediation support during design time such as required for modeling service compositions based on heterogeneous conceptualizations.

3.6 Semantic Information Integration with Ontologies

In recent years the use of ontologies for semantic information integration has received much attention. Prominent Semantic Web scientists such as Uschold and Grüninger regard semantic interoperability as a key application of ontologies [137]. In this sense, ontologies have evolved as a promising approach to preserve the precise meaning as data is moved from one IT system to another. However, as discussed in Section 3.4.3 about the limitations of Semantic Web services the dominant approach of developing globally shared ontology-based standards for information models has turned out limited in real world cross-domain contexts. Accordingly, the discipline has evolved from studying single ontology approaches to multiple ontology approaches including mappings between them. Taking this into account, ontology mapping has emerged as a central research field and is often regarded as the Achilles Heel of the Semantic Web [138].

Basically, the application of ontologies for semantic information integration can be distinguished into three different strategies as illustrated in the following Figure 3-24:

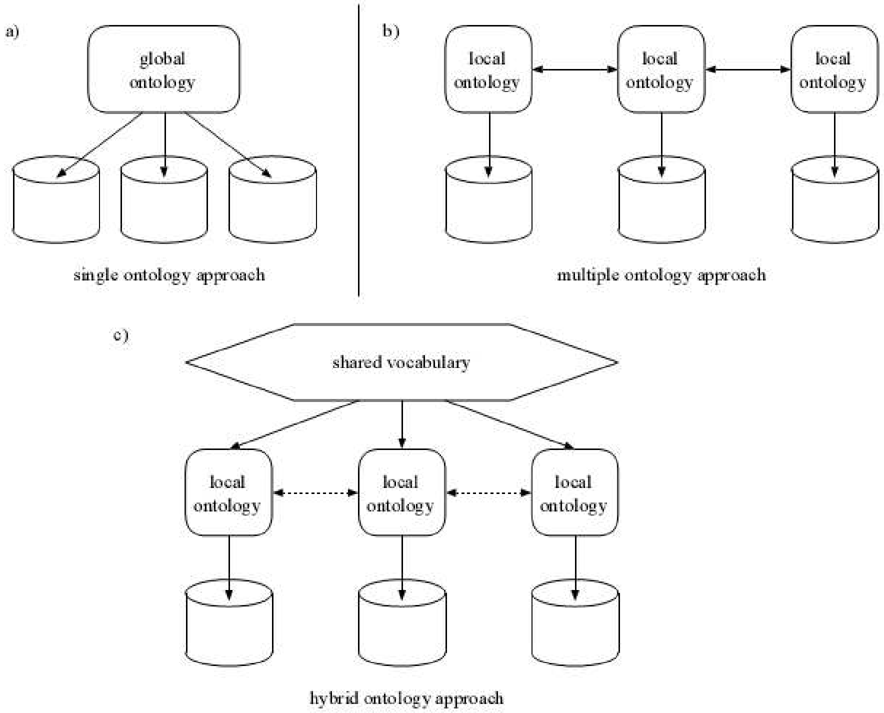


Figure 3-24 Three Ontology-based Semantic Integration Strategies [139]

In the following sections the different strategies are presented and further discussed.

3.6.1 Single Ontology Approach

The most traditional approach is using an ontology representing a global view on various different data sources. This approach follows the core idea of an ontology as a shared specification of a domain conceptualization. Information entities in each source need to be related to one concept in the global ontology. This relation can be preprocessed and the information can be stored centrally in an ontology representation. Another possibility is to express the relation in terms of query reformulation similar to the global-as-view approach in data integration outlined in Section 3.5.1, so that queries to the global ontology are delegated to the different sources during run time.

However, it is not always sufficient to target semantic information integration by mapping different information representations to a single global ontology. As discussed before, from a realistic point of view there is no single ontology describing one domain, rather there exist many different. Different purposes materialized in different applications which follow their own conceptualizations force e.g. different granularity requirements on ontologies. Thus, different domain ontologies may overlap and partly model the same information space, however from different perspectives.

3.6.2 Multiple Ontology Approach with Ontology Mapping

These constraints have led to the multiple ontology approach. Hence, ontologies can be developed independently and provide the accurate level of granularity required for their specific purpose. Semantic information integration is then realized by mappings between different ontologies. These mappings define relations between concepts in different ontologies. In [140] ontology mapping is defined the following way:

Given two ontologies O1 and O2, mapping one ontology onto another means that for each entity (concept, relation or instance) in ontology O1, we try to find a corresponding entity, which has the same intended meaning, in ontology O2.

The example shown in Figure 3-25 illustrates a mapping between two ontologies. The mapping between the two ontologies is marked with dashed lines:

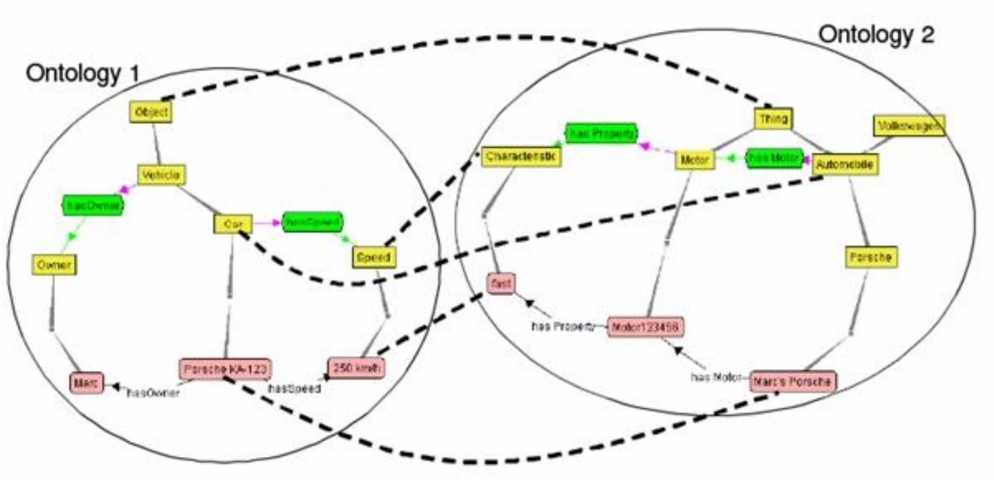


Figure 3-25 Example Ontologies with Mappings [140]

The process of mapping two different ontologies is generally divided into four basic tasks [140] as illustrated in the following figure:

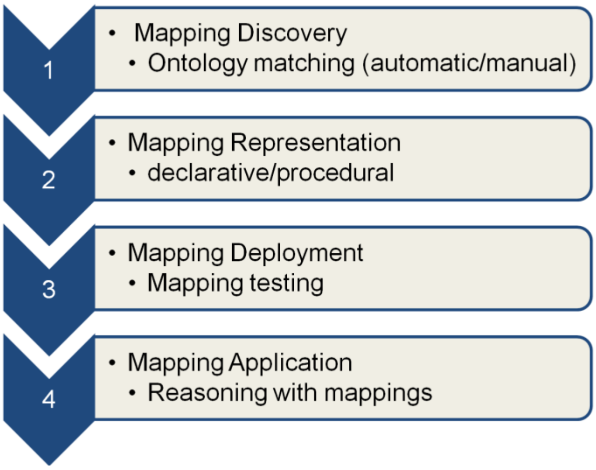


Figure 3-26 Basic Steps in Ontology Mapping

Mapping discovery or also named ontology matching targets the question on how to identify the correspondences between semantically related entities within two given ontologies. Matchings can be found between two ontology objects; then it is called a one-to-one matching. If one object represented in ontology A is represented as two or more objects in another ontology B, then it is called a one-to-many matching. Consequently, there are also many-to-many matchings, if there is any correspondence of aggregated objects in different ontologies. The process of mapping discovery is a complicated task and has been an active research field in recent years. The discipline is closely related to schema matching as known from database integration. In this context various approaches have been analyzed in [127]. Such work has provided substantial foundations for current research in ontology matching. More recently, in [141] various ontology matching techniques have been surveyed, classified and contextualized to application scenarios. Basically, such ontology matching techniques are based on particular characteristics of ontologies. For example this includes structural measures based on graph-theory or lexical as well as statistical similarities which are evaluated to find correspondences between concepts. Moreover, machine-learning methods can be applied that learn how to sort alignments through the presentation of many correct alignments in a learning phase in terms of positive examples [141].

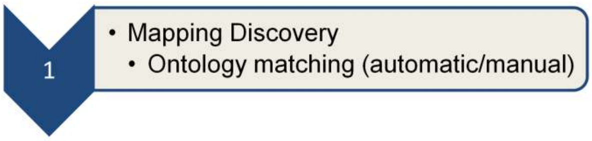


Figure 3-27 Step 1 of Ontology Mapping: Mapping Discovery

Although encouraging results are obtained, this problem is by no means solved and automatically obtained results are not yet good enough in terms of recall and precision [142]. Hence, the analysis turns out that most approaches for automatic ontology matching still require human intervention to generate sufficient results and thus can be considered as semi-automatic matching and heuristic based. A semi-automated mapping process means that a tool proposes a possible mapping to a user and the user has to validate and complete it [143]. Indeed, for finding the correspondences between concepts, it is necessary to understand their meaning. Besides the represented meaning described by model-theoretic semantics, the ultimate meaning of concepts is only in the head of the people who developed those concepts and accordingly the final matching decision can only be performed by them [144]. Thus, it can be stated that the need for user involvement seems to be a natural limitation to automatic ontology matching approaches.

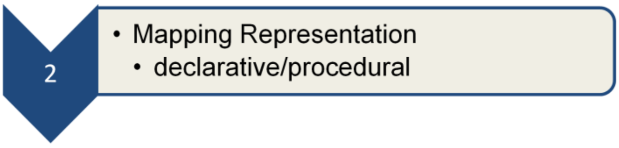


Figure 3-28 Step 2 of Ontology Mapping: Mapping Representation

After having found matching concepts and properties the matching results, i.e. the identified mappings need to be formally defined. On the one hand, the alignment can be utilized to merge the data or conceptual models into a newly created so called merged ontology. This approach basically refers back to the single ontology approach taking into account update of distributed information sources and ontology evolution. On the other hand keeping the approach of multiple ontologies the translation of instances according to the identified matching results need to be considered. In order to express the translation, generally three approaches can be distinguished:

- Views and Queries
- Mapping Ontology
- Bridging Axioms

Using views and queries for describing mappings is similar to approaches discussed in the context of data integration outlined in subsection 3.5.1. D. Calvanese et al. demonstrate in [145] how view-based query answering known from data integration can be applied to ontology mapping. A global schema or ontology is used and queries to the global conceptualization are rewritten in terms of queries to local ontologies. The results are aggregated according to the view mechanism. This approach can also be applied in the context of two or more ontologies that are used on the same level in a peer-to-peer manner.

Another approach for describing mappings is to define a specific mapping ontology which expresses the different relationships between corresponding concepts and properties. A. Maedche et. al. present in [146] a mapping framework for distributed ontologies (MAFRA), for which they have developed a dedicated so called semantic bridge ontology. In order to process such a mapping, a specific inference engine needs to be provided which performs the mapping and enables further reasoning with mapping results.

The third way is to describe the mapping by a set of bridging axioms which refer to concepts or properties of a source ontology and specify how to express them in a target ontology. Thus, bridging axioms can be realized as description logic-based rules which describe the transformation. The advantage of this rule-based approach is that reasoning over the described mappings can be applied straight forward as the mapping rules can be integrated into the regular ontology inference process, such as classifying etc. For example Deijing Dou et. al. have developed an ontology mapping tool OntoMerge [147] based on bridging axioms. It mixes a source and target ontology with bridging axioms to form a so called merged ontology. Individuals expressed in the source ontology are then inferred by bridging axioms to individuals which are additionally expressed in the target ontology. Finally, all source ontology constructs are removed, which is called projection and individuals purely expressed in the target ontology are the result of the transformation. The ontology mapping approach instantiating the concept of loosely coupled information models developed in this work also applies rules to transform concepts from one ontology to a representation in another. Thus, Chapter 4 refers back to OntoMerge and outlines similarities and differences with the presented approach.

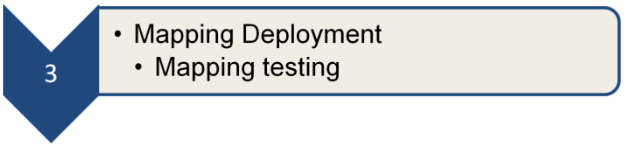


Figure 3-29 Step 3 of Ontology Mapping: Mapping Deployment

Mapping discovery and mapping representation mainly concerns activities during design time of the information systems life-cycle. Moreover, these activities are addressed from a domain perspective by focusing not only on a particular application but rather on a more comprehensive scope targeting an entire application landscape. Consequently, domain experts model their respective domain in terms of ontologies and collaborate, in order to define the mappings between them according to the multiple ontology approach. However, this perspective does not necessarily match the perspective of a concrete application that may only cover partly any particular domain but however addresses multiple ones in terms of cross-organizational processes. In fact, different stakeholders are involved and have different responsibilities. On the one hand, domain experts define ontologies and respective mappings between them as producers and on the other side, application or process experts utilize such ontologies and ontology mappings as users in concrete application scenarios.

Consequently, the deployment of ontology mappings needs to support multi-stakeholder processes taking into account mechanisms for publishing facilities, access and feedback channels as identified in [148]. Despite the increasing interest shown from various communities in which ontologies have been used, there is still a lack of such tools to facilitate deployment and maintenance of ontologies [149]. Some approaches have been identified in [150] and aspects and requirements of a collaborative, community-oriented ontology server have been described. In particular, trust in the developed ontology mappings needs to be ensured when mappings are passed from domain experts to stakeholders focusing on usage in concrete application scenarios. Therefore, measures for validation and correctness as addressed in testing of ontology mappings have to be considered in the deployment process.

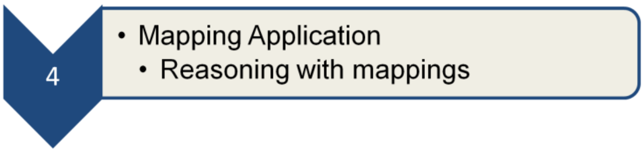


Figure 3-30 Step 4 of Ontology Mapping: Mapping Application

The ultimate goal of ontology mapping is to use the identified and formally defined mappings in multiple ontology-based application contexts. The resulting mappings are used for various integration tasks: data transformation, query answering, or Web service composition, to name a few [124]. For instance, it should be enabled to ask queries using the vocabulary of one ontology and receive answers that do not only consist of instances of this ontology but also of ontologies connected through ontology mappings [151]. Accordingly, in order to process the ontologies the reasoning on them has not only to cover the ontologies themselves but as well the mappings between the ontologies. That means that besides the relations between concepts and properties within an ontology additionally inter-ontology relations between concepts and properties defined in the mappings need to be integrated into the inferencing mechanism. Thereby, the inferencing mechanism strongly depends on the way the mappings are formally expressed as outlined above.

Regarding ontology mappings expressed in terms of database-like views the reasoning is performed in a lazy-evaluation style. I.e. the reasoner does not process the whole knowledge base including source and target ontology as well as ontology mappings but only gets active when queries need to be processed. Then the reasoner interprets the defined views in order to derive rewritten queries according to the underlying information source. The result is then presented homogeneously in terms of the reference ontology. In case of a dedicated mapping ontology that represents the mapping, a specific inference engine has to be provided in order to interpret the declarative mapping. Such a mapping interpreter translates instances that conform to a source ontology to instances conforming to a target ontology. Further reasoning such as subsumption, classification or constraint checking can then be performed on the resulting unified instances. Regarding bridging axioms-based ontology mapping the reasoning process is quite similar. However, taking into account that bridging axioms can be expressed in terms of description logic-based rules no specific mapping interpreter is necessary. Reasoning over the described mappings can be applied straight forward as the mapping rules can be integrated into the regular reasoning process. I.e. besides reasoning according to the underlying rules defining the ontology language semantics as well the rules that define the ontology mapping are handled by the same inference engine.

According to these four basic steps ontology mapping enables an information architecture for semantic integration that is not dependent on one globally shared ontology. The multiple ontologies are linked together by means of ontology mappings and thus enabling information exchange across heterogeneous sources without any semantic centralization.

3.6.3 Hybrid Ontology Approach

Besides the single ontology and multiple ontology approach H. Wache et al. [139] also describe a hybrid approach which combines the before mentioned. Basic common concepts which can be equally used in different local information sources without any restrictions are merged and expressed in a shared vocabulary. Then concepts of local ontologies can be derived from the shared vocabulary, which can be as well considered as a top level ontology. However, the globally shared vocabulary is limited strictly to basic concepts that are not affected by specific local requirements. Context-related concepts capturing different perspectives, coverage or different granularity of local information sources are defined independently in local ontologies. Consequently, such local ontologies are then treated according to the multiple ontology approach with mutual mappings between them.

Having identified ontologies as a suitable means for semantic information integration, three different ontology-based integration strategies can be distinguished. The single ontology approach features a straight forward implementation effort but does not support heterogeneous viewpoints on different information sources. This is the major benefit of the multiple ontology approach that enables different perspectives on heterogeneous information sources combined with independent management of separate ontologies. The hybrid ontology-based integration strategy compared to the two other approaches combines the advantage of supporting heterogeneous perspectives with providing a common base of generalized basic concepts that are shared among the different ontologies. This enables a separation of concerns where common aspects can be shared and at the same it ensures the independency of conceptualizations where different perspectives are required [139].

3.7 Summary and Reflection

This chapter has analyzed the conceptual and technological state-of-the-art in achieving semantic interoperability of heterogeneous systems with particular focus on SOA.

The integration of information is at the heart of semantic interoperability. Hence, an analysis of different approaches in this area has been carried out covering the main strategies in context of traditional purely XML-based Web services, Semantic Web services and related areas such as information integration in distributed database systems and distributed object-oriented systems.

It has turned out that integration on the conceptual level is more efficient and effective than integrating information models which are represented on the more technical logical data model. The formal definition of concepts and their relationships allows for increasing the efficiency of the integration process by lifting it to a higher abstraction level and thus reducing manual semantic interpretation and technical transformations between differences in information representation.

Consequently, a thorough survey of ontology-based Semantic Web services technologies and concepts and corresponding semantic integration approaches has been performed and their advantages have been outlined in contrast to the analyzed capabilities of traditional purely XML-based Web services.

In order to further substantiate the analysis, the shift of abstraction level has been pointed out within the framework of semantic interoperability in SOA developed in Chapter 2. Consequently, a reduction of the semantic interoperability gap (cf. Section 3.4.3) could be shown for semantic integration on the conceptual level compared to semantic integration on the on the lower abstraction levels for information representation.

Furthermore, it has been identified that due to the nature of large-scale SOA landscapes a single globally accepted conceptualization covering all requirements on information models of involved actors in cross-organizational processes has limited practicability. Rather, there exist several conceptualizations which fully or partly cover a given domain. It has been outlined that even in case of relatively distinct identified domains; organizational reasons lead to the development of competing ontologies which thus have to be considered in realistic application scenarios. This is where the multiple ontology approach for semantic integration based on ontology mapping comes into play. Hence, a number of ontology mapping approaches and exemplary tools have been investigated.

In the following Chapter 4, this analysis of heterogeneous conceptualizations is refined and the central artifact of this work, the concept for semantic mediation between loosely coupled information models in SOA is presented.

Chapter 4

Semantic Mediation between Loosely Coupled Information Models in SOA

4.1 Overview

This chapter presents the core conceptual approach of this work. Based on the problem identification and the analysis of the state-of-the-art for achieving semantic interoperability in SOA, a concept for semantic mediation between loosely coupled information models is presented.

The chapter starts by concretizing the requirements already briefly introduced in Chapter 1. Subsequently, the general idea of the developed concept is outlined to provide an overview of the central aspects. These include mainly the shift from monolithic to loosely coupled information models combined with the approach to address semantic integration on a higher abstraction level. The following sections then refine the general idea and provide detailed descriptions and argumentations of the respective conceptual parts.

Firstly, the limitations of standardization with regard to semantic interoperability are pointed out. Then, the underlying reason for context-dependency of information models is deeper analyzed by referring to a model theoretic approach. Based on these findings, the transfer of the principle of loose coupling to the semantic level is discussed and specified.

As the main goal of this work is to contribute to a solution for the semantic interoperability problem in SOA with particular regard to effectiveness and efficiency, the subsequent section reflects the proposed conceptual solution on this regard including a proposed balance between these as competing identified sub-goals.

Furthermore, in order to fulfill the requirements for loose coupling of information models in SOA and enable the proposed concept, a semantic mediation mechanism based on Semantic Web concepts in terms of ontologies and description logic rules is described and specified.

Finally, a conclusion and reflection of the chapter is provided summarizing the main outcomes and relating them to the overall thesis goals.

4.2 Conceptual Goals and Requirements

Starting from the demand to achieve interoperability in IT-supported business processes across intra- and inter-organizational boundaries, the semantic dimension has been identified as critical. The research hypothesis already outlines the direction of the proposed solution. In order to refine what is meant by an effective and efficient approach for achieving semantic interoperability in SOA, the following conceptual requirements or goals are listed:

- The complexity in semantic integration should be decreased by separating technical issues from business issues. This should result in a reduced demand for overlapping skills of involved stakeholders. These skills include technical expertise, expertise in different business domains that are affected by cross-organizational processes and as well the respective business process expertise itself.
- Heterogeneity and differences in business requirements and organizational boundaries should be respected. In particular, it should be anticipated that conceptualization and information modeling is strongly bound to organizational structures including the scope of influence and the feasibility of community processes to ensure agreement and standardization of information models. Therefore, organizational independence should be reflected in an information architecture for large-scale SOA landscapes.
- In a concrete semantic integration scenario, i.e. the realization of a cross-organizational business process, the status quo of complicated and highly technical transformation coding for bridging heterogeneous information representations should be overcome. Furthermore, recurring manual efforts for integrating these transformations in the specific application or process context should be consolidated and automation for these tasks should be improved.
- As differences in information model representations can be complex, the mediation mechanism should be able to cover that complexity in terms of completeness and should as well remain easy to maintain by domain experts.
- The developed conceptual approach should remain consistent to best practice SOA methodologies. According to this, it should be based on process orientation starting with business analysis leading to process models, followed by deconstruction into building blocks in terms of services, which can be independently realized and maintained based on standardized interfaces and service integration infrastructures. In particular, this implies that no opposing approaches are followed such as artificial intelligence motivated attempts that try to substitute the human business analyst and process modeler in terms of automating the design of processes with planning algorithms (cf. Section 3.4.3).
- The approach should rely on existing concepts and standards of the World Wide Web, as it constitutes the dominant IT infrastructure for cross-organizational interaction. Accordingly, technological path dependency should be considered and thus it has to be ensured that any proposed solution can be based upon existing technology.

As described in the introduction, the methodology of this work is aligned to the approach of design research (cf. Section 1.3), which is motivated by improving the state-of-the-art in terms of solving practical problems. Accordingly, the above determined requirements share the focus on practical relevance and business suitability.

4.3 General Idea

Considering typical cross-organizational processes in large-scale SOA landscapes, not only two actors as in a classical integration scenario need to be integrated but rather multiple actors have to be connected within process chains and process networks. To tackle the complexity of such integration scenarios the principle of loose coupling has been evolved as the concept of choice originating from component-based architectures and it has been well reflected in SOA as a fundamental paradigm. However, when it comes to the semantic level as presented in the analysis of the state-of-the-art, the dominant approach aims at developing commonly shared conceptualizations in terms of standards for information models.

From an architectural point of view focusing on the semantic level this constitutes a monolithic approach that stands out contrary to the general strategy tackling complexity in terms of loose coupling. The advent of semantic technologies and particularly ontologies as means for generating explicit formal vocabularies to preserve the precise meaning in information exchange can be seen as the same attempt considering the shared nature of ontologies. But taking into account organizational boundaries to achieve shared agreement among stakeholders, the feasibility of this approach has turned out limited in real-world cross-domain context.

Based on this analysis, the first part of the general idea is to transfer the concept of loose coupling to the semantic level and move from monolithic to loosely coupled information models on domain level as illustrated in the following figure:

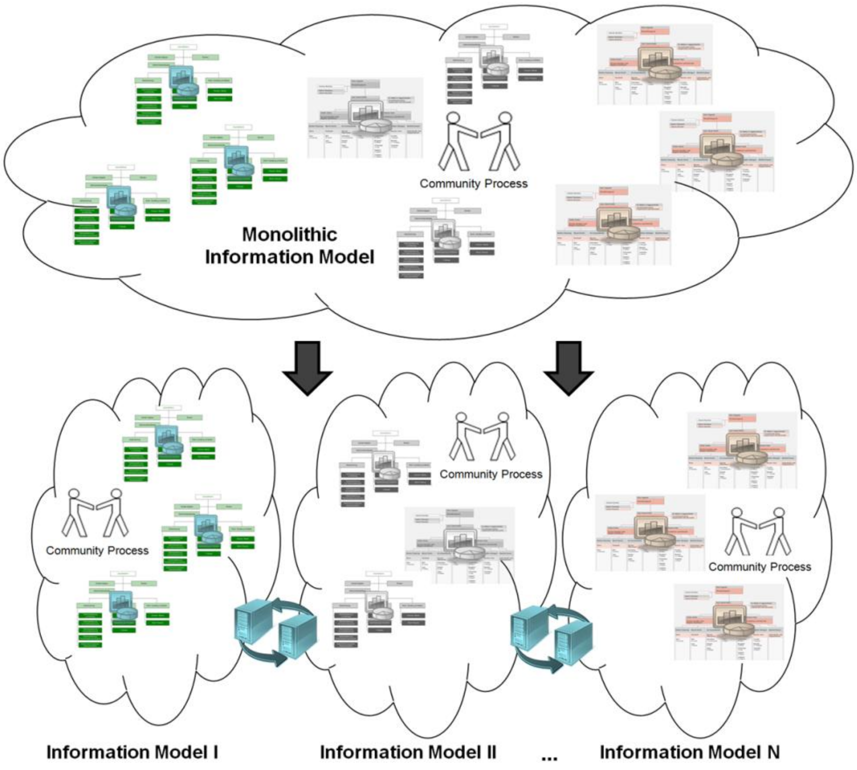


Figure 4-1 From Monolithic to Loosely Coupled Information Models on Domain Level

The approach preserves the basic idea of an ontology as a shared conceptualization but limits the scope of its application to an organizational feasible extent. Accordingly, common information models are established on the so called domain level, where community processes aiming to reach shared agreement about conceptualizations are bound to stakeholders belonging to the same domain. In this context, domain refers to a group of organizations or organizational units represented as organigrams in Figure 4-1 that share common requirements and exhibit the capability for effective collaborative decision making.

In order to achieve semantic interoperability across domain borders as required in cross-organizational processes, the independent and heterogeneous information models need to be loosely coupled in terms of so called semantic bridges. Such semantic bridges enable the mediation between domain-specific differences and ensure a coexistence of independently managed even possibly conceptually overlapping information models.

The scope of loosely coupled information models on domain level takes into account the aim for an efficient solution, as not every single organization develops and maintains their own information model but separates this task to specific domain experts. Consequently, also the mediation between different information models, i.e. different representations of overlapping conceptualizations, is only done once on domain level instead of performing it repeatedly on the concrete application level for each single cross-organizational process.

Moreover, efficiency is addressed by shifting the semantic integration task onto a higher abstraction level and thus reducing technical efforts for the mediation between heterogeneous information models. The following Figure 4-2 illustrates the shift of semantic integration with loosely coupled ontologies based on the framework of semantic interoperability developed in Section 2.5:

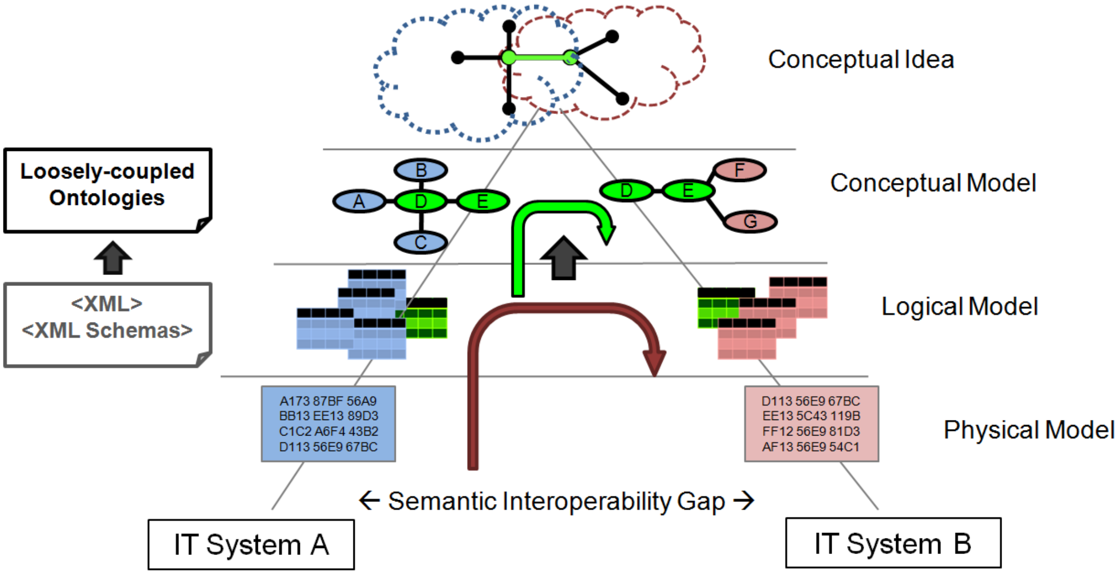


Figure 4-2 Shift of Semantic Integration with Loosely Coupled Ontologies

As described in the state-of-the-art analysis, traditional Web service descriptions are based on XML and XML Schema to specify information models including meta-data of services (cf. Section 3.2). Inherently, due to the limited semantic expressiveness of XML and XML Schema, technically complex transformation procedures are necessary to perform the round-trip across the semantic interoperability gap.

Aiming at overcoming this drawback, the second part of the general idea is to shift the abstraction level of semantic integration and thus reducing the semantic interoperability gap. Thereby, the gap is not bridged directly by using the same conceptual model in terms of a shared ontology as applied in classical Semantic Web service approaches. However, the gap is narrowed as heterogeneities can be overcome less technically based on the conceptual abstraction level of information models. The round-trip across the semantic interoperability gap then can be performed by utilizing rules-based declarative description logic, which enables a mechanism for loose coupling between the heterogeneous and independent domain ontologies describing the Web services.

To summarize the general idea, it can be stated that the particular aim for effectiveness and efficiency regarding a solution of the semantic interoperability problem in SOA is targeted the following way:

- Effectiveness is targeted by anticipating that reaching agreement about shared conceptualizations between multiple heterogeneous stakeholders involved in cross-organizational processes is often beyond de facto organizational capabilities. In order to address the complexity resulting from heterogeneous requirements and perspectives, the central idea is to transfer the concept of loose coupling to the semantic level and consequently move from monolithic to loosely coupled information models.
- Efficiency is targeted by addressing semantic integration of different information models resulting from the loosely coupled approach on domain level rather than repeatedly on the concrete application level for each single cross-organizational process. And secondly, efficiency is targeted through the shift of abstraction level for semantic integration. Instead of overcoming the heterogeneities of different information models on a rather technical level, efforts and complexity are reduced by performing the mediation between heterogeneities on the more business-oriented conceptual level.

The following sections further refine the general idea and provide detailed analysis and argumentations of the respective conceptual parts outlined above. The requirement for loose coupling of information models in SOA landscapes is examined in Section 4.4, which further discusses the limitations of standardization for semantic interoperability in SOA and Section 4.5, which relates this analysis to the context dependency of information models. In Section 4.6, the core concept of loose coupling on the semantic level is deeper analyzed and proposed as a design principle for the architecture of information models in SOA landscapes. Then, Section 4.7 reflects on an identified trade-off between efficiency and effectiveness and explains why positioning loose coupling on the domain level provides an appropriate balance between these as competing identified sub-goals. Finally, Section 4.8 presents and specifies a concrete mediation mechanism based on semantic bridges for loose coupling of domain ontologies that builds upon declarative rules-based description logic.

4.4 Limitations of Standardization for Semantic Interoperability in SOA

Basis of all IT systems that support business processes are information models which describe a certain domain and try to capture the relevant information that needs to be processed in a certain organizational context. Information models are the central artifact when it comes to semantic interoperability. Different systems and actors need to understand and interpret information models in the same manner to ensure that the information is processed consistently between the communication partners. The dominant top-down approach is to address this issue with semantic standardization so that conceptualizations are consistently shared among all participating stakeholders. This section discusses the possibilities and limitations which come with this approach and tries to derive the adequate scope and extent of standardization regarding information models.

4.4.1 Standardization vs. Mediation

As already outlined in Section 2.3.3 which has introduced the framework describing the semantic interoperability gap, there are generally two options for achieving semantic interoperability between distributed IT systems in SOA landscapes:

1. Standardization of information models, i.e. solving semantic heterogeneity by forcing homogeneity
2. Translation or transformation between different information models, i.e. solving the semantic heterogeneity by mediation

The following figure illustrates the two general strategies:

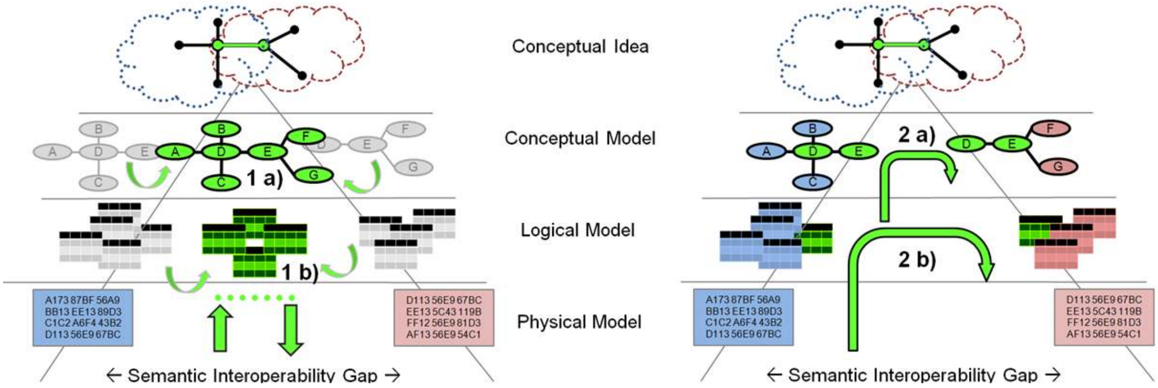


Figure 4-3 Semantic Standardization vs. Semantic Mediation

Standardization of information models (as illustrated on the left hand side in Figure 4-3) can be carried out either on the conceptual level (1a) or on the logical level (1b). On the conceptual level (1a) standardization can be established e.g. in terms of UML described information models or in terms of ontologies as applied in the context of so called Semantic Web services based on a shared domain ontology (cf. Section 3.4). For example the international standard “United Nations Electronic Data Interchange For Administration, Commerce and Transport”, builds upon the UML-based UN/CEFACT's Modeling Methodology (UMM) for capturing collaboration patterns and messages between enterprises [152]. Alternatively or in addition, standardization can be addressed as well on the logical level (1b) for instance in terms of XML-Schema-based standards. For example the RosettaNet industry standards [153] or the OASIS standardized Universal Business Language (UBL) [154] instantiation of the ebXML Core Components Technical Specification [155] are XML-based. Both cases (1a) and (1b) bridge the semantic interoperability gap by virtually narrowing it to zero as the representation of information models used by the different communication partners is the same. This can be either achieved directly in terms of harmonizing internal information models of involved IT systems or indirectly by means of a so called lingua franca used as a single reference standard for exchanged service messages.

The other strategy for achieving semantic interoperability abstains from a shared standardized information model and employs mediation between the heterogeneously modeled IT systems as illustrated on the right hand side in Figure 4-3. Consequently, a kind of intermediate translation has to be performed and it needs to be assured that the meaning is preserved adequately. Accordingly, on the logical level of information models (2 b) this mediation can be addressed in terms of transformation procedures between different schemas for instance by using the industry standard XSL transformations as part of XML-based Web service compositions (cf. Section 3.2.3). Furthermore, symmetrically to the before discussed cases, it is possible to bridge the semantic interoperability gap already on the conceptual level (2 a). This approach can be achieved by applying ontology mappings (so called semantic bridges) which are discussed in more detail later in Section 4.8.

Moreover, a combination of the two general strategies is possible. Such a hybrid approach combines semantic standardization with semantic mediation (cf. Section 3.6.3). On the one hand, basic common concepts which can be equally used in different contexts without any restrictions are merged and expressed in a shared standardized information model. However, on the other hand, the shared and standardized information model is strictly limited to basic concepts that are not affected by specific local business requirements. Context-related concepts capturing different perspectives, coverage or different granularity of local information models are defined independently and consequently semantic mediation between them is required.

This hybrid strategy is also targeted in this work. Nevertheless, the focus is put on the mediation part as the non-context dependent part of information models can be observed as rather low as analyzed later on in Section 4.5 about context dependency of information models. Moreover, later on this chapter further describes how to perform such semantic mediations between different heterogeneous conceptual information models exploiting ontology mappings as introduced in Section 3.6.2. However, before presenting deeper the concrete developed approach, firstly the argumentation line is provided that leads from the limitations of semantic standardization-based strategies to the pillars of the aimed solution.

4.4.2 From Technical Standards to Semantic Standards

The advent of SOA and widely accepted Web service standards has benefited technical interoperability in recent years substantially. Taking this into account standardization has also been regarded as the key towards semantic interoperability within SOA. However, standardization brings both advantages and disadvantages. In the following this section analysis some factors in this context in order to discuss whether or to which extent the success regarding technical standardization in SOA can be transferred to the semantic level.

On the one hand, Web services are still an emerging technology and are subject of ongoing innovation resulting in redevelopment and technological adjustment. In such a context standardization is often seen as hampering advancement or trailing behind the market. On the other hand, standardization remains the absolute condition for progress in virtually all fields of technology and especially in information technology [156]. Thereby, a multitude of dimensions can be considered. These include economics of standards, standardization policies and intellectual property rights, an aligned overall development process including actors and their rights, roles and responsibilities and a political dimension about what is socially desirable and where consensus might be reached. The last two aspects also needs to address the limiting factor of standardization in terms of optimizing that the output of standardization is promptly and manages to address the actual market's needs [156].

With regard to SOA and Web services such factors of standardization has led to platform and vendor independent interface and interaction specifications for distributed IT systems. Economies of scales in terms of increased productivity by rationalization could be realized on vendor side regarding investment protection as well as on user side taking into account that by adopting Web service standards just one type of technical interface specification needs to be supported instead of multiple ones following different paradigms. On the micro-level for the end user this results in lower integration cost and faster integration cycles by minimized risk.

The following figure illustrates the reduced complexity resulting from broad adoption of Web services standards:

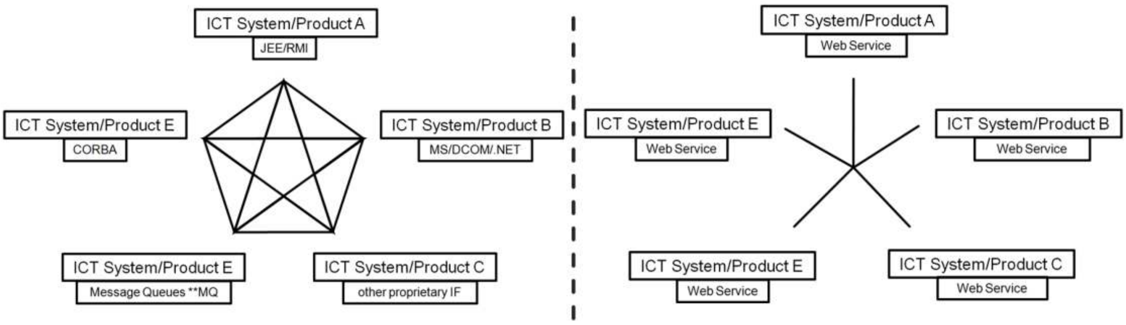


Figure 4-4 Integration of Multiple Interface Technologies vs. Web Service Standards

On the one hand, integration complexity of N actors potentially increases by N^2 as all combinations between different interface technologies have to be taken into account and a classical point-to-point integration is necessary (as illustrated on the left hand side of Figure 4-4). On the other hand, if all actors consistently adopt Web services standards, integration complexity remains linear as each actor just has to consider one interface technology for all cooperation partners (as illustrated on the right hand side of Figure 4-4).

Regarding the overall standardization process of Web services it can be noticed that as characteristically in information technology traditional government-driven standardization organizations did not play a central role. In contrast, private standardization organizations around industry consortia such as the W3C or OASIS have been mainly responsible as the standards development organization for Web services. This can be seen as a reaction to alleviate the limiting factor of standardization often regarded as a heavy vehicle [157]. Due to the dynamics of innovation with non-linear and much shorter life-cycles in information technology such standardization processes driven by industry consortia provide much more agility and ensure market relevance, however to the cost of broad consensus [158]. In the field of the Web service standardization this has been the case as only corresponding major industry players have been involved. However, the standards are provided on a royalty free basis and in the last decade Web services have gained considerable popularity and broad market uptake.

One major success factor in the standardization of Web services and its market adoption can be seen in the political or business dimension regarding the consensus level. In the context of standardization around distributed object-oriented systems the industry uptake has been limited as not all major industry players adopted the developed standards in their product portfolio. No industry-wide consensus could be reached regarding a shared agreement of common object models and corresponding programming language models (cf. Section 3.2). Whereas, Web services built on more light-weight concepts focusing on message-based and document-based interaction, they consequently implied less dependency to existing technology and thus could reach the consensus level for broad market uptake. Thus, a good trade-off between the enabling and limiting factors of standardization could be found and made standardization of interface descriptions and interaction patterns by means of Web services the dominant state-of-art method of enabling technical interoperability in SOA.

This success story has led to the conception that developing industry-wide accepted standards for common information models could be also the key towards semantic interoperability in SOA. However, this has often lead to an oversimplification of the problem not reflecting the business driven fundamental need for flexibility in the specification of service granularity and their information models reflecting dynamic business requirements. Consequently, it can be noticed that a unified adoption of business-oriented semantic standards such as the before mentioned UN/CEFACT or RosettaNet (cf. Section 4.4.1) is limited. Rather, there are various standardization initiatives for so called B2B standards and selecting an appropriate B2B standard is depending on the specific business situation [159]. Some prominent B2B standards besides RosettaNet or UN/EDIFACT further include but are not limited to ebXML [160], SPEC2000 [161], CIDX [162], cXML [163], xCBL [164], etc. Moreover, the concrete standards are additionally customized and thus deviate from the original standardized specification to meet the needs of the concrete business processes context.

Accordingly, it can be stated that regarding semantic standards the required positive trade-off between the advantages and disadvantages of standardization could not be found. The successful shift in the context of technical standards from slower government standardization bodies to more agile industry-based consortia could apparently not compensate and meet the high requirements for business adaptability taking also into account business diversity. Moreover, a light-weight consensus limiting the dependencies as reached in terms of Web services on the technical level could not be established in terms of standardization on the semantic level, which can be explained with the fact that information models are heavily depending on the application context which may differ from business to business. Thus, any standardization measures lead to the consequence that internal information models are more or less tightly-coupled to a fixed conceptualization and therefore imply limited adoption of such standards.

4.4.3 Semantic Standardization and Monolithic Information Models

On the technical level Web service standards have enabled modular architectures, as they allow to decouple functional components and service users from service providers in context of Web service development, Web service operation and Web service management. In contrast, applying standards on the semantic level leads to opposite effects regarding architectural principles. In fact, it can be argued that they establish in analogy to common architecture patterns so called monolithic structures even not on the functional level but on the semantic level.

Standards for information models can be built in modular units. Accordingly, the concept of building blocks is well adopted in prominent B2B standards. As mentioned above, e.g. the ebXML Core Components approach [154] catalogues business-related information components that are of horizontal nature, which then can be reused in different business-contexts to build more specific information components. However, the more specific information components need to be derived from the core components. This enables semantic interoperability for the basic parts with overall relevance but has the consequence that business-context specific information components are tightly-coupled to the general core components as they cannot exist independently. Therefore, this tight-coupling establishes a monolithic architecture on the semantic level because any part relevant for information interchange is realized according to one single conceptualization.

Such a monolithic architecture on the semantic level caused by rigorous semantic standardization brings analog to the field of software architecture the problem of poor maintainability. This disadvantage is serious because with growth of coverage of the semantic standard maintainability efforts rise disproportional. The various stakeholders in the standardization process which represent the diversity of business requirements need to coordinate each other in order to achieve consensus. But due to the monolithic structure including tight dependencies no efficient and effective separation of concerns can be established. Thus, the coordination of N actors and business requirements results in N^2 complexity [165] as they need to be mutually synchronized.

Consequently, based on the analysis above it can be concluded that a rigorous standardization of information models attempting to reach maximal coverage does not provide an optimal solution to the problem of semantic interoperability especially in large-scale SOA landscapes involving multiple organizations. However, in order to overcome a similar situation on the semantic level of necessary point-to-point integration as described in Figure 4-4, the approach of standardization cannot be completely dismissed. Rather, an appropriate degree and coverage of semantic standardization with reflection of the consensus dimension needs to be found. The following subsection continues in particular with these aspects.

4.4.4 Consensus Degree and Adequate Scope of Semantic Standards

As already noticed in the context of Web services and in general in standardization of information technology the trade-off between the limiting factors of standardization have led to a shift from government-driven standardization organizations to industry consortia. This can be seen as a reaction to the wide believe that especially in the Internet era, the consensual standardization process is too slow to produce anything relevant for business especially with regard to maintenance of standards and adoption of new requirements [158]. Typically, government-driven standards organizations have well defined and agreed procedures for development and approval of standards, dispute resolution and fair representation for affected parties. In contrast, industry consortia often appear for specific purposes on a temporary basis perhaps set in motion by one or more companies with common interest. Such standardization initiatives may have less complex procedures with the advantage of higher speed. However, they cannot insure participation from all entities with a material interest [166]. Due to the high degree of innovation with requirements such as rapid change and flexibility of standardization procedures the consensus degree is limited and leads to a smaller scope of standardization.

Another type of standardization that addresses this trade-off are public specifications. They are published by a standards organization but are controlled by the sponsor or a group of sponsors, which respectively does not reflect the consensus of all possible stakeholders but allows prompt publication and usage. And finally there are so called in-house standards, which are usually non-public and just used within an enterprise or an enterprise network. The following Figure 4-5 illustrates the different types of standardization along the trade-off between consensus-degree and the adequate scope of standardization.

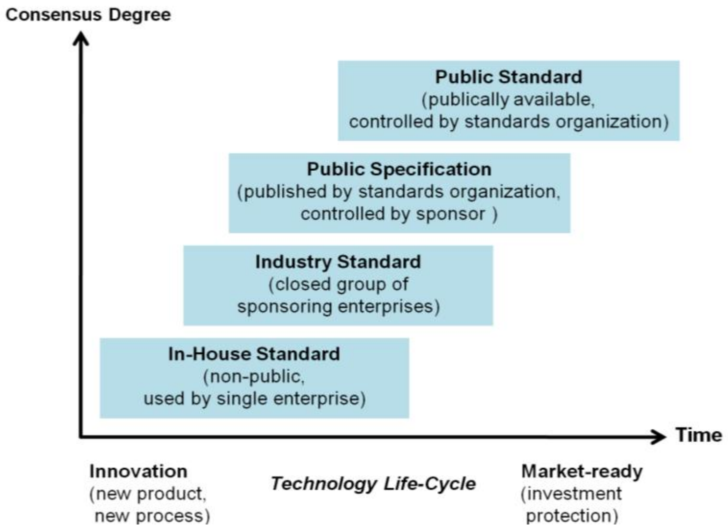


Figure 4-5 Consensus Degree and Appropriate Scope of Standards [167]

Figure 4-5 demonstrates that products or processes with a high innovation rate requiring for frequent change and flexibility in standardization processes are related to a lower consensus degree of stakeholders and thus result in smaller groups and therefore smaller scope of standardization.

Reflecting this in the context of standardization of information models it can stated that the subject of standardization undergoes rapid innovation. Defining final standards for business semantics, a standard that is suitable for any current and future business processes, is an ideal that cannot be reached [168]. Business processes and scenarios are highly dynamic and object of continuous adaption and advancements. Furthermore, there are conflicting goals of stakeholders. On the one hand, business processes targeted by semantic standardization are internal or within closely related organizational units and their requirements need to be reflected. On the other hand, they are in competition with the requirements for cross-organizational processes and both need to be aligned within each organization participating in the standardization process.

Consequently, the following conclusion regarding the limitations of standardization on the semantic level can be made: Even the conceptual question where to position semantic standardization within the discussed Figure 4-5 cannot be finally answered, it can be clearly derived that the scope is not on the level of universal semantic standards. Information models, which are subject to frequent business changes, can be classified as highly innovative with a lower consensus degree and thus their standardization should be located further in the direction of the lower left-hand side corner of Figure 4-5.

This analysis explains why attempts for universal semantic business standards such as the aforementioned UN/CEFACT or RosettaNet (cf. Section 4.4.2) have limited market success. Consequently, the more adequate scope of semantic standardization can be seen on covering smaller groups of interest allowing also multiple coexisting standards for information models covering in principle similar issues.

This section has provided an argumentation of the inadequacy of common standards for information models in highly distributed and dynamic environments targeted in cross-organizational SOA landscapes and the need for multiple coexisting information models in this context. The main argumentation line therein has been the need for continuous adaption and advancements of information models which leads to limited consensus degree. The following section aims at further strengthening the argumentation for multiple coexisting standards of information models. However, it does not focus on the dynamics of business requirements that causes the limited consensus degree but focuses on the nature of information models and its inherent context dependency.

4.5 Context Dependency of Information Models

In general, a model can be used to describe, explain or predict certain aspects of the reality. Models are abstract artifacts in terms of an abstraction on parts of the real world driven by a specific purpose. Information models, which are the central artifact when it comes to semantic interoperability, are specific models. Information itself is an abstract artifact and information models add another abstraction layer to model relevant information in a certain context or domain.

4.5.1 Heterogeneity of Information Models

As already stated in the analysis above (cf. Section 4.4), information models in the business context tend to be diverse and changing over time due to the frequent changes in business processes and different context-specific business requirements in general. In particular, information models therefore are characterized by the following differences such as [124]:

- Different terminology for information entities
- Different modeling conventions
- Different (possibly overlapping) coverage
- Different granularity

Whereas the first two points may constitute differences that could be overcome in terms of adequate organizational measures; however, the last two points represent more fundamental differences. The different coverage of information models which are possibly overlapping or the different granularity or deepness of the modeled information entities can be linked to the different purposes and contexts, which information models need to reflect.

In order to further understand the nature of the differences in information models and their relation to multiple and heterogeneous representations, the following section applies a general model theory to information models. A deductive approach is followed to derive specific characteristics of information models from a general model theoretic perspective. Firstly, the central concepts of a selected model theory are outlined and then related to aspects of information models in terms of analogies. Finally, conclusions are derived in order to provide further argumentation for the requirement of loosely coupled information models in large-scale SOA landscapes.

4.5.2 Model of Conception and Information Models

The model of conception [169] provides a model theory which puts special emphasis on the context dependency of models and its conception by an inherently subjective actor. The main ideas of the model of conception point out that the constituting parts of a model are entities of the real world conceived as a model by a subjective actor. That means that such entities only exist as a model, in so far as they are being conceived as a model by a subject. Furthermore, entities are related to other entities. The relation between one entity and the others is also part of the conception of a subject and constitutes the context of the entity [170]. The following figure illustrates the basic concepts of the model of conception:

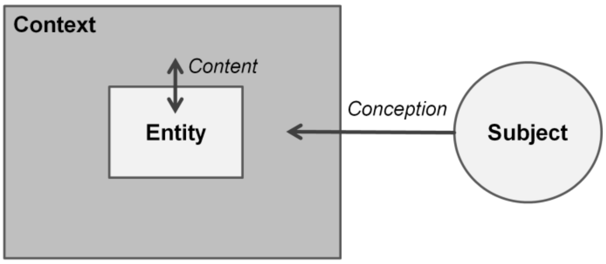


Figure 4-6 Model of Conception [170]

The subject may be a human being, but also a group of people or any other actor which may be expected to make a judgment. Furthermore, the content of the conception is determined by the contextual relations of the entity [169].

This general model of conception can be applied to information models. With regard to information models, the basic concepts of the model of conception such as *subject*, *entity* and *context* can be interpreted as illustrated in the following figure:

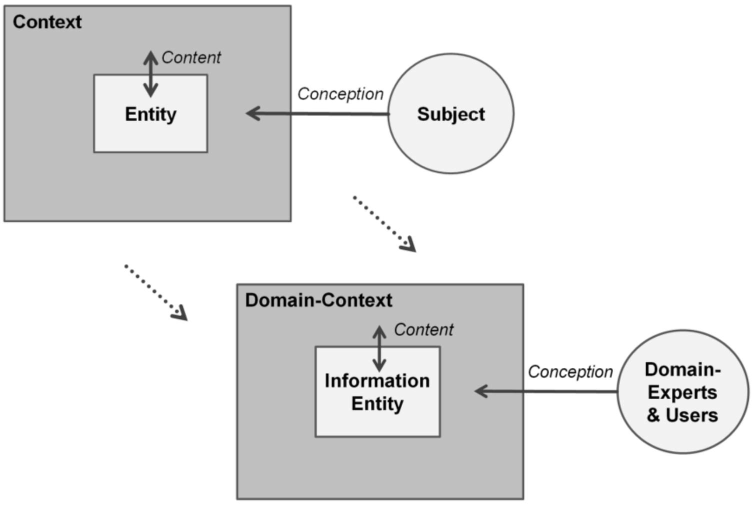


Figure 4-7 Model of Conception Applied to Information Models

The subjective actor can be interpreted as a group of domain experts, who develop a specific information model of a certain as relevant conceived domain. Moreover, the subjective actor can be an application developer, who reuses the developed information model for a certain IT application or a user in terms of a human being which reads and writes information within this IT application. However, the user perspective is discussed later on in the next Section 4.5.3. Furthermore, the entity can be viewed as an information entity e.g. in terms of a customer address expressed in a schema representation or UML diagram on a sheet of paper. However, the actual information model of the customer address only exists in the mind of the domain-experts and multiple views on the in principle same information entity are possible. This can be linked with the observed differences in information models as outlined above. Differences in terminology for information entities and different modeling conventions can be the result, as different domain-experts or groups of them have different conceptions. Moreover, the as relevant conceived domain differs between different actors and consequently results in different possibly overlapping coverage of information models. Finally, the different granularity of information models can be related to the model of conception. The extent to which a larger information entity is subdivided into further information entities depends on what is conceived as relevant to be expressed on the minimal granularity level, which consequently can differ again between different domain experts.

However, so far mainly domain-experts as creators of information models have been discussed and information models have been described as a conception of a domain. But information models are not just created; they are also used or applied to a certain situation and therefore have a specific purpose.

4.5.3 Constructive Model Relations and Information Models

This notion or differentiation of a model as being a model of something and at the same time being a model for something is discussed in the context of the model of conception and has led to the identification of so called constructive model relations [170] as illustrated in the following Figure 4-8. Again, the idea is to briefly outline the general model theoretic concepts and then interpret them in the context of information models.

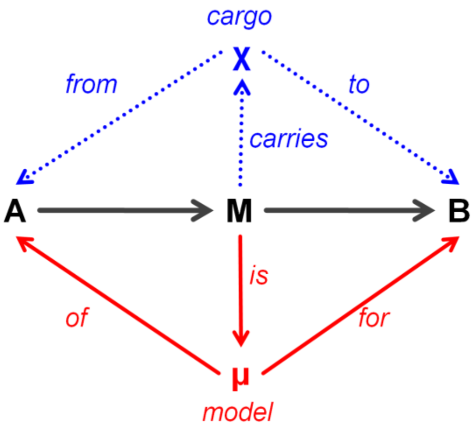


Figure 4-8 Constructive Model Relations

As already stated in context of the model of conception, it is distinguished between the model object or entity M and the purely mental model μ. Furthermore, in [170] it is distinguished between the creation of a model and the application of it. These two actions constitute constructive model relationships as they are established by a constructive act, which is either thought or actually performed. Thus, on the one hand, the model creation is a constructive model relationship starting from an initial object A which may be something observed or a set of requirements and resulting in the actual model object M via some kind of abstraction or selection. On the other hand, as well the model application is a constructive model relationship, starting from the created model object M which may be a prototype or a reference object and realizing the application in terms of an object B via a kind of production, role assignment, or mapping.

Consequently, it can be stated that a model is related not only to something (A) of which it is a model but also to something (B) for which it is a model [170].

Moreover, in order to be useful for the application to a specific situation or problem, the model object must ensure to exhibit certain qualities which it transfers from something observed or a set of requirements A to the application B and which is problem relevant. What is transferred in this way via a model object is called the cargo of the model [170] as illustrated with the blue arrows in Figure 4-9. The object M, which is conceived of as a model, cannot get its role as a model and its function as the carrier of a cargo for a resulting object B unplanned or in an accidental manner. The model object M has to be well chosen, produced, or evaluated as a model in the conception of the judging subject, in such a way that it actually carries the described cargo to serve as a model. Accordingly, the practice of modeling has to aim at ‘working the cargo into the model object’ during the model creating process in a way that the cargo may also be transferred to the resulting object B in the course of model application [170].

Analog to the model of conception, which has been applied to information models, also the model theoretic concepts of constructive model relations can be applied to the context of information models. The following Figure 4-9 shows this conceptual instantiation. It further integrates the distinction between different abstraction levels of information models (cf. Section 2.3.3) from the conceptual idea in the mind to the physical model as a representation of the actual information.

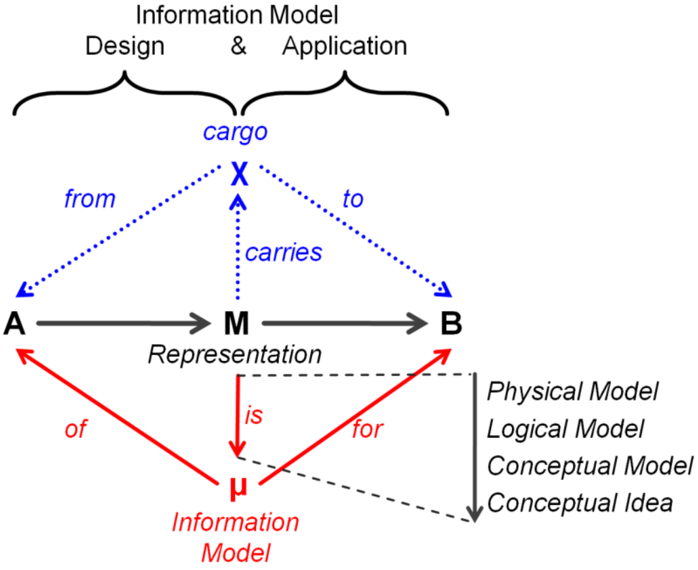


Figure 4-9 Constructive Model Relations and Information Models

The two constructive model relationships constitute the information model design phase one the hand and the information model application phase on the other hand. The analogy that a model M is a model for something B explains why an information model is always for some specific application or application domain according to a subjective conception.

Furthermore, the differentiation between the actual information model μ in the mind and its representation through the model object M can explain, that in a semantic integration scenario of two applications B and B’ even two different mental information models may match, but the corresponding representations e.g. in terms of different message schemas M and M’ may not enable seamless information exchange. This can be also linked back to the distinction between different abstraction levels of information models (cf. Section 2.3.3) from the conceptual idea in the mind, over the conceptual model, to the logical model and to the physical model as a representation of the actual information. With each lower abstraction level further technical and application specific conditions and constraints need to be reflected, which leads to the heterogeneous occurrences of information models on the concrete representation level.

Coming back to the constructive model relationships, this explanation can be further substantiated by the concept of a cargo which is transferred through the model object or here the information representation format. In the context of information models that means, that an information model representation M might carry the relevant characteristics X of required information for the purpose of application B, but when it should be applied to the purpose or application of B´ this representation and corresponding cargo X might not be sufficient. In this scenario the concrete information representation M carries X to be adequate for application B but it does not carry X´ required for application B´.

4.5.4 Conclusions and Implications for Information Models

In summary, the application of the model of conception and the transfer of the constructive model relationships to the realm of information models can be concluded as follows:

- The general argumentation that context dependency of models is an inherent characteristic of models and a determinant factor to cope with complexity [171] is particularly valid for information models as any statement taken out of context cannot be understood. Only by putting it into a context provides it with meaning.
- Furthermore, the model theoretic perspective shows that context cannot be regarded as an objective issue; rather it is determined by the subjective conception of an actor. Applied to the realm of information models, this leads to the understanding that information models of a certain domain are naturally not unique and uniform but different depending on the different actors developing and using them.
- The last point is further stressed by the central statement, that a model is related not only to something (A) of which it is a model but also to something (B) for which it is a model [170]. Transferred to information models, this shows that information models are not only dependent on the actual business domain on which they provide an abstraction, but moreover this abstraction cannot be undirected but must be related to a specific purpose or respectively to concrete IT applications. Consequently, due to their very nature, information models cannot be general, unique and uniform but rather they are diverse, heterogeneous and consequently are likely to occur as multiple possibly conceptually overlapping variations.

This section has analyzed the context dependency of information models. It has argued that in a multi-stakeholder environment such as given in the targeted large-scale SOA context, the subjective conception provides the determent factor of information models. Therefore, from a theoretical initial point, no general objective or in mathematical terms no canonical form of an information model of a business domain exists per se but rather multiple different coexist, according to the different conceptions of the involved stakeholders. This understanding identifies the challenge of integrating the different conceptions to achieve semantic interoperability.

As already outlined in Section 4.4, there are generally two strategies for the integration of different conceptions or for bridging the semantic interoperability gap. On the one hand, semantic integration can be targeted by aiming for a common and shared information model. This requires the alignment of the different subjective conceptions into a uniform one which suits all involved stakeholders. On the other hand, the divergent conceptions and resulting information models can be accepted in their heterogeneity by keeping their independence and integrating them in a flexible manner in terms of mediation between them. Such an approach can be regarded as following the basic principle of loose coupling, which is known as an effective instrument to cope with complexity in distributed systems. The following section presents this approach that applies the concept of loose coupling to the semantic level. The subsequent section then provides further analysis with regard to the research hypothesis of this work and provides a comparison regarding effectiveness and efficiency between the here presented approach of semantic mediation between loosely coupled information models and the traditional alternative approach of semantic standardization.

4.6 Loose Coupling on the Semantic Level

The principle of loose coupling can be regarded as one of the fundamental concepts in the evolution of computer system engineering. Notably, loose coupling is not a feature that is necessary for a system to operate but rather it is essential to manage the complexity in a system under change. However, it was originally introduced into organization science by Karl Weick. In [172] loose coupling is described as a situation in which a series of stable subassemblies are responsive but retain evidence of separateness, independency and identity. Furthermore, loose coupling is evident when elements affect each other suddenly rather than constantly, negligibly rather than significantly, indirectly rather than directly and eventually rather than immediately. In organizational science the notion of loose coupling suggests that some classical principles of administration like centralized control and rational planning may not be useful as often believed. It emphasizes the limitations to administrator’s abilities to shape the instructional process.

Taking into account the organizational aspects and challenges identified in the context of semantic harmonization as analyzed in the previous two sections about limitations of standardization for semantic interoperability in SOA (cf. Section 4.4) and about context dependency of information models (cf. Section 4.5), the notion of loose coupling promises to be an adequate mechanism to manage the complexity with regard to change of business semantics.

Especially in distributed systems and particularly in the field of SOA, the principle of loose coupling represents a core mechanism to cope with complexity. The following figure illustrates the basic idea of transferring the principle of loose coupling to the semantic level, namely from loosely coupled services to loosely coupled information models:

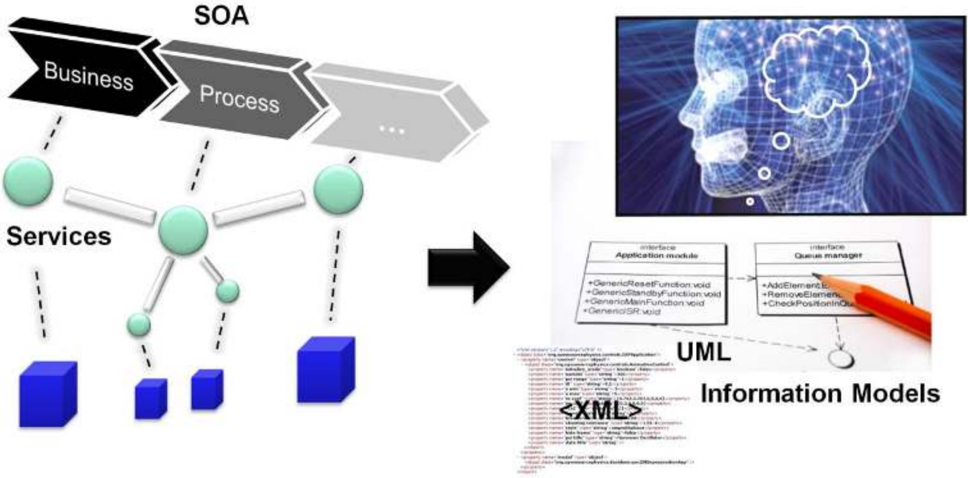


Figure 4-10 Transfer of Loose Coupling to the Semantic Level

In SOA, loose coupling is applied to tackle the complexity of business IT alignment. Services are loosely coupled to chains in order to realize business processes. Moreover, service users and service providers are loosely coupled based on the abstract concept of a service. And finally, services are decoupled from the actual IT components that provide the services to provide further flexibility. In this sense loose coupling is mainly focused on control and maintenance issues ensuring independent management of services under different ownership.

However, when looking at information models used for message exchange between services a rather tight degree of coupling becomes evident. For the sake of semantic interoperability service users and providers usually are committed to a shared information model in order to ensure that the messages exchanged can be understood and processed as indented. Aggregation, association or inheritance relationships between information entities are dependent on an overall conceptualization. Practically this means that information model representations of service providers are imposed on service requesters, which can be regarded as tight coupling on the semantic level. However, in distributed environments with limited centralized (i.e. top-down) management authority the commitment of independent stakeholders to a shared information model leads to the challenges discussed before.

In order to transfer the principle of loose coupling to the semantic level this section analysis the principle of loose coupling in computer science in general and more detailed with a focus on SOA and identifies the core characteristics that constitute this kind of architecture style. Then it is discussed how and to which extent loose coupling can be applied to information models and which conclusions can be derived from this.

4.6.1 The Principle of Loose Coupling in Computer Science

Firstly, selected definitions of loose coupling from the perspective of computer science are discussed. Then the main motivations for loose coupling are outlined before the central characteristics are extracted, in order to provide a basis for the transfer of the concept to information models.

Definitions

Although the term loose coupling is widely used in discussions of computing and software architectures the term has a relatively diffuse definition. However, there seems to be a general convergence towards the following shared understanding [173]:

- In computer science, coupling or dependency is the degree to which each program module relies on each one of the other modules.
- Loosely coupled systems are considered useful when either the source or the destination module is subject to frequent changes.

A more analytical definition is provided in [174], where the degree of loose coupling is defined along the three dimensions: Knowledge, Availability and Trust:

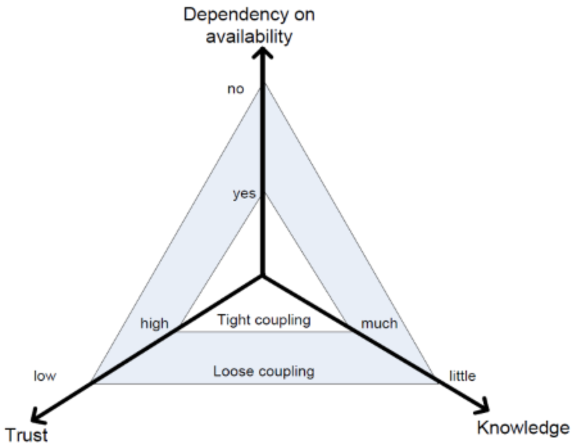


Figure 4-11 Dimensions of Coupling [175]

A module A is related or has a dependency to component or module B. Then module A is considered loosely coupled with module B if [174]:

(1) Knowledge - A has minimal knowledge of B.

(2) Dependency on availability - A can work even when B (or the communication link to B) is temporarily not available.

(3) Trust - B does not need to trust that A satisfies pre-conditions and A does not need to trust that B satisfies post-conditions.

The extent or degree to which the modules in a system are coupled is regarded as a relative or qualitative notion. Furthermore, these dimensions can be related to the functional distance between modules, which can be used to define the adequate degree of coupling in a certain situation. The following Figure 4-12 illustrates the relation between the degree of coupling and the functional distance between different modules:

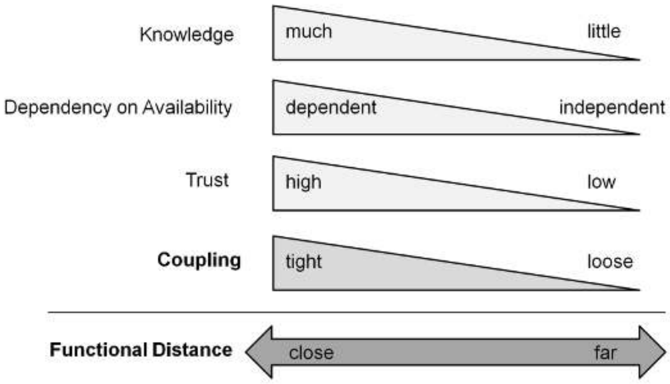


Figure 4-12 Degree of Coupling and Functional Distance [176]

The notion of functional distance can be also found in the concept of cohesion between or within modules or components. In the context of structured analysis and more particularly in context of structured design [177], cohesion is strongly related to loose coupling as usually coupling is contrasted with cohesion. In structured design the concept of cohesion is defined as the degree to which the internal contents of a module are related. And coupling is defined as the degree to which a module depends upon other modules. Furthermore, cohesion is concerned with the grouping of functionality into a set or a particular module. Thereby, functional closely related elements should be grouped into the same module whereas the boundary to other modules should be clearly defined with a precise scope. In conclusion modules should be designed according to strong cohesion regarding internal contents and loosely coupled with regard to dependencies between different modules.

Motivation

As indicated by the definitions above, loose coupling is an attribute of IT systems with the aim of reducing the interdependencies across modules or components. In particular, loose coupling is motivated by reducing the risk that changes within one module will create unanticipated changes within other modules which requires change in their implementation. This approach specifically aims at increased flexibility while deploying, modifying, or replacing modules without necessarily affecting other modules that communicate or share information with it.

Furthermore, loose coupling eases testing and maintenance because problems are easy to isolate and unlikely to propagate. Moreover, the isolation and clear distinction between modules simplifies to understand the semantic of modules. This fosters reuse as modules can be handled separately and related or dependent modules do not need to be integrated in parallel.

Particularly, in SOA loose coupling provides the foundation that enables IT supported business processes across technological heterogeneous IT landscapes. Loosely coupled services provided on different platforms, which may be based on incompatible system technologies, can be joined together in order to create composite services and thus realize business processes.

However, loose coupling is not universally positive. It is necessary to balance the trade-off between utility and costs. On the one hand, loose coupling promotes more flexibility but on the other hand these benefits cause drawbacks. For example if systems are de-coupled in time using e.g. a message-oriented middleware as usual in SOA, it is difficult to also provide transactional integrity [173]. Moreover, loose coupling increases costs in terms of additional efforts for the integration. At least this applies initially due to the distributed approach and finally regarding performance as additional communication efforts are required. To conclude, it can be stated that it remains the art of the architect to find the adequate degree of loose coupling in a concrete integration scenario [174].

4.6.2 Transferrable Characteristics of Loose Coupling

In the previous section the principle of loose coupling in computer science has been discussed. The main focus has been put on decoupling IT functionality in order to cope with the complexity of large and distributed IT systems. With regard to the goal of transferring the concept of loose coupling from a well-known functional dimension to the semantic dimension this section pinpoints the central aspects of loose coupling which need to be covered as well when discussed in the context of loose coupling of information models. Based on the analysis above, the following three central characteristics of loose coupling can be distinguished and should be utilized as reference points for the transfer of the concept:

- Autonomy – Loosely coupled systems are based on decomposition into multiple components or modules that are as autonomous as possible. The modules are self-contained and discrete. The aimed independence between the modules thereby focuses mainly on management, i.e. leaving the control of module's internal aspects to the respective module owner.
- Flexibly binding – Loosely coupled modules are related to each other by means of flexible binding mechanisms that allow to integrate the modules. In contrast to tightly

coupled systems, where different modules or components are statically bound, loosely coupled systems exhibit dynamic or late binding, in order to increase independence and the above described autonomy between different modules.

- Encapsulation – Loosely coupled modules are clearly defined and exhibit a precise scope. Hence, loose coupling follows the principles of separation of concerns and information hiding, i.e. keeping design and maintenance decisions at the module's owner. Accordingly, module interaction is based on stable interfaces which ensure that functionality can be used without being concerned about the other module's internal implementation. Thus, the implemented application logic behind the interface can be changed without affecting other modules that communicate or share data with it. In this sense interfaces can be regarded as intermediary points between interacting loosely coupled modules.

Taking furthermore into account the aim of developing a methodology on how to apply loosely coupled information models in concrete SOA scenarios (cf. Chapter 5) here also the practical perspective on how to apply the general concept of loose coupling should be given. Hence, the general method of applying loose coupling focusing on the functional dimension can be summarized as follows:

1) Decompose a single unit into multiple independent components or modules (autonomy).
2) Place an intermediary point by means of an interface between two interacting end points (encapsulation).
3) Define concrete relations or interactions between different modules as late as possible in terms of dynamic mechanisms (flexible binding).

Consequently, the following section continues with discussing how the concept of loose coupling can be applied to information models.

4.6.3 Loosely Coupled Information Models

The transfer of the concept of loose coupling to the semantic level implies that the core principles of loose coupling can be applied to information models. Therefore, this section targets the question how the above identified central characteristics of loose coupling, namely autonomy, flexible binding and encapsulation can be applied to the semantic level. In the following a specification covering these three characteristics is provided.

Autonomy of Loosely Coupled Information Models

With regard to autonomy, it can be stated that loose coupling on the semantic level can be identified when the potential information space is decomposed into multiple coexisting, self-contained and independent information models. Analogical to loosely coupled functional components the feature of independence is mainly related to the management of information models which is under the control of different ownership. I.e. issues related to the life-cycle, maintenance or evolution of information models are kept under the authority of the respective information model owner. In particular, this implies that no cross-references exist between loosely coupled information models such as generalization, specialization or aggregation of entities of different origin. That would cause that the existence of one information model is dependent on another which consequently contradicts the aim for autonomy.

The approach of multiple independent information models takes into account the realistic point of view that the existence of an overall generalized so called canonical information model cannot be assumed in heterogeneous scenarios including autonomous actors and organizations. Organizational boundaries in community processes for the development and maintenance of information models and the complexity deriving from inherent domain-specific differences in requirements force a coexistence of independently managed information models.

Consequently, it can be stated that autonomy can be reflected on the semantic level and moreover forms an integral characteristic of loosely coupled information models.

Flexibly Binding of Loosely Coupled Information Models

Certainly, mutually independent and autonomous information models need to be interlinked and related to each other in order to achieve semantic interoperability in a concrete integration scenario. However, such integration similar to loosely coupled functional components does not need to be permanent or static. But rather information models can be interlinked to each other in a dynamic manner by means of mediation between them. Hence, semantic interoperability can be achieved as differences in e.g. information representation or heterogeneous granularity levels in cross-organizational scenarios are aligned in the moment when information needs to be exchanged.

In order to achieve a high degree of flexibility the mediation mechanism has to cover the whole spectrum of potential differences in information modeling. Furthermore, the mediation should be kept as transparent as possible to the user by hiding the complexity and providing lightweight maintainability. The original information models can remain unchanged and independent of each other and thus ensure the autonomy between the different information models.

Accordingly, it can be concluded that the aspect of flexible binding can be transferred to the semantic level and that a mediation mechanism is the constituting part that enables the flexible binding between loosely coupled information models.

Encapsulation of Loosely Coupled Information Models

Analogically to loosely coupled modules in a functional architecture loosely coupled information models as well should be clearly defined and exhibit a precise scope. The principles of separation of concerns and information hiding can be reflected by loosely coupled information models taking into account the different abstraction levels of information representation (cf. Section 2.3.2). Consequently, loosely coupled information models should be encapsulated on the most abstract (the conceptual abstraction level) to hide the concrete internal representation on lower abstraction levels such as the logical or the physical abstraction level. Accordingly, the conceptual abstraction level acts similar to an interface encapsulating the internals of information models.

Decoupling in terms of mapping between the conceptual representation and the internal realization ensures that internal changes do not affect the external representation of the information model. Thus, the mediation mechanism can be based on the conceptual abstraction level and internal changes of the information model do not affect other information models and consequently independence is fostered. In this sense the conceptual abstraction together with the mediation mechanism acts as the intermediary between interacting loosely coupled information models.

To conclude it can be stated that as well encapsulation can be reflected on the semantic level. The main idea is that the conceptual abstraction level of information models provides an encapsulation of information model internals and thus enables their loose coupling.

Definition of Loosely Coupled Information Models

Based on the above discussion it can be concluded that the major characteristics of loose coupling can be transferred to the semantic level. Reflecting the above derived analogies the following definition of loosely coupled information models can be given:

Loosely coupled information models:

a) are mutually independent and self-contained information models that exist and are managed autonomously.

b) exhibit flexible binding as they can be dynamically interlinked to each other to ensure semantic interoperability in a concrete integration situation based on a mediation mechanism.

c) are encapsulated by an external representation on the conceptual abstraction level that hides the internal representation on the logical or physical abstraction level to prevent unwanted change propagation of internals.

The following Figure 4-13 illustrates the definition and highlights the main conceptual parts of loosely coupled information models:

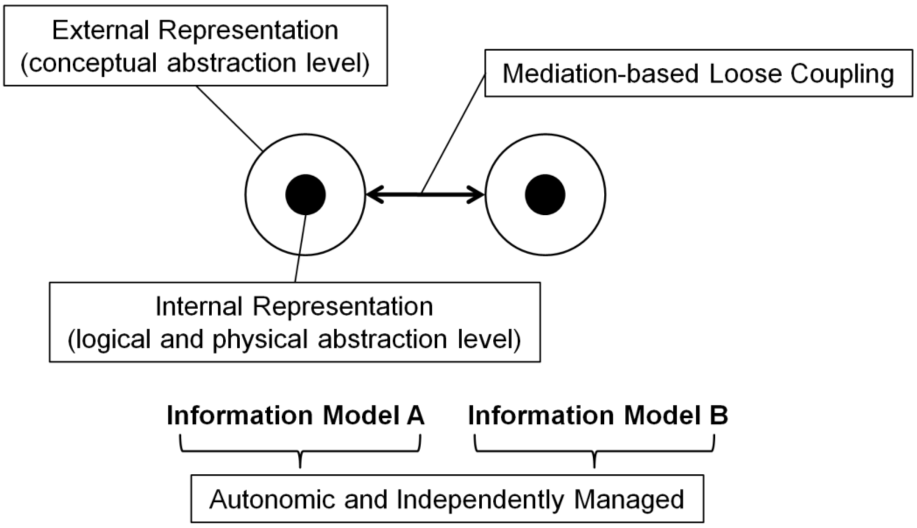


Figure 4-13 Definition of Loosely Coupled Information Models

4.6.4 Limitations in the Transfer of Loose Coupling and Open Issues

As examined above the concept of loose coupling as used in functional architectures of IT systems can be transferred to the semantic level in terms of loosely coupled information models. However, besides the analogies regarding the major aspects of loose coupling as well limitations of the transferability have to be noticed.

Usually, in loosely coupled functional architectures the functionality of the IT system is distributed among the independent functional components or modules in a disjunctive manner. Each module is defined by a clear scope in order to avoid overlapping responsibilities for certain functionality. However, regarding loosely coupled information models such a distinct distribution of scope is not reflected. In contrast, loosely coupled information models tend to be conceptually overlapping on purpose in order to allow the modeling of the same domain or parts of related domains from different perspectives (cf. Section 4.5). This coexistence ensures that information models can be independently designed according to the information model owner specific needs. Then, the overlapping parts of the loosely coupled information models need to be addressed in a concrete semantic integration situation, which should be considered by an appropriate mediation mechanism.

Furthermore, it has to be noticed that in the above discussion and definition of loosely coupled information models the mediation mechanism enabling the loose coupling plays a central role, although it has not yet been particularly specified. In order to specify the mediation mechanism its core characteristics need to be defined. Taking into account the research hypothesis of this work (cf. Section 1.3) the main targeted characteristics of the semantic mediation mechanism are effectiveness and efficiency. Consequently, the next section analyzes the role of effectiveness and efficiency and the relation between them, in order to derive a more concrete idea on how the semantic mediation mechanism should be designed.

4.7 Trade-off between Effectiveness and Efficiency

Effectiveness and efficiency in achieving semantic interoperability in large-scale SOA landscapes are two fundamental factors that need to be distinguished. On the one hand, effectiveness plays a central role in the way semantic interoperability is achieved. Each proposed solution is subject to certain explicit or implicit assumptions such as a high or low consensus degree enabling or limiting standardized canonical information models across organizational boundaries. Taking into account the fact that achieving semantic interoperability not only involves technical aspects but as well exhibits a challenging organizational dimension, the question regarding feasibility of a proposed solution in practice needs to be addressed. On the other hand, the efficiency of the mechanism for achieving semantic interoperability strongly influences the degree to which semantic interoperability is achieved. The question is how the complexity of semantic heterogeneity is tackled and how much manual integration efforts need to be invested in order to bridge divergent information representations.

Often effectiveness and efficiency are competing goals. However, a sufficient solution for a mediation mechanism resolving the semantic interoperability problem needs to provide both qualities. In the following, three different mechanisms for semantic mediation are presented. They are examined with regard to the question on how they address effectiveness and efficiency and how they relate to the trade-off.

4.7.1 Point-to-Point Mediation

The first approach is based on a direct application of the concept of independent information models and mutual mediation between them (cf. (2) in Section 4.4.1). Each organization manages their own internal information model, which may be implemented by state-of-the-art technologies such as XML Schema or SQL database schemata to be located on the logical or physical level. This internal information model is encapsulated by a conceptual model, which can be realized in terms of an ontology (cf. Section 3.6). Accordingly, the main semantic integration takes place in the mappings between the different external information models based on ontology mappings. Consequently, a point-to-point mediation between the heterogeneous information models is required. The following Figure 4-14 illustrates this approach:

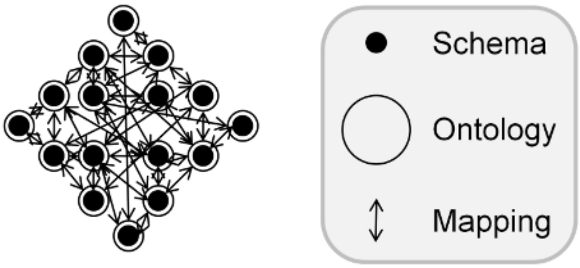


Figure 4-14 Point-to-Point Mediation

The approach bears the advantage of loosely coupled information models as semantic integration is performed on the conceptual level instead of the technical level. In the context of SOA this can be enabled by Semantic Web service technologies (cf. Section 3.4). Addressing the semantic integration problem on a higher abstraction level responds to the high complexity of integration by separating semantic integration from technical issues. Thus, the integration efforts can be reduced.

Furthermore, it can be stated that the presented approach provides an effective solution. Meaningful information exchange between the different systems and applications using the heterogeneous information models can be ensured in terms of mappings between the ontologies and thus the goal of semantic interoperability can be achieved.

However, point-to-point mediation features an essential drawback regarding efficiency. Although a single semantic integration between two interacting organizations can be addressed with reasonable efforts as described above the overall semantic integration costs for the whole ecosystem are much higher. The point-to-point mediation implies potentially $N \, x \, N$ mappings to ensure semantic interoperability between all organizations. This means that the approach of point-to-point mediation has a quadratic complexity. Therefore, the approach does not scale as integration cost are increasing disproportionally when further organizations need to be integrated into the ecosystem. However, the flexible integration of new business partners into cross-organizational business processes constitutes a key promise of SOA.

The point-to-point approach represents an extreme variant with regard to the granularity of mediated partners. The following approach presents the opposite variant in order to set the range of possible options in this trade-off between effectiveness and efficiency.

4.7.2 Pivot Ontology based Standardization

The second approach follows the idea of a standardized canonical information model (cf. (1) in Section 4.4.1). One global information model is shared. This can be specified on the conceptual level e.g. in terms of an ontology, in order to be independent of the different heterogeneous internal information models of the involved organizations. Consequently, the main semantic integration takes place between the internal e.g. schema-based information models and the global canonical information models in terms of a so called pivot ontology. As the mediation needs to cover both directions: from the internal lower abstract information model to the external higher level and vice versa, the mapping can be also referred to as lifting and lowering. Thus, a message sent from organization A is firstly lifted to the shared pivot ontology and then lowered to the appropriate internal information model used in organization B. This approach represents the dominant semantic integration approach in conjunction with ontologies and can be found in the majority of research projects applying Semantic Web technologies for information integration (cf. Section 3.6 and Section 5.9.3). The following Figure 4-15 illustrates the semantic standardization approach based on a pivot ontology.

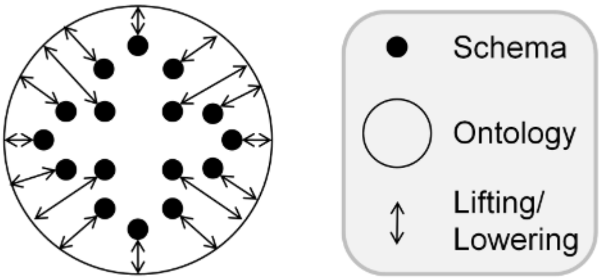


Figure 4-15 Pivot Ontology based Standardization

In contrast to the previous approach semantic standardization based on a pivot ontology just requires N mappings (or liftings/lowerings). Each organization's internal information model has to be mapped only once bidirectional to the pivot ontology to ensure semantic interoperability during information exchange. Thus, from the perspective of required mappings it can be stated that the approach offers an efficient solution.

However, with regard to effectiveness the already discussed limited feasibility of standardized information models in cross-organizational SOA landscapes has to be taken into account (cf. Section 4.4). As new services are added to the architecture, extensions to the common message model are needed to meet their specific data requirements. As more services are added the model grows, complexity rises and it becomes difficult to understand. Moreover, conflicting requirements lead to inconsistencies and required compensation mechanisms. From an analytical point of view consensus for such a canonical information model in terms of a pivot ontology, requires N actors to align their different requirements for information exchange with external partners. The mutual coordination of N actors potentially implies $N \, x \, N$ coordination tasks, which also lead to quadratic complexity. Due to the disproportionate coordination complexity some actors may leave the consensus finding process. Thus, the goal of achieving semantic interoperability within the ecosystem covering all actors is missed. This drawback is also described in [178], which states that it is increasingly unlikely that a single ontology will both adequately capture the domain in question and also be consensual among all interested parties. Consequently, this non-scaling coordination complexity hinders an effective solution.

Hence, the approach based on a pivot ontology provides a sufficient solution with regard to efficiency, however it has shortcomings regarding effectiveness. Basically, it provides a shift from integration efforts for e.g. mapping development to organizational coordination efforts required for alignment of the different perspectives of the involved actors.

4.7.3 Semantic Mediation on Domain Level

Having presented the two opposing approaches for achieving semantic interoperability within the trade-off between effectiveness and efficiency, the following approach represents a combination of the two approaches, which tries to exploit the respective advantages and to minimize the discussed drawbacks.

The limiting factor of the first approach has been inefficiency with regard to high integration cost caused by too many required mappings. The second approach has exhibited shortcomings in effectiveness with regard to high organizational complexity as too many actors need to be coordinated. Taken this into account, the approach of semantic mediation on domain level tries to minimize the required mappings as well as the involved actors to be coordinated. The main idea of this twofold reduction lies in the aggregation of distributed information models to a set of multiple domain information models. Consequently, the semantic mapping can be targeted at the domain level. The following figure illustrates this approach:

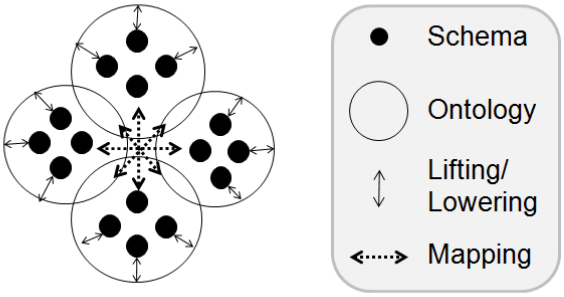


Figure 4-16 Semantic Mediation on Domain Level

The domain models do not necessarily have to be disjoint as in practice domains inherently overlap and community- and application-specific requirements have produced and will always produce different conceptualizations for one and the same problem (cf. Section 4.5). However, in order to achieve a smaller number of actors the scope of a pivot ontology is limited to the domain level. Instead of aiming for one unique shared pivot ontology, the coexistence of multiple domain-specific pivot ontologies is intended. Consequently, the reduction of actors can limit the complexity in the consensus finding process to a feasible extent.

Thereby, the notion of a domain can be understood relatively flexible e.g. in terms of a selected area of activity. The concrete specification of a domain depends on the actual application context. Thus, on the general level required for this approach the concept of a domain is used in context of domain information models that cover a limited scope but however contain or span across multiple individual organizational information models.

Accordingly, semantic interoperability within a domain is achieved by means of a domain ontology. The domain ontology acts as a pivot information model to which the heterogeneous internal information models of the different organizations are mapped in terms of liftings and lowerings. Aggregated over the ecosystem of organizations this results in N mappings analog to the efficient but however not effective approach of overall semantic standardization based on one single pivot ontology.

In order to achieve semantic interoperability not only within each domain but across the ecosystem of organizations a further step is necessary. The required semantic mediation between the domain information models can be enabled by means of ontology mappings as represented by the dashed arrows in Figure 4-16. The semantic integration efforts for these mappings also have quadratic complexity. However, the problem size has been reduced substantially. Considering m domains, where m is much smaller than the N counting the number of organizations, the semantic mediation between the domain information models requires $m \times m$ mappings. The aggregation of total mapping efforts for this approach results in a complexity of $N + m \times m$.

Consequently, the approach of semantic mediation on domain level reduces the complexity as follows:

- from $N \times N$ mappings (point-to-point mediation)
- or from N mappings combined with $N \times N$ coordination efforts (single pivot-ontology based standardization)
- to $N + m \times m$ mappings combined with $\left(\frac{N}{m}\right) \times \left(\frac{N}{m}\right) \times m$ coordination efforts

Thereby m represents the number of domains which is much smaller than the number of organizations or actors N. Furthermore, it has to be taken into account that the coordination efforts for semantic mediation on domain level can be performed in parallel in each domain. To summarize it can be stated that the approach combines the advantages of the before discussed approaches and minimizes their drawbacks. On the one hand, the amount of required semantic mappings is minimized and on the other hand, the number of involved actors to be coordinated is reduced.

4.7.4 Alleviation of Trade-Off between Effectiveness and Efficiency

However, the approach of semantic mediation on domain level just provides a shift within the trade-off between effectiveness and efficiency and cannot resolve it in principle. Nevertheless, the adequate application of Semantic Web technologies can enable an alleviation within the trade-off. The following qualitative analysis illustrates this argumentation:

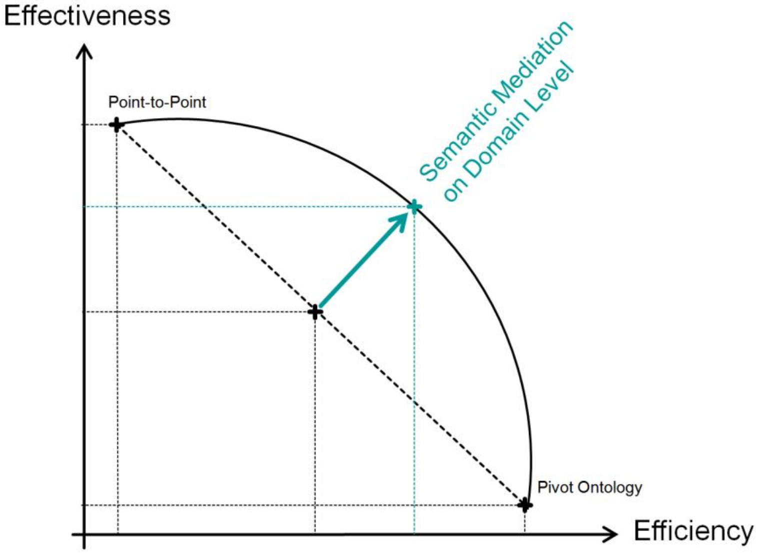


Figure 4-17 Effectiveness and Efficiency Gain

The two opposing approaches of point-to-point mediation and pivot ontology-based standardization lack either sufficient effectiveness or efficiency. Considering the two options in a two dimensional projection they span a line of possible combinations within the trade-off between the two competing goals. Starting from the pivot-ontology based approach the number of actors which need to be coordinated can be reduced. This leads to higher effectiveness as the feasibility of consensus is increased but to the cost of less efficiency as additional efforts for mappings between multiple pivot ontologies are required. Hence, a shift on the line of possible combinations from the lower-right corner up to the center is performed. On the other hand starting from the point-to-point mediation approach the conceptual information models of multiple actors can be united. Thus, the number of point-to-point mappings between the aligned conceptual information models becomes obsolete and thus the total number gets reduced; however to the cost of coordination complexity which is now required for the consensus finding process. Accordingly, this can be represented as a shift on the line of possible combinations from the higher-left corner down to the center.

Besides providing such a combination on the line between the two opposing approaches, semantic mediation on domain level provides an additional alleviation within the trade-off. Exploiting Semantic Web technologies in terms of ontologies for representing the external conceptual information models and ontology mapping to enable the loosely coupled semantic mediation mechanism brings the following advantages:

- Firstly, semantic mediation is targeted on the conceptual level. Consequently, the higher abstraction level reduces mapping efforts substantially as technical issues can be handled with enhanced transparency.
- Furthermore, efficiency is improved as the mediation between different heterogeneous representations in overlapping conceptualizations is only done once on the domain level instead of performing it repeatedly on the technical level in concrete application or process scenarios. Hence, ontology mappings once developed can be reused for various integration scenarios between the involved domains. Additionally, the process expert can focus on process specific concerns and can leave the task of semantic mediation to a domain experts thus taking further advantage of separation of concerns.
- Regarding effectiveness, the formal nature of conceptual models instantiated by the exploitation of ontologies eases the coordination and consensus finding process. Semantics can be discussed on an unambiguous and explicitly defined level irrespective of technical issues. For this purpose ontologies provide an effective instrument to domain experts to specify clear conceptualizations and communicate them in the development process of domain information models.

In this sense the focus on the domain level for addressing the semantic mediation challenge can be understood from two perspectives. Firstly, the domain level refers to the scope covered by the loosely coupled information models. And secondly the domain level further refers to the conceptual nature of a domain model supporting the shift of abstraction level for semantic mediation as described above.

Having discussed the general advantages of Semantic Web technologies for the approach of semantic mediation on domain level the following section discusses how particular features of this technology can be exploited to enable the mediation mechanism between domain ontologies.

4.8 Semantic Bridges for Loose Coupling of Domain Ontologies

The semantic mediation mechanism can be based on the general idea of semantic bridges as introduced in [146]. Semantic bridges provide a framework for ontology mappings as described in Section 3.6.2. They are utilized to describe the relations between concepts defined in different ontologies which are not shared but which intuitionally have an equal or similar meaning. Furthermore, semantic bridges define mappings, i.e. a translation between these concepts to be used in concrete information flows across heterogeneous ontologies. In [127] a matching function is formally defined. The matching function takes two different schemas and provides a mapping between them. A semantic bridge represents the output of such a matching function and thus describes the mapping.

The ontology mappings of semantic bridges can be either manually developed or (semi) automatically by utilizing ontology mapping tools such as presented in [179], [140]. Such mapping tools reuse the approaches developed in the context of schema mapping, on which [127] provides an overview. However, the extent of automation is limited as the identification of similar meaning of separately defined concepts is often context-based and human intervention in the matching process is necessary (cf. Section 3.6.2).

Considering these context dependencies and different granularity levels in representations, defining these mappings to create semantic bridges is a complex task. In particular, semantic bridges should cover general semantic correspondences between domain ontologies rather than covering specific application logics. This means that concepts from different domain ontologies might be regarded as matching in a specific application context; however this relation does not hold in other cases taken into account the overall domain perspective. Therefore, only those concepts should be mapped that semantics generally match, i.e. in all application contexts where those concepts are involved.

After having found matching concepts and properties the matching results i.e. the identified alignments need to be formally defined (cf. Section 3.6.2.). In the context of database schemas these mappings can be formulated e.g. as queries or views (cf. Section 3.5.1). In the context of XML-based approaches these mappings can be implemented in terms of structure-based transformation languages such as XSLT (cf. Section 3.2.2).

However, in the context of ontologies based on formal description logic it is beneficial to express these mappings as well with description logics. In order to serve the given circumstances of heterogeneous and independent domain ontologies the added value resulting from the application of Semantic Web technology lies in exploiting specific features of this technology. Three such features are the following which are described in more detail below:

- Generalization and Polymorphism
- Facet Analysis Classification
- Declarative Rule-based Entity Manipulation

4.8.1 Generalization and Polymorphism

Ontology languages strongly support the concept of generalization and inheritance. A concept and in particular a class can be defined as a subconcept of another and thus inherits the whole specification of its superconcept and can further specialize it. Consequently, ontology languages support the concept of polymorphism as instances of a subclass are as well instances of their superclass (cf. Section 3.3.2). This is in contrast to purely XML based approaches, where language concepts such as real inheritance support is missing (cf. Section 3.2.3). As it has been argued in [181], the static type bindings do not allow for effective polymorphism in XML. A given XML element is marked by only one specific tag, hence, in fact it has to remain monomorph. I.e. instances cannot be processed in a polymorph manner, because the processing of XPath or XSLT, etc. can only match a unique tree path of tags or respective attributes. For example an XPath expression */root/vehicle* cannot match an instance tagged by *<car>*, even if it provides all features of a vehicle. This fact can be ascribed to the structured-based approach of XML processing where a consequent polymorph type system is missing.

However, polymorphism provides an essential advantage taken into account the requirements posed by semantic mediation between overlapping domain ontologies. During the mediation process inheritance relations can be utilized to describe the relations between corresponding concepts defined in different ontologies. Furthermore, as information flow has to be realized across domain borders the mediation approach has to ensure that an information entity expressed as an instance of a concept from one domain ontology can be as well processed as an instance of the corresponding concept from the other overlapping ontology. Thereby, polymorphism can be exploited during the mediation process as instances of corresponding concepts from different ontologies can be handled as the same entity providing a polymorphic representation that conforms to both concept definitions. Hence, the different perspectives on the information space reflected in the different domain ontologies can be technically materialized by the exploitation of a polymorph type system for information entities.

4.8.2 Facet Analysis Classification

A further essential feature of Semantic Web technology, which can be exploited for the mediation mechanism, is the ability of facet analysis classification. Facet analysis classification can be regarded as one of the core concepts of ontology design. A concept or class is defined in terms of so called facets each member of the set of instances it describes needs to provide. Here, facets stand for clearly defined, mutually exclusive, and collectively exhaustive aspects, properties, or characteristics of a class or specific subject [182]. Any individual featuring such a specific set of facets is then classified as an instance of the class. This flexible classification approach based on the theory of facet analysis can be dated back to the work of the Indian librarian Ranganathan, who developed a faceted classification scheme in the 1930's [183]. This property-focused classification can be found in OWL in terms of class descriptions through property restrictions [184]. Such classes describe a set of instances or individuals that satisfy the restrictions. Thus, an OWL reasoner is able to analyze individuals and if they provide the properties declared in the restriction, they are classified as a member of the class. Consequently, applying facet analysis classification to information entities in a semantic integration scenario can support polymorph entity handling as described above. Whenever an instance of a concept from one domain ontology satisfies as well the property restrictions of the corresponding concept from the other overlapping ontology, this instance can be classified additionally to the type of the corresponding concept and thus becomes polymorph.

4.8.3 Declarative Rule-based Entity Manipulation

Another feature of Semantic Web technology, which is exploited in the developed mediation mechanism of this work, are description logic-based rules to express semantic bridges. Basically, semantic bridge rules are specified in such a way that they infer in terms of declarative entity manipulation a polymorph representation. The polymorph representation then provides the basis to allow the seamless information processing across different domains as described above. This ability of declarative rule-based entity manipulation fits well to the targeted approach of exploiting polymorphism for the mediation mechanism. Consequently, in the context of semantic mediation on domain level rule-based bridging axioms provide the option of choice besides other approaches discussed in Section 3.6.2 such as views and queries or dedicated mapping ontologies. Moreover, the advantage of the bridging axiom approach is that the use of a rule language does not only allow for describing the semantic relationships: Given a suitable inference engine also it allows for performing the transformations between related concepts. Then the reasoning of the described mappings can be applied straight forward as the transformation rules can be integrated into the regular ontology inference process such as classifying etc. Thus, no additional specific inference engine is needed for the mapping process such as required in the approach based on a particularly designed mapping ontology.

To summarize it can be stated that the semantic mediation mechanism can be based on the ability of Semantic Web technologies to express generalization and polymorphism and to infer this polymorphism by a combination of facet classification and declarative rule-based entity manipulation. In the following section these three features are combined to describe the concrete operation of a rule-based semantic bridge, which materializes the mediation mechanism between loosely coupled information models.

4.8.4 Operation of Semantic Bridges

The operation of semantic bridges is illustrated based on a simple example of two concept definitions: *Address* and *PostalAddress*. It is assumed that the two concept definitions although representing the same conceptual idea have been defined independently and by different domain experts in separate domain ontologies. The following Figure 4-18 shows an outline of these heterogeneous information models, which can be formalized in the OWL ontology language using defined classes (cf. Section 3.3.2). I.e. the concept definitions follow the principle of facet analysis classification and are formulated as a set of required properties that characterize this particular concept. Thereby, the important aspect is that the concept definitions differ in their semantic sub-graph as illustrated in the following Figure 4-18:

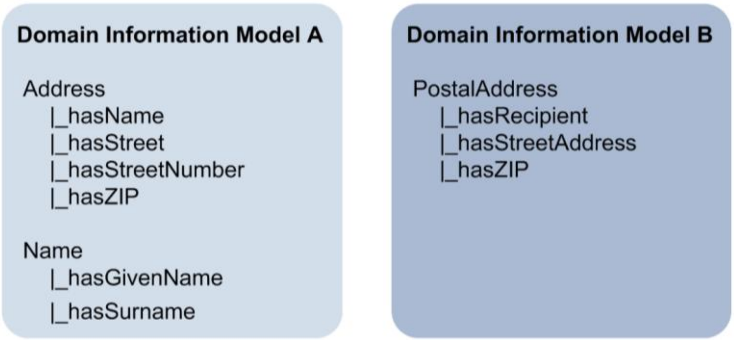


Figure 4-18 Heterogeneous Domain Information Models

Obviously, information entities instantiating either the *Address* or *PostalAddress* concept presented above cannot be exchanged between communicating partners by default, although they represent the same conceptual idea. For instance, in domain information model A the concept *Address* among others is defined in terms of four properties, whereas the property *hasName* refers to a further concept *Name* which is defined in terms of two properties. In contrast in domain information model B the semantically equal concept *PostalAddress* just features three properties containing the same information, however defined at a lower level of granularity. For example only one property *hasStreetAddress* is representing the *street name* and *street number* together and *hasRecipient* is representing the full name. Moreover, the two representations do not only differ in granularity, they also show a structural difference in the semantic sub-graph as the first ontology encapsulates *Name* as an extra concept, whereas the second ontology does not.

Consequently, the task of the semantic bridge is to provide a mechanism that mediates between these different concepts, so that instances of these concepts e.g. input parameters or output parameters of Web services can be seamlessly processed despite of their different representations. As mentioned before semantic bridges can be expressed as rules which are based on description logic implications so called forward-chaining rules. The following Figure 4-19 illustrates the first step of the semantic bridge operation where such rules are applied to infer additional properties.

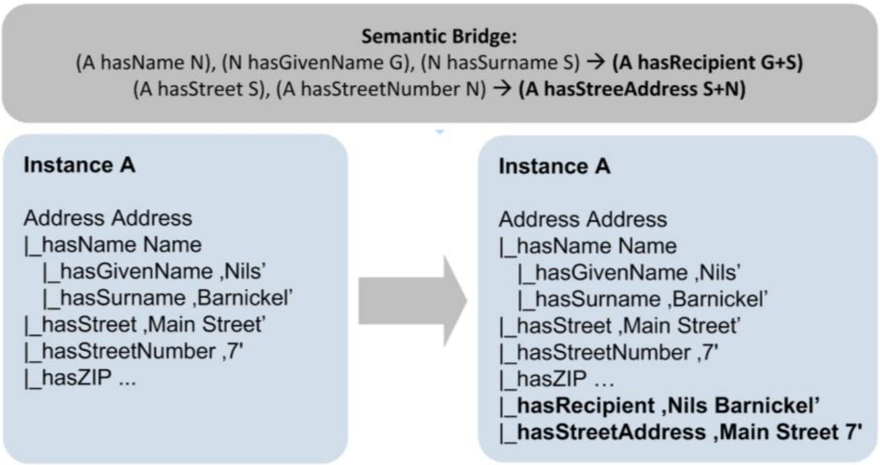


Figure 4-19 Semantic Bridge Operation (Step 1)

By applying the semantic bridge rules an instance of type *Address* is furnished with additional properties e.g. with *hasRecipient* combining the values of the *hasGivenName* and the *hasSurname* properties from the entity *Name* as illustrated in Figure 4-19. Analogically *hasStreetAddress* is inferred, which combines the values of the *Address* properties *hasStreet* and *hasStreetNumber*.

In a second step having the class definitions on hand, a reasoner is now able to classify the instance as a member of the defined class *PostalAddress*, since all required properties *hasRecipient* and *hasStreetAddress* are present. The following Figure 4-20 illustrates the second step of the semantic bridge operation:

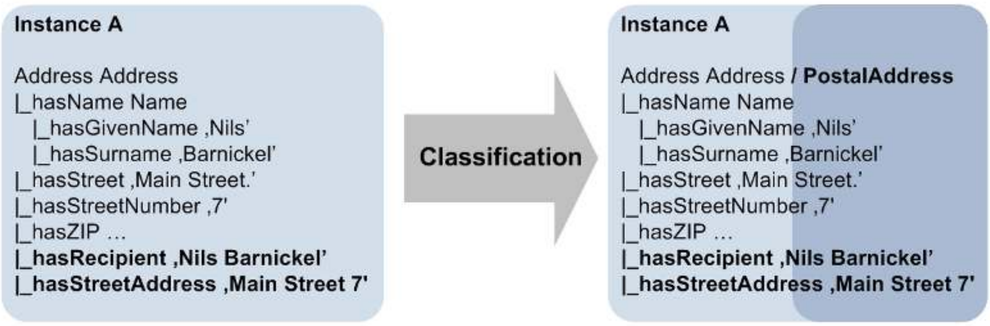


Figure 4-20 Semantic Bridge Operation (Step 2)

Thus, within the scope of a business process any service, independently to which domain it belongs, can now make use of this transformed and reclassified instance as it is polymorph of type *Address* and *PostalAddress*, i.e. semantic interoperability has been established.

4.8.5 Benefits of Developed Approach for Semantic Bridges

Firstly, the declarative approach enables the targeted separation of concerns. Accordingly, the absence of technical transformation code increases maintainability of the semantic bridges and thus improves efficiency.

Regarding effectiveness it can be noticed that using rules for describing semantic bridges enables expressive mappings including as description logic-based rules are computationally complete. Hence, any kind of mapping relation (one-to-one, one-to-many, many-to-many) can be described. A many-to-one mapping describes the case where two or more concepts from the first ontology need to be taken to represent one concept in the second ontology. Correspondingly, a many-to-many mapping can be understood. For example the presented mapping between the concept *Address* and the concept *PostalAddress* is actually a many-to-one mapping because not only the concept *Address* but also the concept Name, which it refers to are taken both to be mapped to the concept *PostalAddress*.

Moreover, the developed mediation mechanism features following advantage over other approaches applying bridging axioms for ontology mapping such as presented in [147]. By combining description logic based rules with facet analysis classification semantic bridges can be directly applied to instances of e.g. Web service outputs and do not need to be transformed into new instances of a so called merged ontology. Hence, during the mediation, the object identity can be kept and further processing of the original parameter is enabled. Accordingly, different actors of different domains, e.g. in terms of interacting services in a cross-organization business process, can have different views on the same service parameter realized through polymorphism.

However, in this context it should be noticed that the additional inferred properties create redundancy in the description of service parameters. Thus, in case of interacting services just read-only operations on the service parameters can be performed without the risk of inconsistency between the redundant descriptions. Nevertheless, parameter manipulation is not hindered by the approach. Before a service changes property values, it should be ensured that only the properties defined in the service’s domain ontology are present. Then in case of further service interactions again the application of semantic bridges should start from the beginning. This matches well with the characteristic of flexible binding in the definition of loosely coupled information models (cf. Section 4.6.3), which states that semantic mediation should be dynamically performed just right in the moment when information needs to be exchanged.

Furthermore, due to the nature of rule-based inference engines an additional benefit of the presented approach for realizing semantic bridges lies in its ability to easily derive transitive mappings. That means, that e.g. if domain ontology A is mapped to domain ontology B and domain ontology B is mapped to domain ontology C, then also A is automatically mapped to C. This benefit can further improve the efficiency of the semantic mediation mechanism on domain level as the number of required semantic bridges is further reduced.

The presented approach for realizing semantic bridges enables to express and perform mappings between concepts from different domain ontologies that cannot be expressed by common ontology language constructs. However, this does not imply that semantic interoperability is only ensured by the integration of semantic bridges. Common ontology language constructs such as equality or inheritance relations between concepts of e.g. top level ontologies and domain ontologies can also contribute to semantic interoperability. Such concepts from top level ontologies are equally used by different organization even originating from different domains. Thus, the standardization of these concepts which are not business context dependent ensures semantic interoperability for commonly used information entities. This is for example the case when using top level ontologies describing general aspects of Semantic Web services (cf. Section 3.4.2). This combination refers back to the hybrid ontology approach discussed in Section 3.6 about semantic information integration with ontologies as illustrated Figure 3-24. From this perspective semantic bridges provide the horizontal connections between the different local ontologies, whereas the vertical connections to the shared ontologies can be expressed with common ontology language constructs.

Finally, it should be noted that the here developed approach for semantic bridges as an instantiation of the semantic mediation concept has been presented and published at the

European Semantic Web Conference 2008 [185] describing as well the opportunities for its application within the SOA lifecycle as discussed in the following chapter.

4.9 Summary and Reflection

This chapter has presented the developed concept of semantic mediation between loosely coupled information models in SOA. The concept has been derived from several approaches analyzed in the state-of-the-art in Chapter 3 focusing on information integration and Semantic Web approaches in combination with concepts from service-oriented architectures. The principle idea has been to transfer the concept of loose coupling to the semantic level. According to the hypothesis of this work, the approach aims at providing not only an efficient but moreover an effective solution to the problem of semantic interoperability in large-scale SOA, which goes beyond dominant current Semantic Web service-based approaches.

In order to set the requirements for the solution targeted by the concept of semantic mediation between loosely coupled information models in SOA, the chapter has firstly listed required qualities which need to be covered by the developed concept. The main requirements which have been identified are: separation of concerns regarding business process expertise and between semantic and technical issues, anticipation of organizational boundaries to agree on and share information models across multiple domains, to address semantic integration on the higher abstract conceptual level, to remain consistent with best practice SOA methodologies, i.e. starting with business analysis leading to process models and keeping the process planning to a human process expert, and finally anticipating technological path dependency and thus to build upon existing technologies and standards of the World Wide Web.

Subsequently, the general idea of the developed concept has been outlined in order to provide an overview of the central aspects. These include mainly the shift from monolithic to loosely coupled information models combined with the approach to address semantic integration on a higher abstraction level. Thereby, the notion of a shift to a higher abstraction level has been elaborated along two dimensions: On the one hand, semantic integration is shifted from the schema or structure-based logical abstraction level to the conceptual abstraction level. And on the other hand, semantic mediation is performed on domain level instead of recurrently on application or process level. The following sections then have refined the general idea and provided detailed descriptions and argumentations of the respective conceptual parts.

Firstly, the limitations of standardization with regard to semantic interoperability have been discussed to point out the demand for loosely coupled information models. In order to clarify the understanding of semantic standardization, this general approach has been put in contrast to the targeted approach of semantic mediation for bridging the semantic interoperability gap. Furthermore, the differences between the advantages of technical standards in contrast to implications of semantic standards have been shown. On the one hand, technical standards principally increase productivity by rationalization as integration and adaptation costs for different interfaces, protocols and coding formats can be reduced. On the other hand, standards on the semantic level targeting a universal scope lead to inflexible monolithic information models, which cannot adequately respond to changing business requirements. Taking this into account, analogies between general standardization processes and semantic standardization could be derived. It has turned out that similar to technologies which face high innovation and frequent changes the consensus degree for agreeing on semantic standards is limited. Consequently, it could be concluded that the adequate scope of semantic standards has to be restricted to an organizational feasible extend and should not target the level of universal adoption.

Having derived a first rational for multiple coexisting information models, the subsequent section has further substantiated the argumentation by analyzing the underlying reason for context-dependency of information models. Therefore, a general model theoretic perspective has been presented that applies a model of conception to information models. The main conclusions have been that context dependency of models is an inherent characteristic of models and a determinant factor to cope with complexity, which is particularly valid for information models. Furthermore, the model theoretic approach has shown that the context cannot be regarded as an objective issue but is determined by a subjective conception. Consequently, information models of a certain domain are not unique and uniform but may differ dependent on the different actors developing and using it. Finally, the analysis has shown that information models are not only related to their domain but also to a specific purpose or respectively to concrete IT applications. Thus, due to their very nature, information models cannot be general, unique and uniform but rather they are diverse, heterogeneous and occur as multiple possibly overlapping variations.

Taken into account this requirement for the coexistence of multiple overlapping information models, the need for a mechanism to overcome the heterogeneous conceptualization in a concrete integration scenario has been pointed out. Based on these findings, the transfer of the principle of loose coupling to the semantic level has been discussed. Three fundamental and transferrable characteristics of loose coupling have been extracted from an analysis of different definitions of the functional dimension of loose coupling, namely: autonomy, flexible binding and encapsulation. Transferred to the semantic level, autonomy of loosely coupled information models can be understood as the independence of their management which is under the control of different ownership. Furthermore, flexible binding of information models describes the non-permanent but dynamic nature of the coupling mechanism between two information models. Finally, as well encapsulation is reflected on the semantic level. The main idea for this point is that the conceptual abstraction level of information models provides an encapsulation of information model internals and thus enables their loose coupling independently from internal changes or representation formats. Based on this discussion, a definition of loosely coupled information models has been provided that reflects these three central aspects.

As the main goal of this work is to contribute to a solution for the semantic interoperability problem in SOA with particular focus on effectiveness and efficiency, the subsequent section has reflected the proposed conceptual solution on this regard. A trade-off between effectiveness and efficiency has been pointed out. On the one hand, an extreme application of loose coupling given by means of a point-to-point mediation of organization’s information models provides an effective solution but does not scale efficiently as in principle integration costs increase with quadratic complexity. On the other hand, the opposite approach of semantic standardization via a single pivot ontology provides an efficient but no effective solution. Efficiency is ensured as only linear costs for mapping each organization’s information model to the pivot ontology are required. However, the required coordination efforts for consensus finding between multiple actors lead as well to quadratic complexity. Thus, it has been shown that the pivot ontology approach just provides a shift from operational integration efforts for seamless information flow to organizational coordination efforts required for the alignment of the different perspectives of the involved actors. In order to overcome the drawbacks of the analyzed approaches as well as exploiting their respective advantages, a combination of the two semantic integration strategies has been presented. Basic idea of this derived balanced approach is to target semantic mediation on the domain level. Thus, firstly the required number of semantic mappings is reduced as only mediation between different domains instead of between each single organization has to be covered. And secondly, the involved actors are reduced, as with the limited scope of the standardized domain information models as well the number of involved actors can be limited to a feasible extent.

However, even if providing a more balanced approach, it has been argued that the approach of semantic mediation on domain level just provides a shift within the trade-off between effectiveness and efficiency and that the approach cannot resolve it in principle. Nevertheless, it has been shown that the dedicated application of Semantic Web technologies as exploited in the concept of loosely coupled information models enables an alleviation of the trade-off. As semantic mediation is targeted on the conceptual level, consequently mapping efforts are reduced substantially as technical issues can be handled with enhanced transparency. Moreover, efficiency is improved, as the mediation between different heterogeneous representations in overlapping conceptualizations is only done once on the domain level instead of performing it repeatedly in concrete application or process scenarios. Regarding effectiveness, it has been argued that the formal nature of conceptual models instantiated by the utilization of ontologies eases the coordination and consensus finding process.

Finally, the chapter has presented the developed semantic mediation mechanism which enables the proposed concept of loose coupling of information models in SOA. The semantic mediation mechanism instantiates the general idea of semantic bridges based on selected Semantic Web technologies, in particular ontologies and description logic rules. The requirements for the semantic bridges to target loose coupling of domain ontologies have been addressed by exploiting features such as generalization and polymorphism, facet analysis classification and declarative rule-based entity manipulation. In order to show how these features can be combined together to serve the desired goals, the operation of semantic bridges has been explained and the benefits of the developed approach could be pointed out.

By means of a connecting step between theory and experiment, the following chapter maps the here presented theoretical concept to the concrete application domain of SOA. Therefore, a dedicated semantic mediation methodology is developed that determines the basic steps relevant for the application of the concept of semantic mediation to the SOA life-cycle.

Chapter 5

Methodology and Functional Architecture for Semantic Mediation in SOA

5.1 Overview

This chapter presents the developed semantic mediation methodology which maps the concept of loosely coupled information models to the basic phases of the SOA life-cycle. The relevant phases of the SOA life-cycle where mediation between heterogeneous information models is required are identified and it is discussed how the before developed conceptual solution can be applied.

The chapter starts by refining the general conceptual requirements identified in Section 4.2 with regard to concrete conditions and implications for the semantic mediation methodology. Furthermore, domain specific considerations are derived in terms of an actors model describing the involved stakeholders relevant for semantic mediation in SOA. Subsequently, an overview of the individual steps of the semantic mediation methodology is given, which is aligned to the basic phases of the SOA life-cycle. The following sections then discuss the particular methodology steps in more detail with regard to performed tasks and required functionalities. In order to prepare an experimental confirmation of key steps of the methodology in terms of a prototypical semantic mediation toolkit, as well a high-level view on the functional architecture for each methodological step is derived. Therein, it is distinguished between methodological steps which can be addressed sufficiently with existing work including respective tools and steps which cannot be mapped to available functionality. These missing functionalities are then addressed in the subsequent Chapter 6 presenting the developed prototypical instantiation of the semantic mediation toolkit based on the in this chapter derived requirements and functional architectures. Finally, a conclusion and reflection of the chapter is provided.

5.2 Methodology Requirements and Domain-specific Considerations

The general conceptual requirements for semantic mediation between loosely coupled information models have been already discussed in Chapter 4 (cf. Section 4.2). They need to be refined, in order to derive concrete implications for their practical application to the SOA life-cycle (cf. Section 2.4).

The first conceptual requirement has addressed the reduction of complexity in semantic integration in terms of separation of concerns. Consequently, the demand for overlapping skills of involved stakeholders should be reduced. Thus, the semantic mediation methodology should be based on a dedicated actor model that reflects the different expertise requirements of the assigned roles. In particular, the responsibilities of the defined roles should be highly disjoint, in order to minimize the demand for overlapping skills that are a major factor of complexity in semantic integration.

The second conceptual requirement has dealt with organizational boundaries and limitations in the scalability of community processes to ensure agreement and standardization of cross-organizational information models. As the concept of loosely coupled information models is applied in the methodology, this requirement is already reflected per se and no further implications for the semantic mediation methodology can be derived. However, the methodology should address this cross-organizational environment by reflecting that the involved stakeholders identified in the actor model may origin and operate in different organizational contexts.

A further conceptual requirement has been the shift of abstraction level in semantic integration. On the one hand, this requirement is well reflected by the concept of loosely coupled information models. The semantic mediation mechanism exploits ontology-based conceptual information models, which are independent from internal more technical representations of information models. On the other hand, the shift of abstraction level has also been described as the aim for addressing semantic integration just once on domain level instead of recursively on application or process level. This is also reflected by the first methodology requirement taken into account the targeted actor model which should address the different responsibilities between the roles of domain experts and business process experts. Therefore, besides the consequent reflection of the shift of abstraction level no further explicit implications for the semantic mediation methodology can be derived from this conceptual requirement.

Another central conceptual requirement has been the consistency with best practice SOA methodologies. In particular, this means that the concrete service composition is derived from a before undertaken business process analysis and business process modeling phase. Accordingly, the semantic mediation methodology has to be strictly aligned to the SOA life-cycle phases, whereas these phases where semantic heterogeneity is a challenge need to be covered.

The final conceptual requirement has been the constraint to respect technological path dependency. This means that the approach should be based on the well-established concepts and standards of the World Wide Web as it constitutes the dominant IT infrastructure for cross-organizational interaction. Furthermore, as the semantic mediation concept should be applied to the realm of SOA, this means more specifically that the approach should rely on the existing XML-based Web service technologies as standardized by the W3C or OASIS (cf. Section 3.2.2). The on top of this applied Semantic Web technologies therefore should be as well based on the W3C recommendations (cf. Section 3.4.2).

Domain-specific Considerations

Besides the refined methodology requirements further domain specific considerations from the realm of SOA can be derived. Therefore, the following domain actor model is defined, in order to specify the involved stakeholders relevant for the semantic mediation methodology with their roles, responsibilities and general skill profiles. Often a domain actor model is also referred to by the term domain model [186]. However, with the focus on domain information models the term domain model might be misleading and thus in the following just the term domain actor model is used. Figure 5-1 below illustrates the five identified actors and their interrelations: the domain information model expert, the business process expert, the service developer, the service composer and finally the service or process consumer:

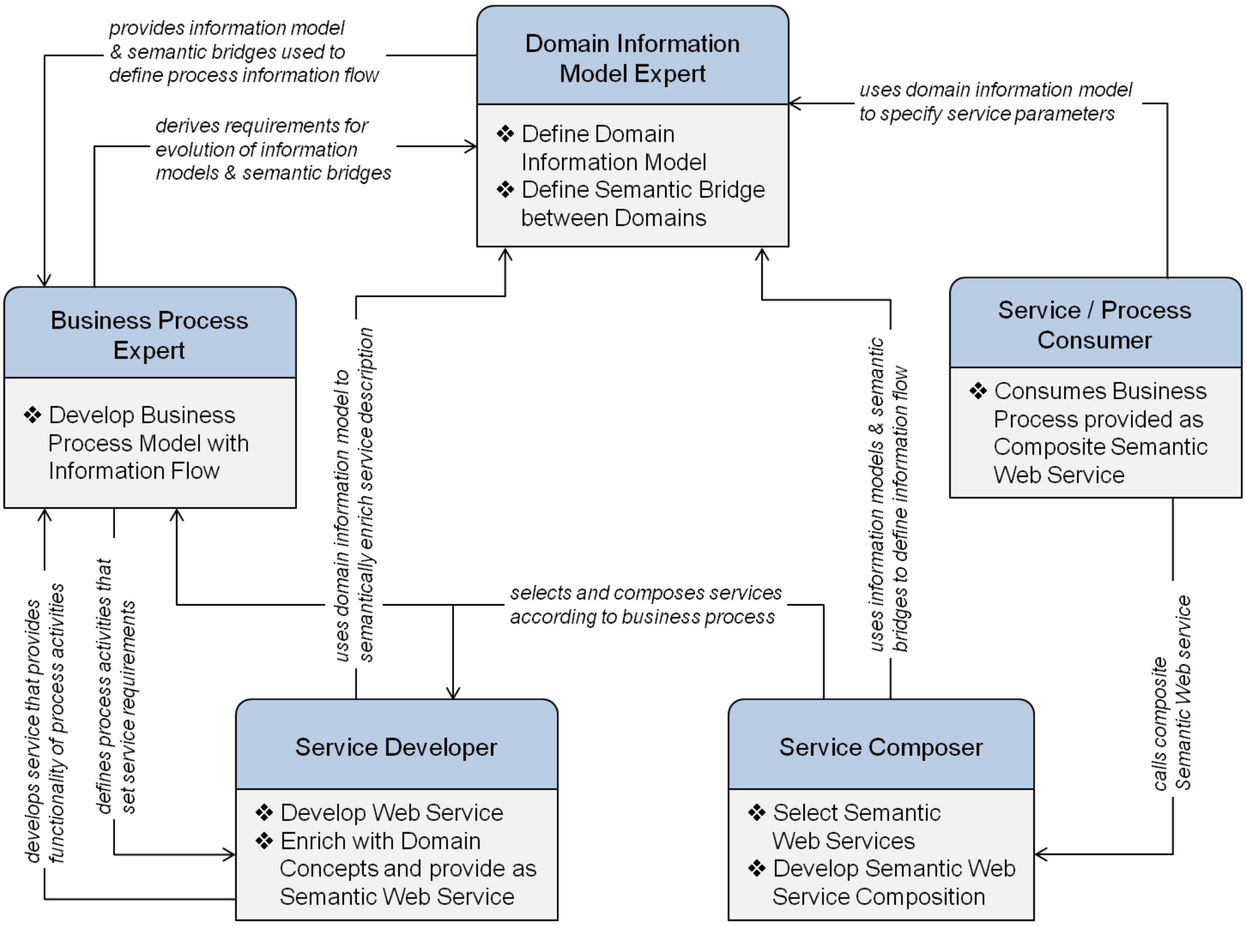


Figure 5-1 Domain Actor Model for Semantic Mediation Methodology

A domain information model expert is responsible for developing an information model covering a particular domain. Therefore, he needs to have comprehensive general business knowledge of the information requirements for the interactions of organizations in this domain. He works on the conceptual level and specifies the domain information model by using ontology languages. Moreover, domain information model experts from different domains need to collaborate, in order to define semantic bridges between different domains. The defined domain information models and corresponding semantic bridges are provided to the business process experts.

A business process expert is in charge of analyzing and modeling concrete business processes of an organization. From the here focused perspective of information models, he is particularly responsible for modeling the information flow within business processes. Accordingly, he needs to have comprehensive knowledge about the concrete information requirements for the particular business activity of the organization. Furthermore, the business process expert contributes to the evolution of domain information models by deriving requirements for missing concepts necessary for the description of business process information flow. Moreover, in order to enable cross-organizational business processes, business process experts from different organizations need to collaborate together. To define the cross-organizational information flow they make use of the provided semantic bridges to overcome semantic heterogeneities. Finally, the business process expert also sets the general requirements for the service development by specifying the individual activities within a business process model.

Accordingly, a service developer is responsible to develop the functionality required for a particular business process activity and to provide it in terms of a Web service. Therefore, he needs to have in-depth technical capabilities for Web service implementation and provision. Furthermore, in order to enable the shift of abstraction level in semantic integration the service developer enriches the services parameters with concepts defined in the domain information model and thus builds the required Semantic Web services.

A service composer then selects from the developed and provided Semantic Web services these candidates which are required to instantiate the business process in terms of a Semantic Web service composition. On the one hand, the service composer needs to have solid understanding of the business process to be implemented by the services composition. On the other hand, his skill profile requires general technical competencies to develop the concrete Semantic Web service composition with specific tool support. Such tool support particularly exploits the semantic layer given in terms of semantically enriched service descriptions and semantic bridges to ensure highly transparent semantic integration for the service composer.

Finally, the actor model contains a service or process consumer. His skill profile also requires general technical competencies as the concrete business process instantiation needs to be called in terms of a composite Semantic Web service request. Therefore, the concepts of the domain information model are used to specify the service parameters.

5.3 Semantic Mediation Aligned to SOA Life-Cycle

Based on the domain actor model the semantic mediation methodology can be defined. The concept of semantic mediation is mapped to relevant phases of the SOA life-cycle. These phases of the SOA life-cycle where mediation between heterogeneous information models is required are identified as steps for the semantic mediation methodology and it is discussed how the before developed conceptual solution can be applied to provide the aimed additional value. The following Figure 5-2 presents an overview on the semantic mediation methodology. The seven steps of the methodology are briefly outlined, whereas the subsequent sections cover the individual steps in detail.

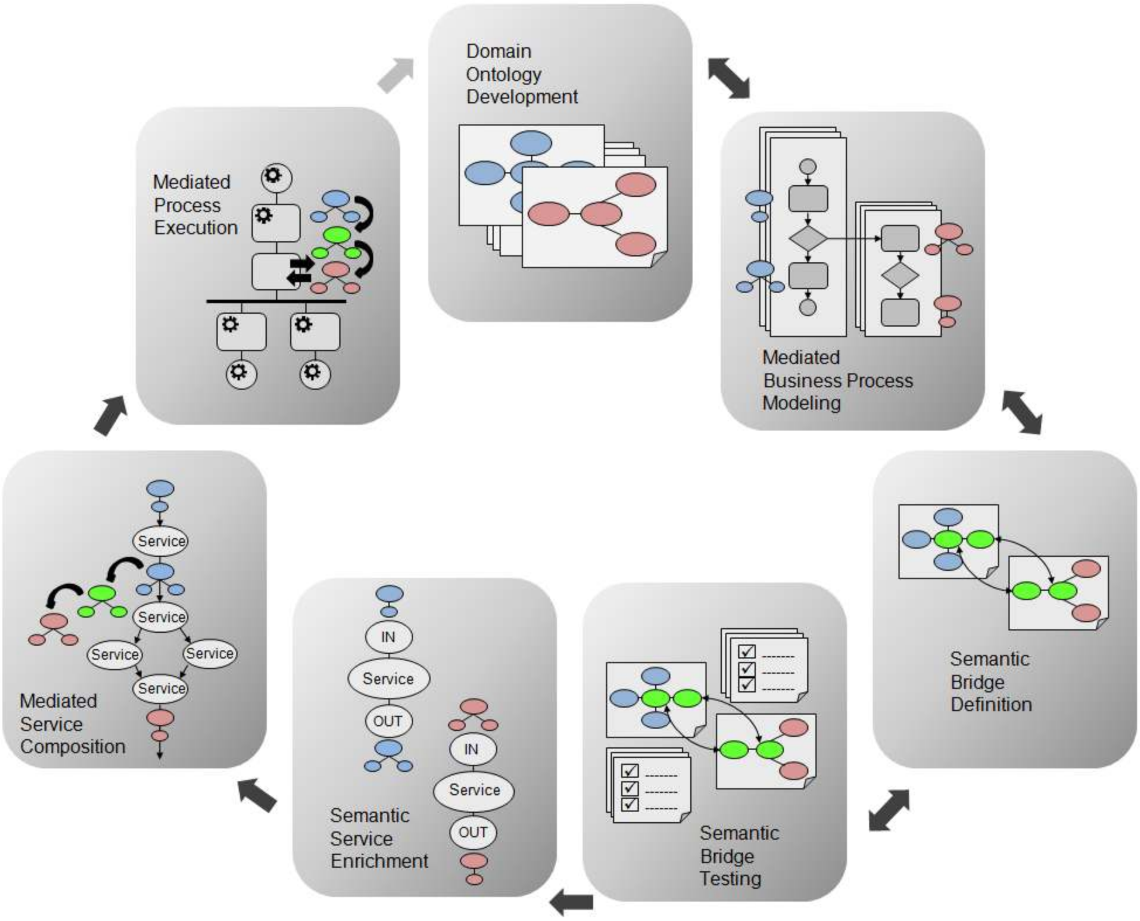


Figure 5-2 Semantic Mediation Methodology

The first step of the semantic mediation methodology starts with enabling the shift onto a higher abstraction level by creating the conceptual information models, which are the central artifact throughout the whole methodology. Therefore, domain information model experts from each domain develop autonomously so called domain ontologies. Thereby, the focus is on the requirements and usage within the respective domain, yet leaving aside any cross-domain aspects.

In the following the methodology steps are described in a sequential order for the purpose of providing a better understanding. However, this order should not be regarded too strictly. Rather, the methodology contains steps which can be also performed in parallel or which feature mutual feedback relations to enable an iterative approach. Especially, the first four steps can be performed in an iterative manner. Accordingly, this is highlighted with double arrows and the possible iterations are mentioned in the descriptions of the individual steps.

The developed domain ontologies provide an input to the next step of the methodology, namely the mediated business process modeling. Aligned to the central SOA phase of business process analyses and business process modeling, this step targets the required high-level modeling of information flow within business processes. Therefore, the business process expert defines a business-oriented information flow on the conceptual level using the concepts from the before developed domain ontologies. Taking into account cross-organizational business processes, heterogeneity between different information models has to be addressed. Therefore, semantic bridges focused in the subsequent methodology step are required to identify and describe information flow across heterogeneous information representations. Moreover, coming from the perspective of agile development and continuous maintenance, information models and correspondingly semantic bridges between them need to evolve over time. According to process-orientation this evolution should be driven by requirements derived from business processes. Consequently, mediated business process modeling not only includes the exploitation of domain ontologies and semantic bridges but also provides specific features for their requirement engineering during business process modeling.

Having already identified certain requirements for semantic bridges during the modeling phase of cross-organizational business processes the dedicated methodology step referred to as semantic bridge definition covers their comprehensive identification and specification. For this step domain information model experts from different domains need to collaboratively develop semantic bridges based on (tool-supported) ontology mappings (cf. 3.6.2).

The roles of semantic bridge developers in terms of domain information model experts on the one hand and semantic bridge users in terms of business process experts and service composers on the other hand are carried out by different actors. They might be even mutually unknown in cross-organizational SOA landscapes; which makes the consideration of trust in the quality of the developed semantic bridges an essential factor. Therefore, the third step of the semantic mediation methodology focuses on the evaluation and particularly the testing of semantic bridges. As it can be shown that in principle no formal verification of the correctness of semantic bridges can be provided, the focus is put on how to apply concepts from software testing to the systematic testing of semantic bridges and the underlying ontology mappings.

After having provided the foundation for addressing semantic interoperability on a higher abstraction level by means of process-oriented domain ontologies and quality assessed semantic bridges between them, the next methodology step is to lift the service descriptions as well on the higher conceptual abstraction level. Therefore, the service developer, responsible for developing Web services that provide the corresponding functionality required by the individual business process activities, also has to provide the semantic service enrichment. This means that the service developer enriches the service description with the concepts defined in the respective domain information model, in order to lift the traditional XML-based Web service description to a so called Semantic Web service description (cf. Section 3.4).

In the following methodology step the semantic service composer combines the before provided Semantic Web services to a mediated service composition. The composition instantiates the previously developed cross-organizational business process model by defining a concrete so called Semantic Web service orchestration plan that can be executed in a process engine. In order to achieve a high level of automation, the semantically enriched service descriptions can be exploited for service selection based on semantic matchmaking. Furthermore, the additional semantic layer is utilized to define the concrete information flow between the involved service input and output parameters. Taking into account that the services are described with concepts from heterogeneous information models, the before provided semantic bridges are used to enable a seamless design of this concrete information flow, whereas semantic integration is handled transparently.

The final methodology step deals with the runtime execution of the Semantic Web service orchestration. Thereby, well established standards for workflow execution are considered. On this regard especially the de-facto industry standard BPEL is addressed, which as according to traditional Web service technology relies on the XML meta-data model. Hence, the challenge to work consistently on the different meta-data models needs to be reflected during runtime and consequently Semantic Web technology has to be incorporated into BPEL-based process runtime middleware. Another challenge thereby lies in assuring a reasonable performance during the rule-based inferencing process, which still often remains a bottleneck of Semantic Web technology.

Usually, after process execution a subsequent phase dealing with process monitoring and process optimization follows, in order to iteratively close the SOA life-cycle and derive new requirements for process evolution. However, this phase has not been covered with a particular semantic mediation methodology step. In fact, the analysis in terms of monitoring and optimization of process data flow combined with heterogeneous information from other data sources within or across organizations requires semantic interoperability, too. However, in order to operationalize such analytical processes as well a service-oriented approach would be followed and consequently a re-instantiation of the seven described methodology steps is required.

Moreover, other possible phases of the SOA life-cycle have not been addressed specifically in the methodology. This includes e.g. dynamic service discovery or dynamic service negotiation, ranking or contracting. Even though such approaches have shown first results in academia (cf. Section 3.4.3) they do not match adequately with current SOA best-practices. Thus, they have not been focused in this work according to the requirements set for the general approach as well as for the semantic mediation methodology.

5.4 Domain Ontology Development

This section describes the semantic mediation methodology step of domain ontology development and the existing work in terms of methods and tools which can be applied to perform this methodology step.

5.4.1 Goals and Tasks

Ontology development is an active field of research. It ranges from the usage of ontologies to ensure a common understanding among human actors to information processing, whereas ontologies provide the means upon which semantic reasoning techniques can be applied for higher degree of automation (cf. Section 3.3.1). Accordingly, the major merit of ontologies is that they can bridge the gap between the real world and IT systems [187].

An important question regarding the concept of loosely coupled information models and semantic mediation on domain level targets the adequate scoping of domains. The question may arise from two perspectives. On the one hand, stakeholders who intend to cooperate in cross-organizational business processes may face defragmented and non-integrated information models. Therefore, they need to define which parts can be aligned to canonical information models based on an ontology or where mediation between independent domain ontologies is the more feasible approach. On the other hand, stakeholders who are already engaged in cross-organizational business processes and share an aligned complex information model may face the lack of ability for efficient and effective maintenance and evolution. In this situation they need to define how the overall information model can be broken down into loosely coupled domain ontologies to ensure more mutual independence and adequate reaction to local requirements.

From both perspectives the way to determine the adequate scope is inherently dependent on the specific scenario. Thus, this question cannot be covered exclusively within the scope of this work, which targets a general approach for achieving semantic interoperability in SOA independent from any particular application scenario. However, it can be stated that the scope of the domains should correlate with existing organizational structures as the consensus finding process requires a certain degree of organizational binding (cf. Section 4.4.4). The following Figure 5-3 Scoping of Domain Ontologies Aligned to Organizational Structures illustrates this correlation:

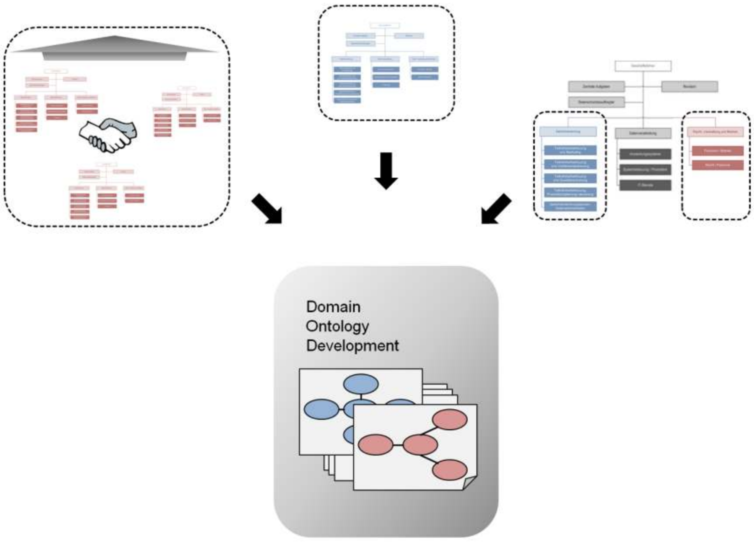


Figure 5-3 Scoping of Domain Ontologies Aligned to Organizational Structures

In one case different organizations may exhibit organizational ties in terms of an umbrella organization or a well-integrated business association. Such an environment can be an enabler for a consensus finding process leading to a shared canonical information model. In this case one domain may cover several organizations that develop collaboratively their domain ontology. In another case one domain may just comprise a single organization. Even one large organization may be divided into different domains to address the possibly divergent requirements from different departments or business units on information models.

At the end, in all cases the domain information model experts define a conceptual model of the identified domain that represents an agreed consensus among the involved parties within this domain. However, the domain ontologies should not be developed in terms of a greenfield approach. Usually, at least parts of domain information models are already available in terms of conceptual e.g. UML or ER models or as well in terms of database schemas or XML-schema representations. Therefore, the main focus of domain ontology development is not to reinvent existing information models but to lift existing parts to an ontology level and ensure a consistent and high quality conceptualization backed by formal logics. Moreover, taking into account iterative development and continuous maintenance, domain information models and their representation in terms of domain ontologies need to evolve over time. According to process-orientation this evolution should be driven by requirements derived from business processes. Hence, domain information models can be developed in a so called top-down and bottom-up mixed approach, which has been presented in [188]. This leads already to the next methodology step about mediated business process modeling, which is presented in detail in the following section covering as well the process-oriented requirement (re-)engineering of domain information models.

5.4.2 Existing Work

As having stated above, ontology development is an active research field and consequently an impressive amount of work is available ranging from concepts and methodologies on how to structure the development process to concrete tools and ontology editors. A general introduction to ontology development is given in [189]. A specific approach and methodology addressing the cross-organizational nature in terms of collaborative ontology development is presented in [190]. Concrete tools are available in terms of the mainly academic-driven Protégé ontology editor, which provides comprehensive functionality and visualization as illustrated in Figure 5-4. Within the last years as well commercial tools such as SemanticWorks [192] have become available, which demonstrates the maturity and first industry adoption within the field of ontology development and usage. Taking into account this stage of existing work and available tools the here presented step of the semantic mediation methodology about domain ontology development can be instantiated with existing approaches and tools. Therefore, no further specification of required functionality for this step needs to be provided.

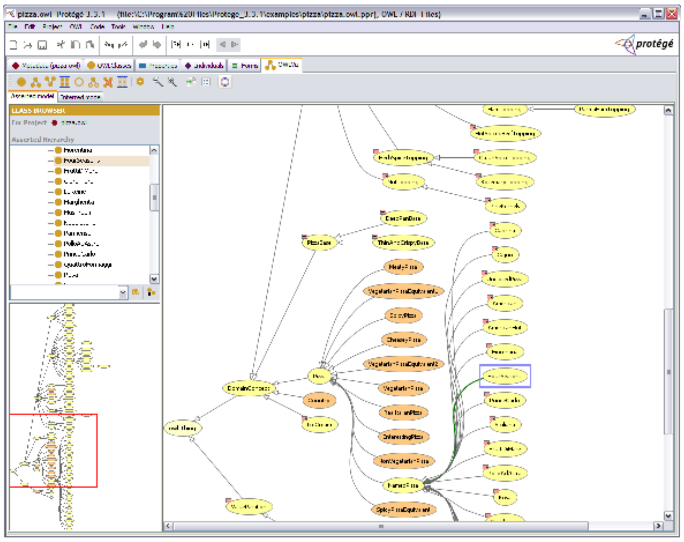


Figure 5-4 Protégé Ontology Editor [191]

5.5 Mediated Business Process Modeling

This section describes the semantic mediation methodology step of mediated business process modeling and the required functional architecture of a tool to perform this methodology step. The concrete realization of this functionality is then further described in context of the developed semantic mediation toolkit in Section 6.3.

5.5.1 Goals and Tasks

As discussed in Section 5.3, the SOA life-cycle starts from the business perspective on how processes can be supported by IT systems. In this context, taking into account cross-organizational business processes, the challenge of heterogeneous information models also affects the design phase of business process modeling and in particular the definition of information flow. Usually, business process experts use business-oriented high-level descriptions of information entities which are non-formal and non-technical to define the general information flow in business processes. However, in cross-organizational business processes e.g. the usage of mismatching terms for semantically equal information entities can hinder the sound design of information flow across organizational borders. Moreover, the non-formal nature increases the so called business-IT gap as the used terms are not explicitly linked to already existing information or data models of the organizations. This causes additional efforts and iterations for aligning the top-down requirements-driven business perspective with the bottom-up IT perspective focusing on reuse of existing assets.

Whereas the previous methodology step has provided the foundation for semantic mediation on a higher abstraction level, the goal of this step is to exploit the ontology-based information models for the sound design of information flow in business process models. The idea is that already during business process modeling the before developed ontology concepts are used to define the information flow on a non-technical conceptual level suitable for business process experts. However, due to the formal nature of ontology-based information models a consistent link between the business or conceptual level and the underlying technical information or data models can be derived.

Furthermore, having the formal domain information models at hand facilitates the mediation between heterogeneous conceptualizations by different organizations by integrating the developed mechanism of semantic bridges. Thus, the business process expert is enabled to seamlessly design the cross-organizational information flow, whereas semantic heterogeneities can be handled transparently based on semantic technology-based tool support.

Moreover, taking into account as well the perspective of agile development and continuous maintenance, domain information models and correspondingly semantic bridges between them need to evolve over time. According to process-orientation this evolution should be driven by requirements derived from business processes. Consequently, mediated business process modeling not only includes the exploitation of domain ontologies and semantic bridges as described above but also should provide specific features for their requirement engineering during business process modeling. This demand-driven evolution includes for example the possibility for the process expert to specify information entities which are not already reflected within the ontology-based domain information model as well to identify missing semantic mapping rules between information entities of different domain information models not reflected in the available set of semantic bridges.

The following Figure 5-5 illustrates the three major tasks for the business process expert during mediated business process modeling:

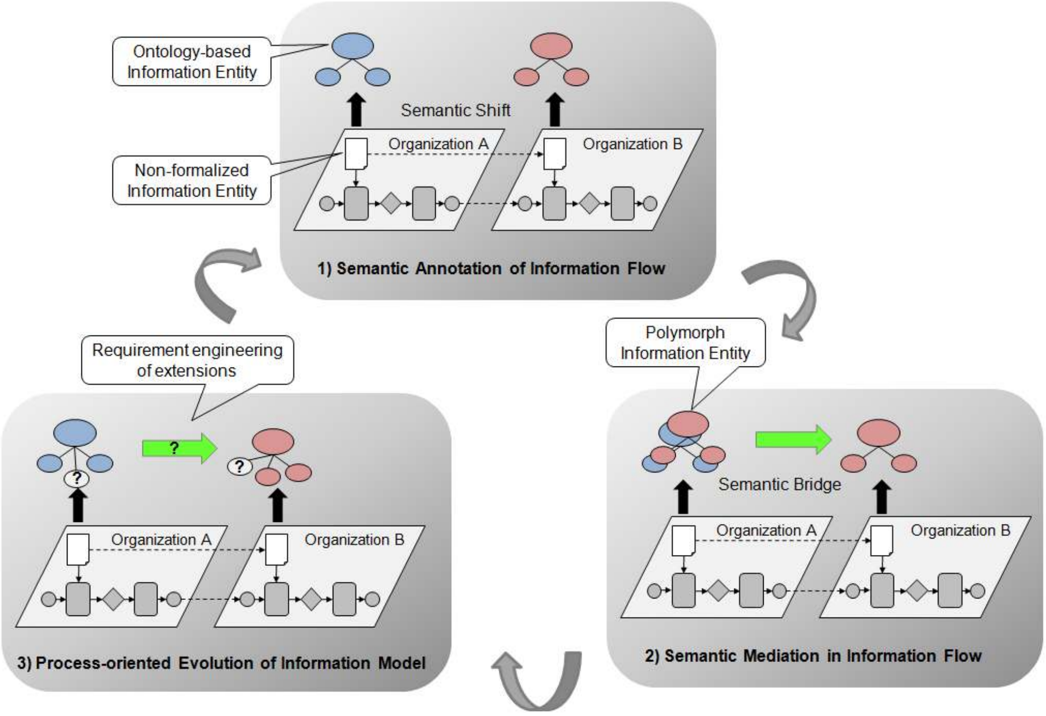


Figure 5-5 Semantic Mediated Business Process Modeling

The first task deals with the semantic annotation of information flow in business processes within one organizational domain. The business process expert defines a business-oriented information flow on the conceptual level using concepts from the before developed domain ontology to annotate the non-formalized information entities and thus shifts them onto the higher semantically explicit level. This task requires a generic extension of the business process modeling notation enabling to visualize the higher expression level in terms of semantic sub-graph of information entities in contrast to flat representations provided in current modeling notations. However, it is important to note that during this task each organization just has to use their specific domain ontology without any dependencies to other domains.

Then, in the second task semantic bridges are applied to the ontology-based information entities, in order to obtain polymorph information entities and thus overcome semantic heterogeneities between the different domain ontologies. In particular, semantic mediation based on semantic bridges can be exploited to suggest matching information entities in process parts of different organizations. This enables seamless information flow design keeping information representation differences transparent for the process expert.

The third and last task focuses on the process-oriented evolution of information models and semantic bridges. Missing information entities and semantic bridges required for the information flow in the concrete business process but not already reflected in the existing domain information models can be specified by the process expert. Then, in a further external step they can be defined by domain information model experts in terms of iterative and demand-driven development. Consequently, these evolutionary developed ontologies and semantic bridges can be further utilized for semantic annotation and mediation of the information flow and thus close the so to say micro-life-cycle of mediated business process modeling.

Even though several approaches for integrating semantic technologies into business process modeling exist (cf. Section 5.5.3), the presented approach for mediated business process modeling requires a dedicated solution that supports semantic mediation based on semantic bridges. Taking into account that the field of business process modeling is well covered by mature industry tools and products, the required tool-based realization of the methodology step of mediated business process modeling should reflect such existing work. Therefore, the following section discusses a possible functional architecture of a tool as an extension of a state-of-the-art business process modeling tool with integrated features for semantic mediation during information flow design. The concrete realization of this functionality is then further described in context of the developed semantic mediation toolkit in Section 6.3.

5.5.2 Functional Architecture

In this section the functional architecture for the mediated business process modeling tool is presented and explained. As the tool should be realized as an extension of a state-of-the-art business process modeling tool, the architecture has to incorporate an abstraction of it, in order to remain independent of any concrete tool or product. Accordingly, based on the goals and tasks described in the previous section a systematic use case analysis has been performed, which is described in detail in [193]. Based on this requirements engineering the following functional architecture has been developed which provides an overview of the main functional components and how they interact with each other:

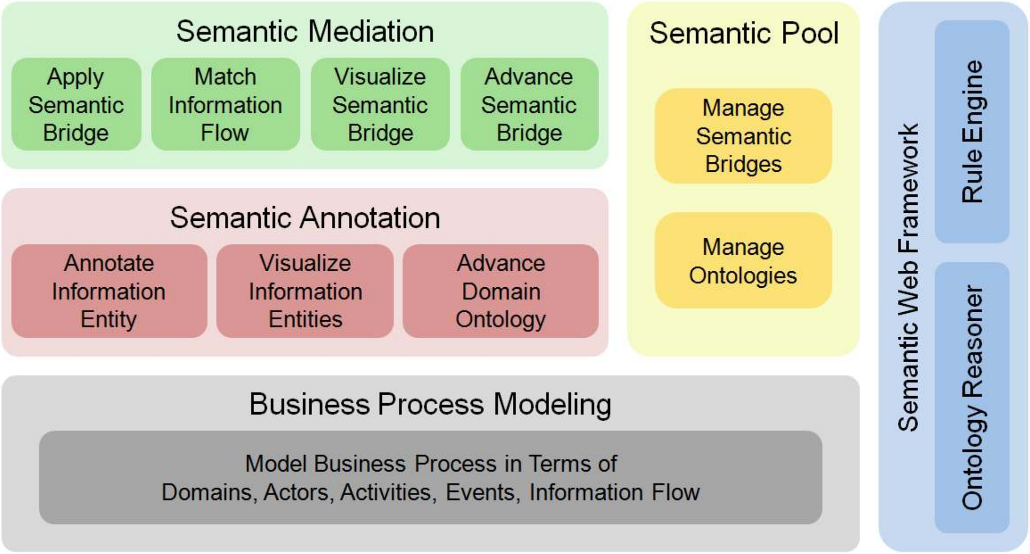


Figure 5-6 Functional Architecture Mediated Business Process Modeling

The resulting architecture is based on state-of-the-art functionality for business process modeling. On top of this bottom layer concerned with business process modeling two more layers are added, which are enabled by means of Semantic Web technologies provided by the vertically shown Semantic Web framework. The first additional layer provides the semantic annotation of information entities within the business process information flow. The second additional layer then contains the functionality for semantic mediation based on polymorph information entities to facilitate sound design of cross-organizational information flow. A further vertical layer named semantic pool provides complementary functionality to the before presented layers in terms of management functionality for utilized ontologies and semantic bridges. In the following these five basic components are described in more detail:

Business Process Modeling: Provides state-of-the-art business process modeling functionality based on a visual editor for modeling domains, actors, events, activities and information flow, whereas the functionality should be abstract from any concrete modeling notation such as BPMN [194] or EPC [195] as multiple notation languages should be supported.

Semantic Annotation: Provides the functionality to annotate basic information entities in cross-organizational process models with concepts of domain ontologies. Further functionality is provided to advance the semantic annotation in terms of process-oriented requirement engineering for extensions of domain ontologies. In particular, missing information entities or properties of them as well as semantic bridges or missing individual mappings within them required for the process information flow can be specified by the process expert. Finally, the higher semantically explicit level of information entities should be visualized in the business process modeling notation. Therefore, a generic extension is provided that highlights the semantic sub-graph of information entities in contrast to flat representations provided in current modeling notations. For example a data object *Partner* modeled in BPMN can be annotated with a domain ontology concept describing a *Partner* containing three properties. Accordingly, the three properties can be visualized in the extended notation. The following Figure 5-7 illustrates this generic semantic extension of business process modeling notations:

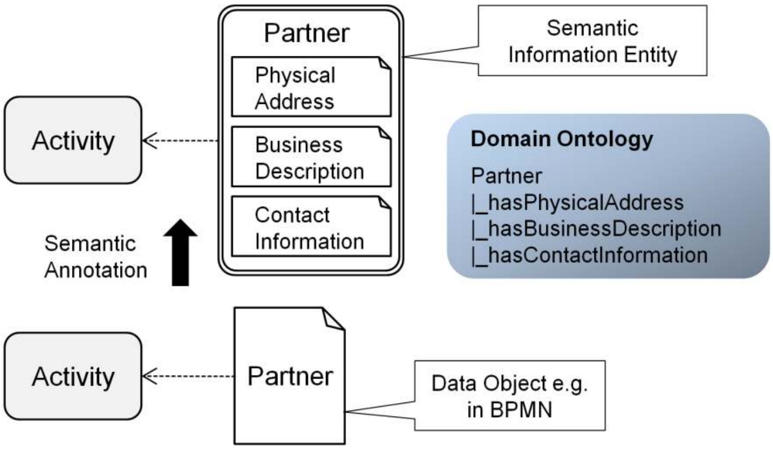


Figure 5-7 Semantic Extension of Business Process Modeling Notation

Semantic Mediation: This layer features the functionality to apply semantic bridges in order to get polymorph information entities that enable seamless information flow across differently conceptualized domains. Furthermore, by exploiting the before described components a matching functionality is provided that matches corresponding information entities that are related to the same meaning but are expressed in different representations. Moreover, visualization of semantic bridges is provided in order to integrate the mediated information flow in the business process modeling notation. Finally, according to the above described approach of process-oriented evolution of domain information models and semantic bridges (cf. Section 5.5.1), functionality is provided to specify requirements for the advancement of semantic bridges such as a further mapping between as corresponding identified concepts or properties within different organizational domains.

Semantic Pool: The Semantic pool lies vertically to the above described layers and provides support functionality for semantic annotation and mediation. On the one hand, it provides a repository to handle and manage domain ontologies to be used during annotation of information flow. On the other hand, the analog functionality is provided to manage the used semantic bridges including import, export, create and versioning operations.

Semantic Web Framework: Finally, the vertically located Semantic Web framework provides the required generic semantic technologies. These includes firstly an ontology reasoner to process the domain ontologies and secondly an integrated rule-engine to perform the semantic bridges, which are based on ontology mapping rules.

5.5.3 Related Work

Several approaches for integrating semantic technologies into business process modeling exist. The basic idea is to utilize formal semantic languages to alleviate the so called Business-IT-gap originating from the different perspectives of business requirements on the one hand side and existing IT systems and resources on the other hand side. In [196] a comprehensive introduction and framework for semantic business process modeling is provided. However, due to the relative novelty of the research field, the available approaches are mostly academic-driven. Nevertheless, some early industry adoptions exist such as SemTalk [197], which is based on the Microsoft Visio [198] modeling and visualization tool. However, this approach does not address the problem of semantic interoperability and works in a single or homogeneous ontology environment.

The most prominent academic example is the European integrated research project SUPER (Semantic Utilized for Process Management within and between Enterprises) [199]. The SUPER project defines a semantic business process management cycle [200] including semantic business process (SBP) analysis, SBP modeling, SBP implementation and SBP execution. Thereby, each phase is based on a so called ontological foundation, i.e. artifacts (e.g. events, activities or data objects) which are created or processed during business process management are consequently enriched with ontology concepts.

However, in contrast to the presented approach of semantic mediation in this work, the SUPER project relies on a single central ontology called business process modeling ontology (BPMO). Thereby, the approach and framework of the Web Service modeling Ontology (WSMO) is integrated (cf. Section 3.4.2), whereas e.g. business process activities are specified via goals, which later can be used to infer matching Web services that provide the designated functionality. Although these approaches are related to this work their focus is on ontology-based annotation of process steps, in order to improve process management and search in process repositories in a homogenous ontology environment. In contrast the research presented in this methodology step aims at mediating between different information models in cross-organizational design of BPM information flow and thus focuses on heterogeneous ontology environments. The SUPER project follows a more general direction towards semantic technology driven alignment between business and IT perspectives focusing on a mediation between these two spheres, whereas the in this work addressed issue of semantic interoperability in cross-organizational scenarios is not focused. The scope of the SUPER project is illustrated in the following Figure 5-8:

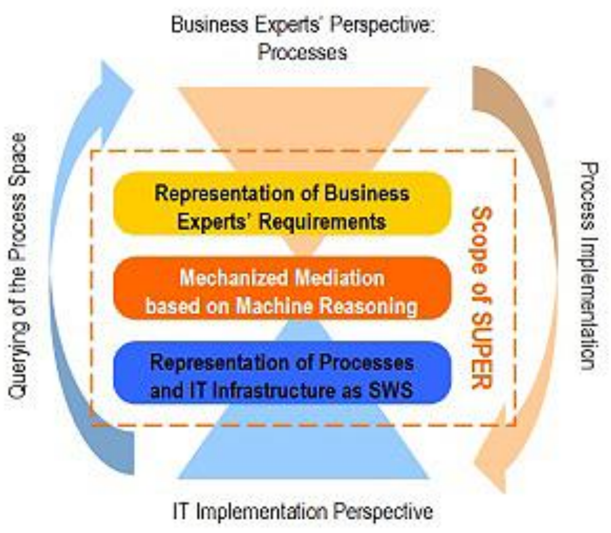


Figure 5-8 Mediation between Business and IT Perspective [199]

5.6 Semantic Bridge Definition

This section describes the semantic mediation methodology step of semantic bridge definition and the existing work in terms of methods and tools which can be applied to perform this methodology step.

5.6.1 Goals and Tasks

Having already identified process-oriented requirements for semantic bridges during mediated business process modeling, this methodology step covers their comprehensive identification and specification. Semantic bridges are the basis for semantic mediation between the different representations of semantically corresponding concepts in different domains. A detailed elaboration of the basic idea and concepts for ontology mapping as the underlying technology for semantic bridges has already been provided in Section 3.6.2. Taken into account the analysis there and the dedicated actor model for this methodology, domain information model experts from need to collaborate to ensure the required cross-domain knowledge and target the two identified tasks (cf. Section 3.6.2):

- Mapping Discovery – Identify the individual mappings between corresponding concepts, which are the continuing parts of a semantic bridge.
- Mapping Representation – Express the semantic bridge as a set of description logic based rules as defined in Section 4.8.

Similar to the development of domain ontologies, a hybrid so called top-down/bottom-up approach is followed for identifying the individual mappings. Therefore, in order to discover potential mappings one way is to start from what is already there in terms of analyzing the two ontologies to be semantically mapped and to detect correspondences between similar concepts. On the other hand a starting point can be the top-down identified requirements for semantic bridges derived during mediated business process modeling as described in the previous Section 5.5. The following Figure 5-9 illustrates these two complementary ways:

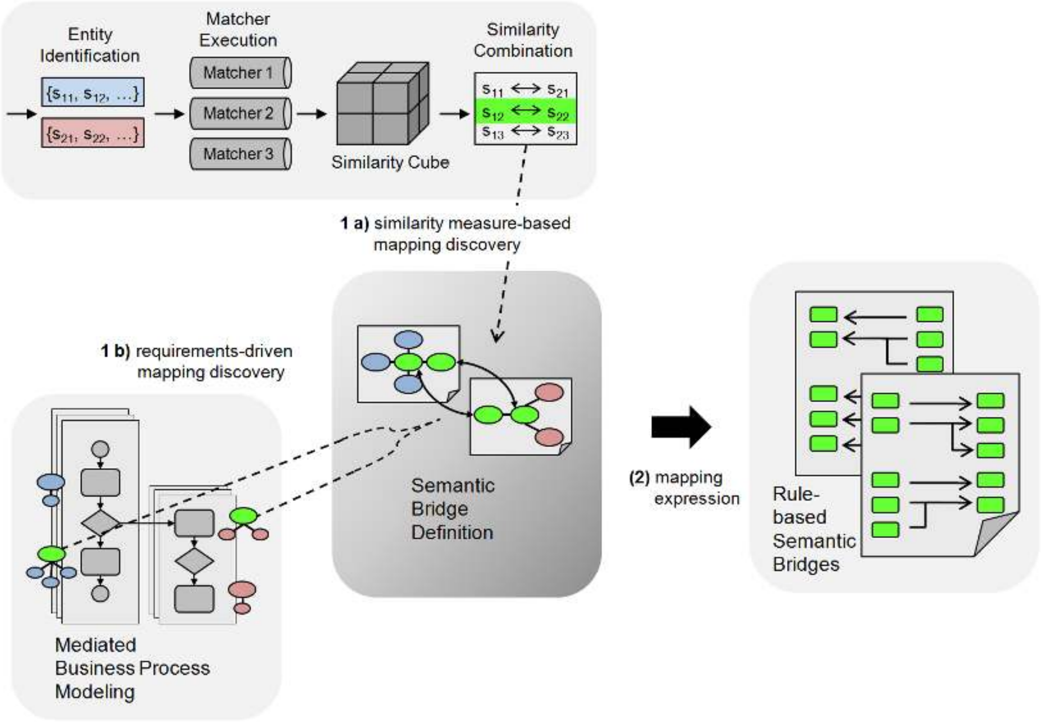


Figure 5-9 Big Picture Semantic Bridge Definition [201]

The (semi-)automatic mapping discovery (cf. 1a) in Figure 5-9) can be based on multiple e.g. lexical or structural matching algorithms calculating a similarity measure between each two concepts from different ontologies. Such a similarity measure-based mapping discovery is exemplarily described in [201]. As manual mapping discovery can be a very complex and a time-consuming task, the application of tool support for automatic or semi-automatic mapping discovery is very useful. As already discussed in Section 3.6.2 encouraging results are obtained, however this problem is by no means solved and automatically obtained results are not yet good enough in terms of recall and precision [202]. It has turned out that most approaches still require human intervention to generate sufficient results and thus need to be considered as semi-automatic. Indeed, for finding the correspondences between concepts, it is necessary to understand their meaning. Besides the represented meaning described by model-theoretic semantics, the ultimate meaning of concepts is only in the head of the people who developed those concepts and accordingly the final matching decision can only be performed by them [203].

On the other hand corresponding concepts from different domain ontologies are identified during cross-organizational business process modeling in a requirements-driven manner (cf. 1b) in Figure 5-9). This approach ensures as well that semantic bridges do not need to be exhaustively engineered but just to the extent of actual needs derived from mediated business process modeling as described in Section 5.5.

Finally, the identified mappings need to be expressed according to the developed approach for realizing semantic bridges by means of description logic rules to infer polymorph representations of concept instances (cf. 2) in Figure 5-9). In particular, the transformation of the different formalized semantic sub-graphs need to be defined as semi-automatic (1a) or requirements-driven mapping discovery approaches just determine which concepts are corresponding but not how they can be transformed into each other by means of declarative rule-based entity manipulation. The following demonstrates this task to be fulfilled by rule-based semantic bridge definition:

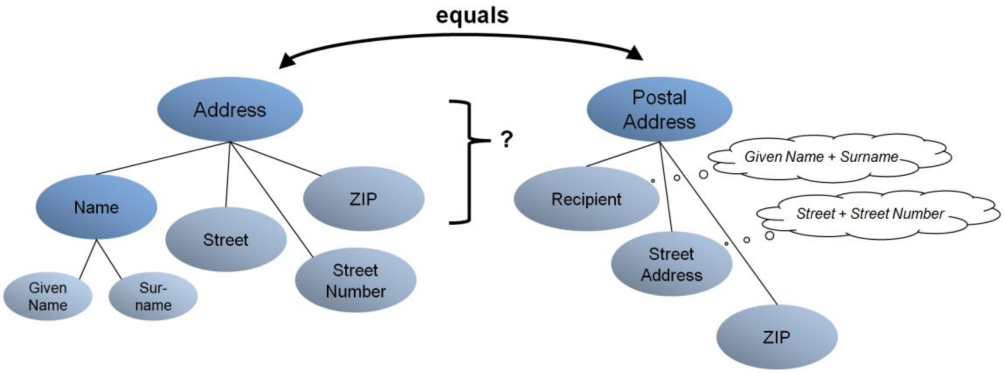


Figure 5-10 Required Entity Manipulation between Different Semantic Sub-Graphs

According to the chosen example already presented in Section 4.8.4 for example an object property *Name* of a concept *Address* is represented in terms of two sub-properties *Given Name* and *Surname*. In contrast in a semantically equal concept *PostalAddress* these two values are aggregated in the property *Recipient*. As the developed semantic bridge approach utilizes forward-chaining rules to express the transformation of the different semantic sub-graphs (cf. Section 4.8) semantic bridges from the first ontology to the second and vice versa have to be defined in order to obtain a bidirectional mapping as illustrated at 2) in Figure 5-9. Since the definition of such declarative mapping rules is a complex and time-consuming process, the application of graphical tool support to assists the domain ontology experts with this task is reasonable as presented in the next section.

5.6.2 Existing Work

As having outlined above and discussed in Section 3.6.2 ontology mapping discovery is an active research field and several heuristic-based and machine-learning approaches for (semi-) automatic mapping discovery have been proposed. To some extent they are similar to approaches for mapping of database schemas and XML structures, since they also use lexical and structural measures to determine correspondences [124]. But since ontology languages allow for more expressiveness, these approaches furthermore exploit features in ontology definitions in order to determine ontology mappings [124]:

- concept names and additional natural-language descriptions
- class hierarchy (subclass–superclass relationships)
- class definitions (equality relationships, defined classes, unions)
- property definitions (domains, ranges, restrictions)
- instances of classes

An introduction and overview of existing approaches is given in [204]. Furthermore, a comprehensive survey of available tools for semi-automatic mapping discovery from a user perspective is provided in [205].

As described above the second task in semantic bridge definition is the expression of ontology mappings. Several general approaches namely utilizing views and queries, mapping ontologies and bridging axioms have been discussed in Section 3.6.2. As the requirements set by the concept of semantic mediation between domain ontologies can be most effectively addressed by rule-based bridging axioms (cf. Section 4.8.5), the instantiation of this mechanism requires a concrete declarative rule language. In particular, the developed semantic bridge approach can be instantiated by means of forward-chaining rules expressed in the Semantic Web Rule Language (SWRL).

SWRL can be used not only to reason about OWL individuals and to infer new knowledge (cf. Section 3.3.2), but also to define entity manipulations describing mappings between ontologies. If an SWRL rule contains entities of a source ontology in its antecedent (so called body of the rule) and entities of a target ontology in its consequent (so called head of the rule), then a rule can be interpreted as a mapping rule. Consequently, upon its execution instances of a source ontology will be transformed into polymorph instances, which belong to two classes and have properties of both source and target ontologies (cf. Section 4.8.4).

A major benefit of applying SWRL rules is the opportunity to directly integrate the rule execution into the ontology inference process. As the goal of SWRL is to be a rule language for the Semantic Web [206], it is well integrated with the major ontology language standard OWL (cf. Section 3.3.2). Hence, if ontology mappings are expressed with rules, then the execution of mappings, i.e. transformation of instances from source ontology into target ontology can be performed with inference engines already in use for OWL-based ontology reasoning. Furthermore, SWRL constructs contain a set of predefined so called built-in functions, which enable highly expressive transformations and mappings e.g. performing mathematical operations, operations with strings, lists, etc. [207].

An additional benefit of using SWRL rules lies in the fact that it is relatively easy and intuitive for users to create rules for example by using a graphical editor. While rule users can express declaratively and visually what the result should be, it is left to the rule inferencing engine to process the desired result. One example to model SWRL rule-based ontology mappings is the graphical editor Snoggle [208]. Snoggle provides means to graphically create rule-based mappings in terms of a flexible set of functionality that allows to support various kinds of rule-based mapping approaches. As well the in this work developed approach for semantic bridges - exploiting in particularly the ontology feature of polymorphism - can be addressed with the Snoggle editor. Snoggle provides a canvas that displays entities from a source ontology and their relating entities from the target ontology. Correspondences between the entities of the two ontologies are visualised as arrows from the left region of the canvas to the right one as illustrated in following Figure 5-11:

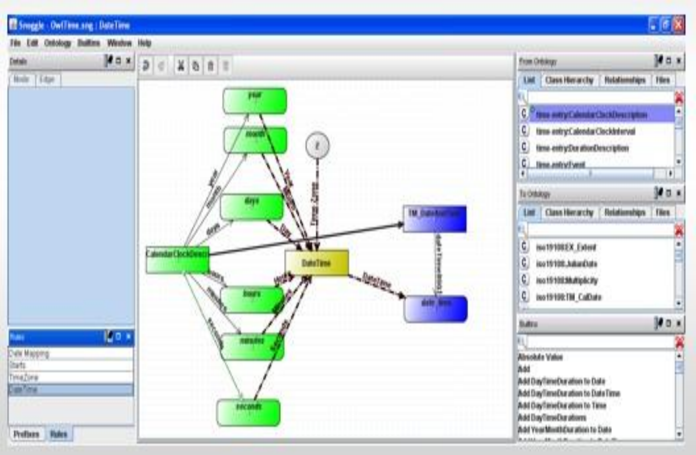


Figure 5-11 Graphical representation of a mapping rule in Snoggle [209]

Consequently, it has to be taken into account that a remarkable amount of existing work is available in terms of methods and tools for both fields: namely similarity-based mapping discovery and rule-based expression of semantic bridges. Therefore, it can be stated the no further functionality besides the already described requirements-driven mapping discovery (cf. Section 5.5) is required to enable the step of semantic bridge definition of the semantic mediation methodology.

5.7 Semantic Bridge Testing

This section describes the semantic mediation methodology step of semantic bridge testing and the required functional architecture of a tool to perform this methodology step. The concrete realization of this functionality is then further described in context of the developed semantic mediation toolkit in Section 6.3.

5.7.1 Goals and Tasks

As discussed in the previous section the development of semantic bridges is a complex and error-prone process. Moreover, the roles of semantic bridge developers in terms of domain information model experts on the one hand and semantic bridge users in terms of business process experts and service composers on the other hand are carried out by different actors. They even might be mutually unknown in cross-organizational SOA landscapes. Therefore, the consideration of trust in the quality of the developed semantic bridges is an essential factor. It can be shown that in principle no formal verification of the correctness of semantic bridges can be provided. Finally, semantic mappings between information entities representing the same meaning can only be determined by a human interpreter (cf. Section 4.5.2).

Therefore, the third step of the semantic mediation methodology focuses on the evaluation and particularly on the testing of semantic bridges. Focus should be put on how to adapt well established concepts from software testing as well as concepts for testing of ontologies, rule bases and XSL transformations to the systematic testing of semantic bridges and the underlying ontology mappings.

In this sense the evaluation of correctness can be understood in the way it is used in software testing: as the degree to which the system performs its intended function [210]. Applied to semantic bridges, evaluation of correctness can be interpreted as the degree to which the underlying ontology mapping performs transformations of ontology instances into polymorph representations as intended. Consequently, in order to quantify the evaluation, a measure for this degree should be provided as well as a measure for the coverage of a particular test with regard to the overall scope of a semantic bridge. The latter point is particularly relevant as in principle it is impossible to test a program exhaustively. This is due to the fact that testing cannot take into account all potential and partly unknown test inputs, execution conditions and their combinations [211]. That is why during testing consequently only a part of all possible test cases can be taken into account. Furthermore, in order to not only support black-box testing[2] but

---

[2] In the black box approach, a tester treats software-under-test as a black box, which means that he has no information about the structure of the program and no program code [210].

also certain aspects of so called white-box testing[3], the concrete identification and localization of defects in semantic bridges should be supported.

One fundamental concept, which can be derived from related research fields for testing as discussed above, is the method of testing with a predefined set of test cases. The basic idea is that during test execution a functional component is executed under certain conditions with a particular input. In order to determine if the test was successful or failed, a tester has to define the expected output of the test and compare it with the actual output produced during test execution. All this information is included in so called test cases. In particular a test case consists of the following parts [210]:

- Test inputs are data, which is passed functional components under test as input and which comes from external sources.
- Execution conditions are conditions, under which tested functional components should be executed.
- Expected outputs are the specified outputs that are supposed to be returned by the functional component under test.

Transferred to testing of semantic bridges this requires the following tasks: On the one hand for domain information model experts, who develop semantic bridges and on the other hand for semantic bridge users, i.e. business process experts and service composers, who have to trust in the quality and correctness of ontology mappings they use.

- Define semantic bridge test cases – Define exemplary instances of concepts from the source ontology as test inputs and expected polymorph instances as expected test outputs expressed as well according to concepts from the target ontology.
- Perform semantic bridge test cases – Provide the execution conditions in terms of source ontology, target ontology and an execution framework for the developed semantic bridge between them and run the test cases.
- Evaluate semantic bridge test cases – Provide an evaluation of the correctness regarding the performed test case based on a gap analysis between intended results defined in the test cases and actual results received after semantic bridge execution.

Accordingly, the basic idea of semantic bridge testing can be illustrated in the following diagram:

---

[3] In the white box testing approach, the inner structure of software-under-test is considered during the development of test cases. Hence, program code should be available to a tester [210].

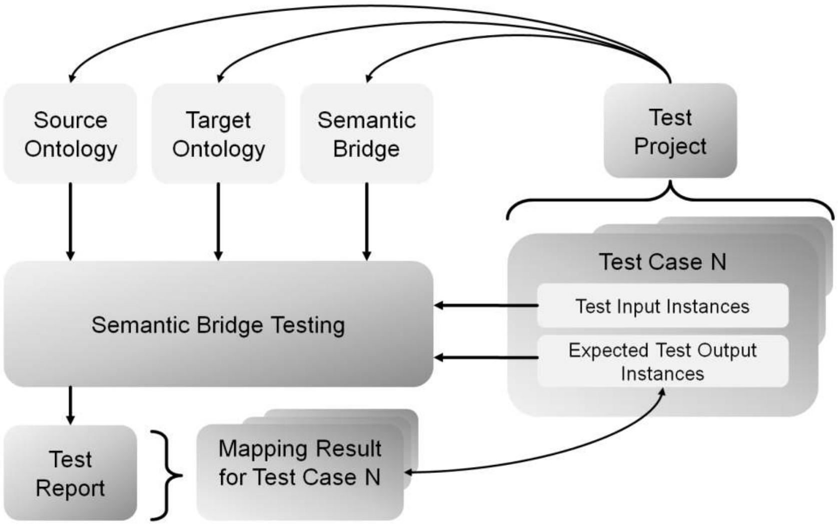


Figure 5-12 Basic Idea of Semantic Bridge Testing

The main inputs for semantic bridge testing are source and target ontologies and the semantic bridge between them. They are referenced in a test project which further consists of several test cases, in order to achieve a high coverage of the ontology mapping rules within the semantic bridge. After test execution the test report evaluates the overall test coverage and provides the gap analysis between the expected test output instances from the test cases with the actual individual mapping results.

Even though first tools e.g. for modeling rule-based ontology mappings such as Snoggle (cf. Section 5.6) are available, the research field is relatively new and thus existing tool support is very limited. Several approaches for testing in related areas have been investigated including ontology testing, testing of rule bases, testing and debugging of XSLT stylesheets and finally testing of ontology mappings (cf. Section 5.7.3). However, until now no dedicated testing methods and tools suitable for rule-based ontology mappings as exploited in the developed semantic bridge approach have been developed. Accordingly, the following section discusses a possible functional architecture of a tool for testing semantic bridges, whereas the concrete realization of the functionality is then further described in context of the developed semantic mediation toolkit in Section 6.3.

5.7.2 Functional Architecture

The main architecture style required for providing the functionality discussed above should follow the Model-View-Controller (MVC) [212] architecture style, in order to support the interactive process of semantic bridge testing. According to the MVC architecture style three types of functional components can be mapped to the requirements of semantic bridge testing:

- Models, which maintain the core information assets including internal representations of source and target ontology, semantic bridges, test projects and contained test cases as well as the test report to be generated.
- Views, providing the GUI enabling the domain information model expert and modeller of business processes or service compositions to define the test cases and investigate the rule-based ontology mappings under test. Thereby, different perspectives can be provided containing an interactive GUI for creating use cases,

testing and debugging semantic bridges and a static report based on the maintained models.

- Controller, which controls the interactions of a user and provides the business logic, i.e. the specific functionality for semantic bridge testing based on the maintained models.

Accordingly, the following Figure 5-13 illustrates the basic logical components of the functional architecture for semantic bridge testing including the functionalities required for the developed semantic bridge approach based on ontology languages and exploitation of declarative rules:

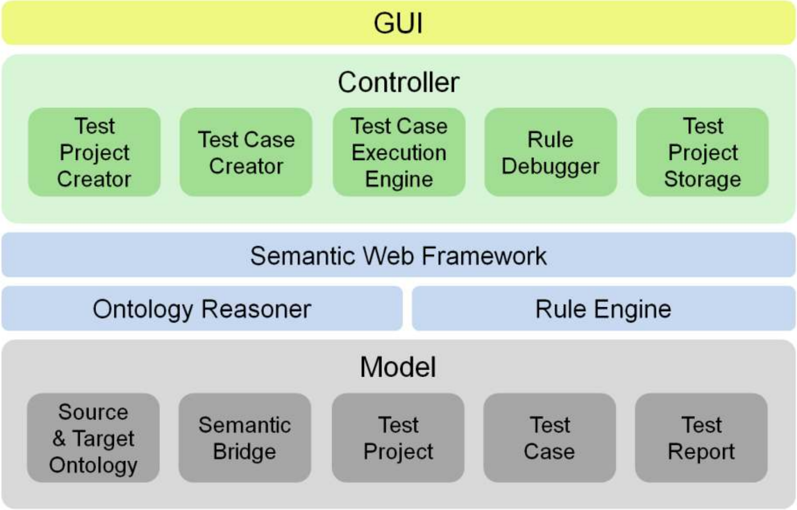


Figure 5-13 Functional Architecture of Semantic Bridge Testing Tool

The main functional components deal with the lifecycle of test cases and particularly their execution and evaluation in terms of a test report. Upon execution of semantic bridge test cases, ontology mapping rules are executed, as it has been described in Section 4.8.4 about the operation of semantic bridges. After semantic bridge test cases have been executed and mapping results have been created, they have to be compared with the expected test output instances. Thereby, the comparison should be performed according to the following qualities:

- If all statements[4] describing expected test output instances are also contained in the ontology mapping result or can be inferred in the ontology mapping result, the test case should be considered as *succeeded*.
- If some of the statements describing expected test output instances are not contained in the ontology mapping result and cannot be inferred as well, the test case should be considered as *failed*.

Furthermore, consistency checks should be performed. The semantic bridges are tested not only with respect to the expected test outputs, but also with respect to the involved ontology definitions as ontology mapping rules can also produce instances that can be inconsistent with the definition of source and target ontologies. Hence, it is reasonable to use consistency checks which are based e.g. on cardinality constraints, disjointness constraints, etc. in order to evaluate the overall correctness of semantic bridge execution or to detect defects in their underlying

---

[4] triple relations between subject, property and object in instance descriptions (cf. Section 3.3.1)

ontology mappings. However, consistency check results should not affect whether a particular test case succeeds or fails, since success or failure is determined only on the basis of comparison of expected test output instances with actual ontology mapping results. Rather, if a test case has succeeded, but the mapping result is not consistent with the source or target ontology definitions, then the test report should contain a warning about inconsistency. Accordingly, in the test report the evaluation can be quantified in terms of listing how many test cases have been succeeded or failed as well as informing about the number of consistency issues.

Moreover, in order to quantify the overall quality or expressiveness of the test project, the coverage of all test cases regarding the complete rule set of a semantic bridge should be calculated. This approach can be transferred from the context of testing rule bases as described in [213]. Consequently, the overall test case coverage of a semantic bridge can be calculated as follows:

$$Overall\ test\ case\ coverage = \frac{Number\ of\ mapping\ rules\ covered\ by\ all\ test\ cases}{Overall\ number\ of\ mapping\ rules\ of\ semantic\ bridge}$$

Hence a test report can be generated that contains the above discussed information. If any test case has failed, then it can serve as an indication that the semantic bridge is not completely correct and most probably contains defects. However, it is important to note that one the other hand if all the test cases in test project succeed, it neither proves that the semantic bridge is completely correct nor that is contains no defects. Nevertheless, a set of test cases, which have been well chosen and defined and which have succeeded during test case execution including a high degree of overall test case coverage, can demonstrate that a semantic bridge performs ontology instance transformations the way it is expected and hence it can increase trust in the tested semantic bridges.

5.7.3 Related Work

Even though first tools e.g. for modeling rule-based ontology mappings such as Snoggle (cf. Section 5.6) are available the research field is relatively new. Several approaches for testing in related areas have been investigated including testing ontology testing (e.g. [214], [215], [216], [217]), testing of rule bases (e.g. [218], [215]), testing and debugging of XSL transformations (e.g. [219], [220]) and finally testing of ontology mappings (e.g. [221]). Key concepts such as the test case-driven approach, white-box and black box testing as well as test coverage quantification could be transferred to semantic bridge testing. However, the analysis has turned out that until now no dedicated testing methods and tools for rule-based ontology mappings as required in the developed semantic bridge approach are available. Accordingly, the above section has discussed a possible functional architecture, in order to enable the methodology step of semantic bridge testing. The concrete realization of this functionality is then further described in context of the developed semantic mediation toolkit in Section 6.3.

5.8 Semantic Service Enrichment

This section describes the semantic mediation methodology step of semantic service enrichment and the existing work in terms of methods and tools which can be applied to perform this methodology step.

5.8.1 Goals and Tasks

After having provided the foundation for addressing semantic interoperability on a higher abstraction level by means of ontology-based domain ontologies and quality assessed semantic bridges between them the next step is to lift the service descriptions as well on the higher conceptual abstraction level. In order to leverage the developed semantic bridges, the existing Web services of an SOA landscape need to be enriched with the additional semantic layer. As already discussed in Section 3.4, the basic idea of Semantic Web services is to wrap existing WSDL-based Web services and XML Schema based message formats with explicit semantics in terms of concepts from domain ontologies. Combined with specific upper ontologies for Web services (cf. Section 3.4.2), which facilitate the machine interpretation of generic service capabilities, so called Semantic Web services can be realized. The following Figure 5-14 illustrates this basic idea of Semantic Web service enrichment:

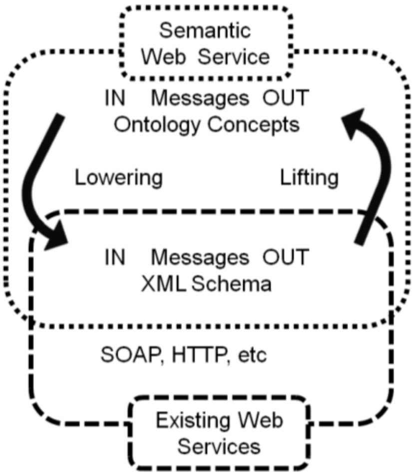


Figure 5-14 Basic Idea of Semantic Web Service Enrichment

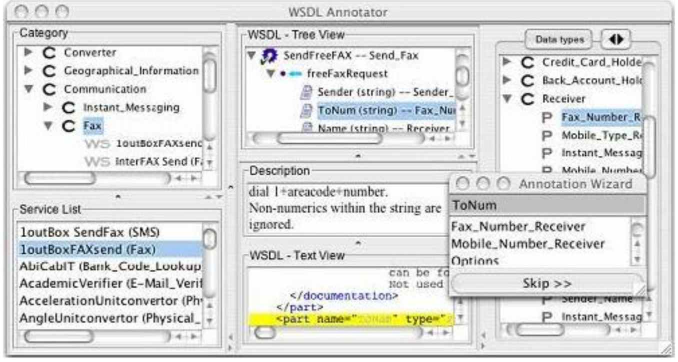
Accordingly, the Semantic Web Services are an extension of existing Web Services technologies and standards. Therefore, the service developer, responsible for developing Web services that provide the functionality required by the individual business process activities, also provides the semantic service enrichment. In particular, this includes the following tasks:

- Annotate service with domain concepts – Select concepts from the domain ontology to describe service input and output message parameters.
- Define lowering – Provide a translation for message parameter instances expressed according to domain ontology concepts to input parameter instances expressed according to XML Schema used by the underlying traditional Web service.
- Define lifting – Provide the reverse translation, i.e. from XML Schema-based Web service outputs to instances expressed according to domain ontology concepts used to describe the output of the Semantic Web service.
- Aggregate artifacts to formal Semantic Web service description – Provide annotations, lifting and lowering definitions in a formal service ontology language defined to express Semantic Web service, e.g. OWL-S (cf. Section 3.4.2)
- Ensure Semantic Web service interpretation and execution – Ensure that appropriate service execution engines are available in the targeted SOA landscape that enable the interpretation and execution of the developed Semantic Web services

It can be noticed that in the above outlined description of necessary tasks for the methodology step of Semantic service enrichment no point is dedicated to the specification of formal descriptions of service goals, preconditions and postconditions as discussed in context of Semantic Web service descriptions in Section 3.4.2. However, as already stated in the evaluation of different Semantic Web service approaches (cf. Section 3.4.3) the approach followed in this work does not cover goal-based plan creation. The approach developed in this work intentionally leaves the planning task (i.e. which services to include at which part into the composition) to the business domain process expert, in order to remain compliant to the well-established SOA life cycle. A further reason is to not overload and mix the approach with concepts such as goal-based planning which do not purely focus on the challenge of semantic interoperability in SOA addressed in this work but rather aim for higher levels of automation and dynamics particularly in service composition leaving aside major real world constraints regarding the heterogeneity of information models (cf. Section 3.4.3).

5.8.2 Existing Work

As having outlined above and discussed in Section 3.4.2 semantic service enrichment is an active research field and several concepts, W3C standards and corresponding tools already exist. This existing work covers both fields discussed above, tool-support for semantic annotation of Web service descriptions during design time and corresponding Semantic Web service engines which are able to interpret the semantic description and execute the underlying services during runtime. This includes for example the tool Assam WSDL Annotator [222]. The basic idea of the tool is to generate an OWL-S service description from an existing WSDL file. The following Figure 5-15 shows the graphical user interface of the Assam WSDL Annotator:



dial 1+areacode+number. Non-numerics within the string are ignored.

```
                    can be fo
                    Not used
      </documentation>
    </part>
    <part name="ToNum" type="
```

Figure 5-15 ASSAM WSDL to OWL-S Annotator GUI [222]

Another tool called WSDL2OWLS converts WSDL-based Web service descriptions to OWL-S descriptions, whereas analog to the Assam tool the concrete lifting and lowering XSL transformations have to be provided externally. The tool is integrated into a Semantic Web service framework developed in the MINDSWAP project at the University of Maryland called OWL-S API [223]. The framework also supports the interpretation and execution of OWL-S described Semantic Web services. A good overview of existing approaches and tools has been provided in context the semantic interoperability work package of the European research project QualiPSo in [224].

Taking into account the amount of existing work, it can be stated that no further general functionality is required to enable the methodology step of semantic service enrichment. However, as turned out during the usage of these tools, one technical enhancement for Semantic Web service execution is required that addresses the challenge to deal with the different meta-data models. Therefore, an extension of the applied OWL-S API for OWL-S execution including lifting and grounding has to be made, which is discussed in context of the realization of the semantic mediation toolkit in particular with regard to mediated process execution in Section 5.10.

5.9 Mediated Service Composition

This section describes the semantic mediation methodology step of mediated service composition and the required functional architecture of a tool to perform this methodology step. The concrete realization of this functionality is then further described in context of the developed semantic mediation toolkit in Section 6.3.

5.9.1 Goals and Tasks

The composition of services is an integral part of the SOA life-cycle. From the perspective of semantic mediation the main focus in this methodology step lies in the information flow design by a semantic service composer who combines the before provided Semantic Web services to a mediated service composition. The goal of the composition task is to instantiate the previously developed cross-organizational business process model by defining a concrete so called Semantic Web service orchestration plan that can be executed in a process engine. In order to achieve a high level of automation in this design process, the semantically enriched service descriptions can be exploited for service selection based on semantic match making between service input and output parameters and seamless information flow design across organizational borders. This can be achieved by performing a reasoning over semantically described relationships (such as inheritance or equality between concepts), thus enabling a composition tool to support the design of interaction patterns by issuing recommendations for suitable assignments between output and input parameters of different Web services.

Taking into account that the input and output parameters of different services are described with ontology concepts from heterogeneous information models the before provided semantic bridges are used to mediate between the different domain conceptualizations. Thereby, the semantic integration can be handled transparently for the service composer and combined with the match making functionality less manual integration efforts are required to achieve semantic interoperability. Consequently, the main benefit of this approach is to overcome technical data flow and transformation coding and thus integration efforts can be reduced significantly. The following Figure 5-16 illustrates the main conceptual artifacts of information flow design in mediated service composition and how they interrelate:

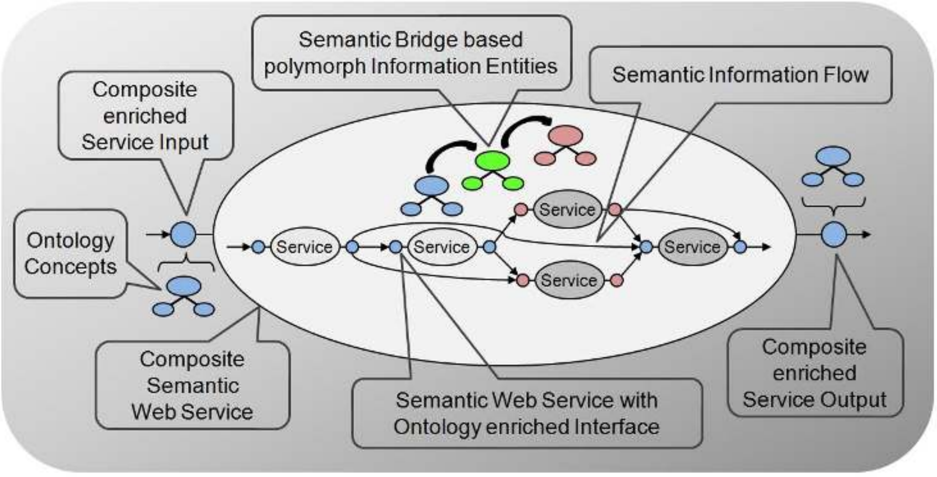


Figure 5-16 Basic Idea of Mediated Service Composition

In order to design the information flow in terms of a mediated service composition, the service composer has to accomplish the following tasks supported by dedicated tool support:

- Firstly, joint semantically enriched input parameters of the whole composition have to be declared. This can be done by means of domain ontology concepts describing them. Furthermore, their assignment to corresponding Web service inputs within the composition needs to be defined. Thus, the whole service composition itself can be regarded as a Semantic Web service with a composite service input.
- In order to design the information flow between the involved Semantic Web services of the composition, the individual service output parameters or certain parts of them need to be assigned to input parameters of services that are succeeding within the composition. Therefore, an automatic matching mechanism can be performed that retrieves a semantically sound information flow based on a so called pull-mode. This means that for each service input (or input part) the potential sources of the information flow targeting this input are analyzed with regard to their semantic suitability. Transparently, semantic bridges are integrated into this matching process, in order to identify matching service parameters across domain borders. The matching candidates are presented to the service composer as an information flow recommendation.
- Based on the semantically sound information flow recommendations the service composer selects appropriate parameter assignments that comply with the desired business process logic. According to the selection the actual information flow needs to be expressed in order to be integrated in the targeted executable orchestration plan. Therefore, the description logic based meta-model for information description has to be reflected which in turn implies that the information flow should be as well expressed in terms of description logic.
- After the information flow within the service composition is defined, analogically to the composite input joint output parameters of the composition have to be declared. Furthermore, their origins from corresponding Semantic Web service outputs within the composition need to be defined.
- Finally, the defined composition needs to be exported to an executable Web service orchestration plan. This part is covered in more detail in the previous section about mediated process execution.

Thereby, the main challenge of dealing with the description logic based meta-model for information description in contrast to the XML meta-data model used in traditional Web service composition can be tackled by consistently remaining on the additional semantic layer during service composition. This includes the usage of Semantic Web services provided from the semantic service enrichment methodology step. Also the semantic bridges exploited for the matching mechanism and as well the information flow definitions remain completely on the level of description logic based meta-models. Thus, no interference with the underlying structural-based XML meta-data model occurs and a highly automated information flow design can be ensured.

Even though various approaches for integrating semantic technologies into service composition exists (cf. Section 5.9.3) the presented approach for mediated service composition features new aspects such as remaining consistently on the added semantic layer and thus requires a dedicated solution as a proof-of-concept. Therefore, the following section discusses a possible functional architecture of a tool for mediated service composition as outlined above. A systematic requirement analysis in terms of use case diagrams describing the above discussed analysis in more detail has been performed in [225]. The concrete realization of this functionality is then further described in context of the developed semantic mediation toolkit.

5.9.2 Functional Architecture

In this section a functional architecture enabling the methodology step of mediated service composition is presented and explained. The central component is the Semantic Web service composition tool, which supports a service composer in modeling the control- and information flow by exploiting the mediation mechanism as described above. To highlight the feature that the composition is consistently performed on the added semantic layer, additionally the in this context involved components are integrated into the functional architecture. The following Figure 5-17 illustrates the main conceptual components of the functional architecture and how they relate to each other:

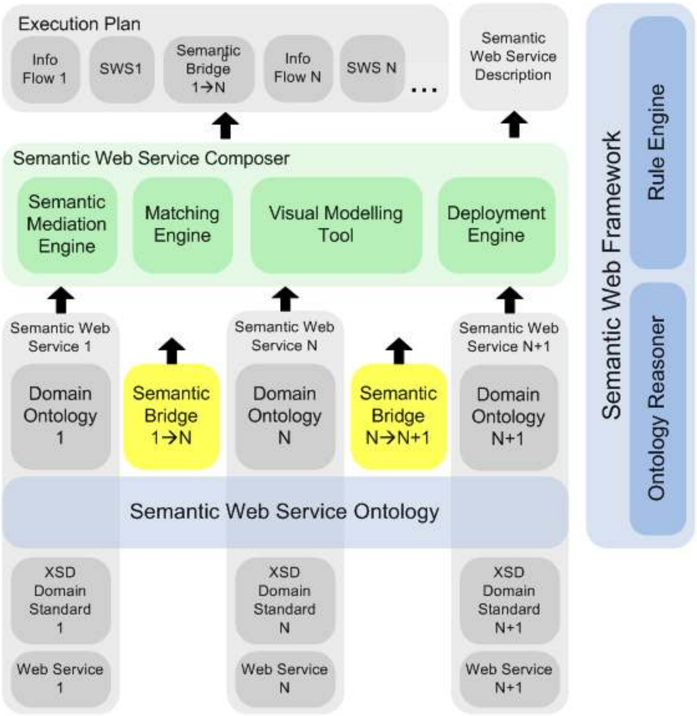


Figure 5-17 Functional Architecture of Mediated Service Composition

The Semantic Web services are based upon traditional WSDL-based Web services, which use XML Schema (XSD)-based domain standards to describe service input and output parameters as explained in the methodology step of semantic service enrichment (cf. Section 5.8). On top of these traditional Web services a Semantic Web service ontology combined with selected concepts from corresponding domain ontologies is utilized to semantically describe the input and output parameters of thus established Semantic Web service descriptions. Furthermore, semantic bridges between overlapping domain ontologies need to be provided, in order to enable semantic mediation between them. Finally, the Semantic Web service descriptions together with the semantic bridges build the input for the Semantic Web service composer.

The central component of the Semantic Web service composer is a visual modeling tool that enables the service composer to orchestrate services. It is based on a Semantic Web framework for handling the Semantic Web service descriptions and domain ontologies. Additionally, an ontology reasoner and a rule engine have to be integrated to support the following components: Firstly, the mediation between the domain ontologies has to be performed by a semantic bridging engine that applies the semantic bridges to the Semantic Web service parameters. Thus, the foundation for seamless design of information flow across heterogeneous Web services from different domains is established. In order to increase automation in this process, the matching engine reasons over available service parameters and recommends matching parameter parts that could be assigned to a certain Web service. According to the actual business process logic the service composer selects the corresponding assignments and additionally defines joint input and output parameters of the composed process by using the visual modeling tool. Based on the service composer’s input, the deployment engine then generates an execution plan of the composed process and deploys it into an execution engine. Finally, the created process is then accessible as a new Semantic Web service. The process execution is further discussed in the next step of the semantic mediation methodology. Moreover, the above presented functional architecture is concretized in Section 6.3 describing the developed semantic mediation toolkit by relating the conceptual components to concrete technologies and their implementation.

5.9.3 Related Work

In recent years the field of semantics-based service composition has been an active research field as discussed in the state-of-the-art analysis about Semantic Web services (cf. Section 3.4). However, the focus is put on services composed in homogenous ontology environments or on the integration of dynamic behavior during the execution of service compositions (cf. Section 3.4.3). In this context semantic mediation is only addressed during runtime, whereas during design time a homogeneous ontology environment is assumed. In the following an exemplary European research project is outlined, which shows the relation to this work in the field of Semantic Web technologies applied to SOA.

The SATINE project [226] has been located within European research activities for networked businesses and governments. It has develop a semantics-based interoperability infrastructure for the tourism industry, which provides tools and mechanisms for publishing, discovering, composing and invoking Web services through their semantics in peer-to-peer networks [227]. A detailed description of the SATINE infrastructure is given in [228]. To focus on the relevant part for comparison with this work, on the one hand, the mediation approach of the SATINE project between semantically heterogeneous service requests during runtime is outlined and on the other hand, the approach to Semantic Web service composition and the execution framework is presented.

Within SATINE ontologies describing the travel domain are applied for Web service annotation. With respect to an open environment SATINE supports mapping between different ontologies. It provides support for the basic steps involved in ontology mapping (cf. Section 3.6.2) by integrating the OWL mapping tool (OWLmt) [229] into the peer-to-peer infrastructure used for Web service discovery and interaction. OWLmt follows the approach of describing the mapping between two ontologies in terms of a dedicated ontology, which is called mapping schema. OWLmt includes a graphical mapping tool which generates instances of the mapping schema to express the mapping between a source and a target ontology. The tool does not provide any automatic mechanism for discovering similarities between corresponding constructs in the source and target ontology. The user has to identify corresponding concepts or properties and relate them graphically supported with specific mapping schema constructs. These constructs include similarity and subclass relations as well as transformation relations to overcome structural differences including one-to-many relations. Transformation relations are defined by specifying instances of concepts or properties in source queries and a plan for target instance construction. In order to enable transformation of values, XPath built-in functions are applied besides an integrated mechanism of user defined JavaScript transformation code to realize more expressive transformations. Consequently, the resulting mapping can then be executed by a specific engine to transform instances defined in the source ontology into new instances defined in the target ontology. Furthermore, mediator nodes within the Web service peer-to-peer infrastructure, similar to interceptors in RM-ODP (cf. Section 3.5.2), detect mismatching service requests and invocation messages and transform them into the appropriate ontology representation needed in a specific context. That enables service discovery and invocation across peers using different ontologies for describing Web services. In comparison with the approach applied in this work, which targets semantic mediation on the conceptual level, it can be stated that the utilized approach of OWLmt in the SATINE project can be located on the logical level as the core of the mapping mechanism are XPath built-in functions and JavaScript transformation codes.

With regard to Semantic Web service composition and execution SATINE supports the orchestration of Web services published in the peer-to-peer network to provide so called added-value services. The following Figure 5-18 SATINE Composition Phases and Tools illustrates the SATINE composition framework, its phases and corresponding tools:

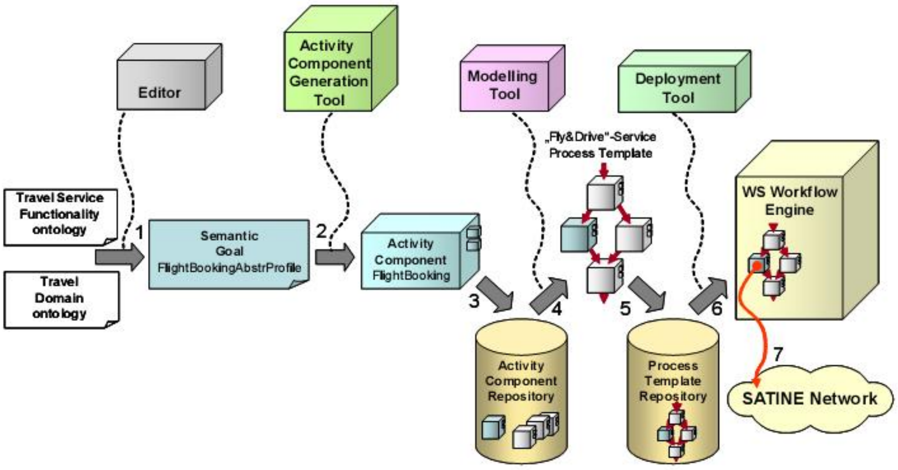


Figure 5-18 SATINE Composition Phases and Tools [230]

The composition framework is based on the concept of activity components [227] which can be described as abstract functional building blocks representing a set of Web services that provide the same functionality. Choosing appropriate concepts from a domain ontology, an OWL-S profile can be defined (step 1). In particular, input and output parameters of the Web service are thus specified. The OWL-S profile is then used to generate an activity component (step 2), which also includes a WSDL view on the service. The WSDL description can be exploited for integration of the Web service into a process description in terms of the Business Process Execution Language (BPEL). Generated activity components are stored in a repository (step 3) to make them available for the composition modeling tool. The modeling tool supports the orchestration of various activity components by assisting in data flow design by means of assignment recommendations between service outputs and inputs of following services. The support for data flow design is based on semantic matching of service parameters. In contrast to the mediation approach of this work, which takes into account heterogeneous domain ontologies, SATINE requires that conceptualizations are homogeneous and shared on the design level. However, during the invocation of services the usage of heterogeneous ontologies is supported as mentioned before by utilizing the OWLmt mapping mechanism. Thus, during design time the single ontology approach is followed, whereas during runtime the multiple ontology approach is applied. In particular, the composition tool supports the matching of concepts that are related by inheritance. However, each ontology class is strictly mapped to one XML Schema type to enable the data flow expression in BPEL, which is based on the data model of XML Schema. Consequently, due to the shortcomings of XML Schema regarding the expression of inheritance, the matching mechanism cannot directly exploit such relations but a technical work-around need to be provided operating on the structural-based XML level. In exchange the composed process can be directly stored as an XML-based process template specified in BPEL in the process template repository (step 5). Furthermore, a configuration tool is used to specify non-functional parameters of activity components, which influence the runtime-selection of concrete services from the SATINE peer-to-peer network.

5.10 Meditated Process Execution

This section describes the goals and tasks of the semantic mediation methodology step of mediated process execution. Furthermore, a functional architecture for a dedicated semantic-based process execution engine including the presented mediation mechanism is presented and discussed in context of related work. The concrete realization of this functionality is then further described in Section 6.3 presenting the developed semantic mediation toolkit.

5.10.1 Goals and Tasks

It has been recognized that the success of Semantic Web technologies relies on the reuse and integration of existing Web standards. The most widely-used standard for the composition of Web services is BPEL (cf. Section 3.2.2). A considerable number of mature BPEL compliant process execution engines testify the broad industrial support for this standard, which provides a rich set of control and data flow constructs for defining and aligning the interactions between Web services in a business process. Consequently, the goal of the mediated process execution approach is to integrate the before composed Semantic Web service composition including the semantic mediation mechanism into an execution plan formalism in terms of the BPEL standard. Thus, the execution in available process engines can be ensured.

The main challenge on this regard is to find a suitable mapping between different abstraction levels: While at design time ontologies and description logic based rules are used for information representation, information flow and semantic mediation, BPEL execution engines make use of hierarchically structured XML Schema types, XPath and XSLT transformations (cf. Section 3.2.3). In order to face this challenge, the starting point is to exploit the RDF/XML serialization of ontologies (cf. Section 3.3.2) for data representation on the BPEL level. Furthermore, BPEL enhancements have to be developed to integrate semantic bridges and to support information flow specifications in terms of description logic based rules. These enhancements should be incorporated as external functions that can be plugged into BPEL engines using standardized extension mechanisms. Finally, as well the invocation of Semantic Web services within the mediated business process has to be covered. Thereby, the mapping of the additional semantic layer (as discussed in context of the methodology step of semantic service enrichment in Section 5.8) to the underlying structural-oriented XML layer, which is natively supported by BPEL, has to be considered. This requires on the one hand, to ensure a BPEL-conform Web service invocation mechanism while at the same time preserving the semantic soundness of the service interaction. The following Figure 5-19 illustrates the above discussed basic idea of mediated business process execution based on BPEL:

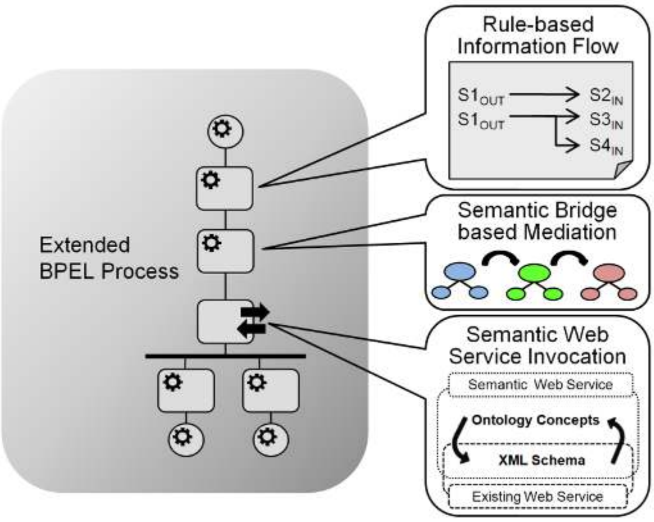


Figure 5-19 Basic Idea of Mediated Process Execution based on BPEL

In order to achieve the above presented approach, firstly a BPEL-conform execution plan including the extensions for rule-based information flow, semantic mediation and Semantic Web service invocation has to be generated. This generation has to take place in the design phase of mediated service composition (cf. Section 5.9). Secondly, components providing the Semantic Web technology-based functionalities need to be incorporated into a BPEL-based process integration middleware (BPEL process engine). This incorporation has requires to reflect the different meta-data models for representing the processed information models on the runtime level.

A systematic requirement analysis in terms of use case diagrams describing the above discussed basic idea in more detail has been performed in [225]. The following section discusses a possible functional architecture of such an extension of a BPEL process engine, whereas the concrete realization of this functionality is then again further described in context of the developed semantic mediation toolkit in Section 6.3.

5.10.2 Functional Architecture

This section presents and explains the functional architecture for an engine enabling the methodology step of mediated process execution. A BPEL-conform process execution engine is responsible for interpreting the BPEL execution plan. This functionality is bundled in the engines core component. Furthermore, the process engine is enhanced with additional functionality to perform the semantic extensions. The relation between the core BPEL functionality and the extensions for semantic mediation and corresponding components are illustrated in the following Figure 5-20:

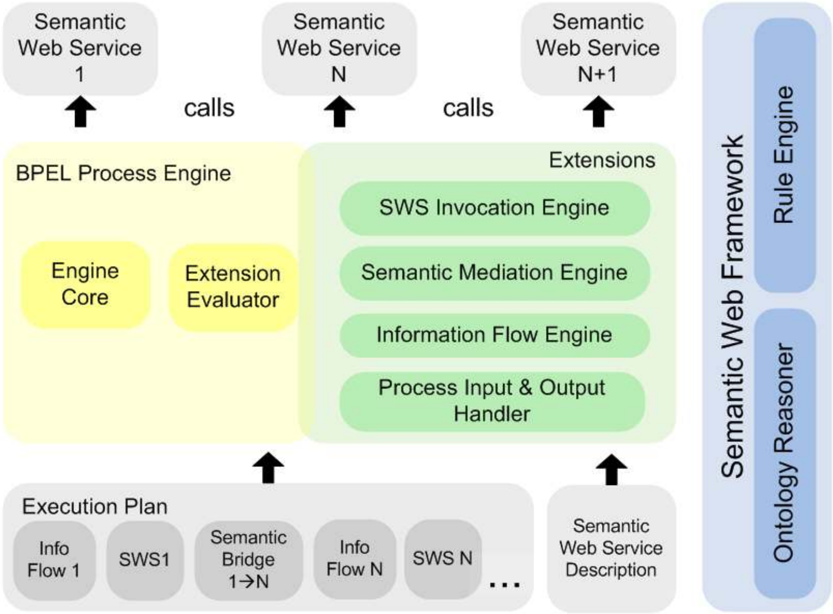


Figure 5-20 Functional Architecture of Mediated Process Execution

The extension evaluator is responsible for evaluating the invocation of additional functionality embedded in the BPEL execution plan. As the composed process is published as a Semantic Web service (SWS) the first extension handles the overall process input and outputs and the translation between the different data meta-models. This includes the lifting and lowering between XML instances of XML Schema types according to the WSDL interface of the BPEL process on the one hand and ontology concept instances as used for semantic mediation on the other hand.

Having lifted the process input data on the semantic level the information flow engine can interpret the description logic based rules that define the information flow. This includes the information flow between process input and service input parameters within the composition as well as between service output and other service input parameters or parts of them. Therefore, again the functionality of a Semantic Web framework has to be utilized, which provides ontology reasoning and rule inferencing as basic support services.

Having all inputs on hand, a Semantic Web service can be called by means of the SWS invocation engine. According to the lifting and grounding mechanism (cf. Section 5.8) the parameters are transformed from ontology-based individuals (ontology concept instances) to instances of XML Schema types and vice versa for the reply after the grounded WSDL-based Web service has been called.

To perform the semantic mediation expressed in the embedded semantic bridges between heterogeneous services, the semantic bridge engine performs the rule-based ontology mappings. As described in Section 4.8.4, the heterogeneous representations are mediated from ontology concepts describing the outputs resulting from the called Semantic Web service to the required input representations for the next Semantic Web service to be called. Again a Semantic Web framework is required to support the handling of domain ontologies and the rule-based mappings between them.

After the last Semantic Web service has been called and the information flow to the joint outputs has been forwarded, the execution engine passes these outputs as results to the caller of the Semantic Web service and the execution of the composition is completed.

5.10.3 Related Work

Integrating semantic extensions into service composition and its execution by means of process engines has been a vital research field in the last years. Accordingly, several approaches exist which target to integrate semantics-based enhancements into BPEL. Related work in this field has been already touched in the related work section of the previous methodology steps of meditated business process modeling (cf. Section 5.5.3) and mediated service composition (cf. Section 5.9.3).

In the related work section of the mediated process modeling step the SUPER research project has been exemplarily highlighted. Besides the modeling aspects including the development of upper ontologies for modeling semantic business processes (SBPMN), the SUPER project has defined semantic extensions of the BPEL specification (sBPEL and BPEL4SWS) and has developed a prototypical semantic BPEL Execution Engine (SBPELEE). The focus of these activities is put on the dynamic discovery and dynamic composition of services, which are based on the goal-oriented approach discussed and evaluated in Section 3.4.3. As already stated there, this approach is difficult to map to non-planning based composition and execution approaches as applied in the work of this thesis.

In context of the service composition step the SATINE research project has been exemplarily presented as related work. Besides the composition framework the SATINE toolkit also includes the export of an execution plan to a BPEL process engine. The designed orchestration plan of Semantic Web services including control and information flow is expressed in the XML-based BPEL language, in order to ensure compatibility to state-of-the-art BPEL execution engines. However the underlying assumption of the SATINE composition and execution approach has a different focus (cf. Section 5.9.3). While in SATINE a homogenous ontology environment during the composition and its BPEL-process representation is assumed, the anticipation of heterogeneous conceptualization of service descriptions in a heterogeneous SOA landscape is at the core of this work.

The concrete realization of this functionality enabling the execution of enhanced BPEL processes including the semantic mediation mechanism between heterogeneous services is then described in context of the developed semantic mediation toolkit in Section 6.3.

5.11 Summary and Reflection

This chapter has presented the developed methodology and functional architecture for semantic mediation in SOA. The principle idea has been to provide a connecting step between the developed theoretical concepts and its experimental confirmation by mapping the concept of semantic mediation to the concrete SOA life-cycle. Accordingly, the semantic mediation methodology has been aligned to the basic steps of the SOA life-cycle where mediation between heterogeneous information models is required.

The chapter has started by discussing relevant requirements for the semantic mediation methodology based on a refinement of the general conceptual requirements identified in Section 4.2. This has included the reflection of organizational boundaries in domain conceptualization or as well the shift of abstraction level for semantic integration. Moreover, the requirement regarding the reduction of complexity in semantic integration in terms of separation of concerns has been addressed in a dedicated domain actor model describing roles and responsibilities. A further refined requirement has been consistency with best practice SOA methodologies, which implies dedicated methodology steps for business process modeling and service composition and respective roles and responsibilities in the actor model. Furthermore, technological path dependency in SOA has been reflected by outlining the restrictions to certain technologies and standards of the W3C and OASIS standards body.

Further domain specific considerations with regard to the application field of SOA have been derived and integrated into the already mentioned domain actor model that specifies stakeholders relevant for the semantic mediation methodology with their roles, responsibilities and general skill profiles. The following actors could be identified: domain information model expert, business process expert, service developer, service composer and service or process consumer.

Subsequently, an overview of the individual steps of the semantic mediation methodology has been given to provide a better understanding in terms of a big picture. The semantic mediation methodology consists of the following seven steps:

- Domain Ontology Development
- Mediated Business Process Modeling
- Semantic Bridge Definition
- Semantic Bridge Testing
- Semantic Service Enrichment
- Mediated Service Composition
- Mediated Business Process Execution

The following sections then have presented the particular methodological steps in more detail with regard to their goals, the performed tasks within each step and the required functionalities. In order to prepare an experimental confirmation of key steps of the methodology by means of a prototypical semantic mediation toolkit, additionally a high-level view on the functional architecture for each methodology step has been derived. Therein, it has been distinguished between methodology steps which can be addressed sufficiently with existing work and respective tools and steps which cannot be mapped to available functionality. Accordingly, for the steps: domain ontology development, semantic bridge definition and semantic service enrichment existing work has been presented, whereas for the methodology steps: mediated business process modeling, semantic bridge testing, mediated service composition and mediated business process execution a dedicated functional architecture has been elaborated.

Based on the in this chapter developed methodology for semantic mediation and the respective functional architectures, the following Chapter 6 now presents the developed prototypical instantiation in terms of the semantic mediation toolkit.

Chapter 6

Realization of Semantic Mediation Toolkit

6.1 Overview

This chapter presents the realization of the developed semantic mediation toolkit for the purpose of providing an experimental proof-of-concept. The toolkit comprises instantiations of the functionality specified in the previous Chapter 5, which has presented the semantic mediation methodology and functional architectures of missing steps for its application. For the steps: domain ontology development, semantic bridge definition and semantic service enrichment existing work and corresponding tools can be directly reused. The here presented toolkit covers the newly developed prototypical tools for the methodology steps: mediated business process modeling, semantic bridge testing, mediated service composition and mediated business process execution.

Each tool is focused in a separate section following a consistent structure for the description of its prototypical realization: Firstly, the respective system requirements are briefly refined based on the goals of the particular methodology step and its enabling functional architecture. The following section covers the design and actual realization of the respective tool. The functional architectures presented in the previous chapter are refined to system architectures and the technical realization of its components and their interrelation are discussed. Then, a further section is devoted to the validation and verification of each developed prototype based on a cross-organizational business scenario, which is highlighted from different perspectives according to the respective focus of each prototype of the toolkit. The validation part analyses how the developed prototype meets the objectives as refined in the section about system requirements and the verification part deals with the question, whether the prototype performs the functionality correctly. An evaluation of the semantic mediation approach itself is given in an extra Chapter 7.

Finally, a brief overview of references to information about the usage and extension of the toolkit is provided, which has been published and made available as four separate open source projects. The chapter finishes with a conclusion and reflection on the realization of the semantic mediation toolkit.

6.2 Mediated Business Process Modeling Tool

This section discusses the realization of the mediated business process modeling tool as part of the semantic mediation toolkit. The section is structured according to the outline presented above (cf. Section 6.1) starting with a summary of the system requirements, over the description of the design and realization of the system architecture and its components to its application in terms of a scenario for the tool's validation and verification.

6.2.1 System Requirements

The main goals of the methodology step of mediated business process modeling together with its functional architecture discussed in Section 5.5 express the system requirements of the envisioned tool. In order to recall them and provide a consistent understanding for the following sections they are summarized as follows:

- Provide functionality for the design of information flow in business process models on a non-technical conceptual level in terms of ontology-based information models suitable for business process experts.
- Ensure a consistent link between the business or conceptual level (ontology-based information models) and the underlying technical information or data models and thus provide improved business IT alignment.
- Integrate the developed semantic mediation mechanism to enable seamless design of cross-organizational information flow, whereas semantic heterogeneities are handled transparently for business process experts.
- Provide functionality for business process-driven evolution and extension of existing information models and semantic bridges with corresponding collaboration support between business process and domain information model experts.

In the following the functional architecture consisting of the three horizontal layers: business process modeling, semantic annotation and semantic mediation combined with the vertical support layers: semantic pool and Semantic Web framework (cf. Section 5.5.2) is refined to the tool's system architecture. Additionally, each component is described in terms of its concrete technical instantiation.

6.2.2 Design and Realization

The prototypical implementation of the tool for mediated business process modeling has been performed according to the Agile Unified Process (AUP) [231], which is a methodology based on the Rational Unified Process (RUP) [232] framework. I.e. the development process is composed of several iterations, where each one has few distinguished phases. After each iteration, a set of artifacts is collected to systematically present the progress of the development. Critical (in terms of most difficult to implement and core components) have been designed first and then prototypically realized before further components have been addressed. The general architecture style is client-server based as this suits best the goal for distributed collaboration between multiple business process experts and domain information model experts in the targeted cross-organizational context. The following Figure 6-1 presents the designed system architecture:

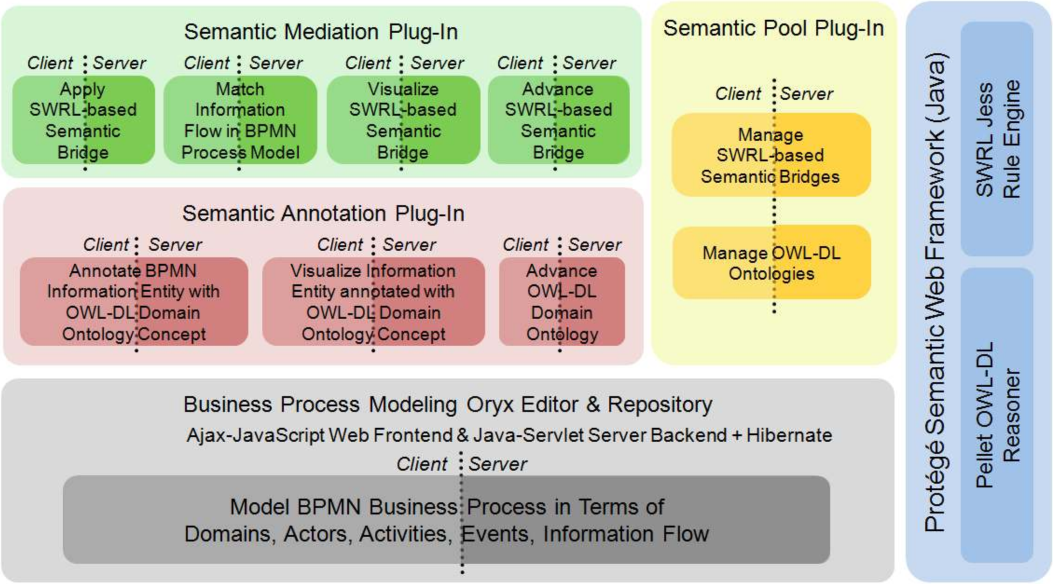


Figure 6-1 System Architecture of Mediated Business Process Modeling Tool

Business Process Modeling: The underlying layer is instantiated by the open-source based Oryx editor and repository for the basic business process modeling functionality. The Oryx editor has been chosen among a set of candidates in a systematic criteria-based evaluation carried out in [233]. The main criteria have been extensibility, support of standardized business process modeling notations such as BPMN (cf. Section 5.5.2), usability and an active developer community with support provision. The Oryx editor is based on an Ajax-JavaScript Web frontend combined with a Java-Servlet-based backend including a Hibernate persistence layer. The open approach combined with its clearly structured and defined plug-in mechanism provides a solid foundation for the realization of the upper layers. A comprehensive description about the Oryx editor can be found in [234].

According to the client-server architecture and the extension mechanism of the Oryx editor and repository the upper layers have been realized as plug-ins. Its functional components contain each a client-side Web frontend for the GUI including as well lightweight application logic and a server-side backend for sophisticated processing (cf. Figure 6-1). The backend includes in particular the functionality for the additional semantic layers using the API of the Protégé Semantic Web framework and the persistence handling of extended business process models with semantic artifacts such as domain ontologies for information entity annotation and semantic bridges for semantic mediation.

Semantic Annotation: The semantic annotation plug-in extends the Oryx BPMN modeling functionality by means of semantically enriched expression of information entities and information flow. The Oryx editor supports multiple notations for business process models. However, BPMN was chosen due to its standardization and wide industry adoption. The developed extension allows to link BPMN information entities to concrete concepts of a domain ontology described in OWL-DL. Among the three available OWL languages levels (cf. Section 3.3.2) OWL-DL has been chosen for the following reasons: OWL-DL imposes some restrictions on the underlying RDF graphs in comparison to OWL-Full, which does not. Therefore, based on limitations in expressiveness OWL-DL is decidable and reasonably computable compared to OWL-Full. Moreover, OWL-DL enables arbitrary values for cardinality restrictions of properties, whereas OWL-Lite only allows to distinguish between 0 and 1 for *minCardinality*, etc. In this context it has to be kept in mind that in the following tools of the semantic mediation toolkit (e.g. in the mediated business process execution tool) OWL concepts need to be mapped to XML Schema types due the aimed anticipation of path dependency of Web service technology. The OWL-DL features for cardinality restrictions allow to cover the XML Schema feature of defining how often an element may occur within a complex type definition by *minOccurs* and *maxOccurs*, which are not restricted to 0 or 1. Therefore, the language level OWL-DL has been regarded as most useful for the developed prototypes including the fact that most OWL reasoners focus as well on OWL-DL.

By selecting a concept from a tree-based representation of the OWL-DL domain ontology graph the adequate semantic annotation for the business process information entity can be identified. This representation also incorporates the domain ontology context and the properties describing the particular concept. In particular, the concept-tree representation focuses on the following OWL-DL ontology features:

- the concept and its sub-concepts
- the concept and its properties
- the properties and their domain and range (cf. Section 3.3.2)

Further more sophisticated ontology features (e.g. the qualification of a property as functional etc.) are not represented, in order to focus on a high-level non-technical visualization for business experts without too much complexity. Therefore, the OWL-DL ontology is processed on the server-side using the Protégé-API. In order to display the ontology structure, an XML object representation is utilized to serialize ontology representations to the client-side. The information flow is represented in BPMN by means of linking annotated information entities to multiple activities between information entities are exchanged. Besides this visualization of the annotation which focuses on the whole domain information model, additionally a directly into BPMN integrated semantic extension of the information entity representation is provided. According to the approach presented in Section 5.5.1 each annotated information entity is visualized in the business process model with the corresponding concept properties as cascading sub-shapes using SVG [235]. Figure 6-2 below illustrates the two perspectives of the semantic annotation in a screenshot of the Web-based GUI of the extended business process modeling tool. The BPMN extension of the annotated information entity is shown on the left hand side and the domain ontology browser including the concept tree on the right hand side. In the center a semantic bridge between the two semantically corresponding information entities *PurchaseOrder* and *Order* in different domains is highlighted, which will be further explained in the next paragraph.

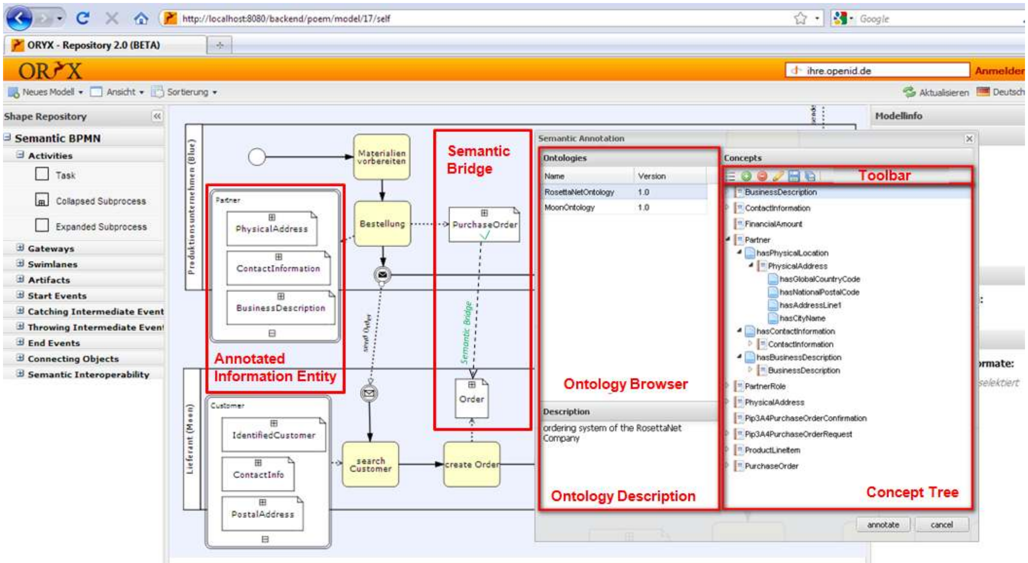


Figure 6-2 GUI of Mediated Business Process Modeling Tool

The functionality to advance the domain ontology or particular concepts is integrated in the tree-based representation of the OWL-DL domain ontology graph. Basic extension features such as adding, changing or deleting a concept or a property can be directly performed in the provided ontology browser. This can be done through specific buttons and context menus highlighted as the toolbar in Figure 6-2. More sophisticated advancements of the ontology can be specified in terms of textual comments by process experts and then have to be externally incorporated into the domain ontology by domain information model experts. They can utilize a separate tool dedicated to ontology editing as presented in context of existing work for the methodology step of domain ontology development (cf. Section 5.4).

Semantic Mediation: The semantic mediation plug-in applies preloaded semantic bridges, in order to match semantically corresponding information entities in the cross-organizational business process model. As the application of semantic bridges represents the core part of the actual semantic mediation mechanism (cf. Section 4.8), its realization is discussed in more detail in this section and is then just referenced when applied in further tools of the toolkit. The semantic bridges are realized in terms of SWRL forward-chaining rules combined with the facet classification semantics of OWL (cf. Section 3.3.2). Therefore, the OWL feature of class definitions through property restrictions [236] is exploited. This way classes are defined in terms of at least one necessary and sufficient condition describing exactly the properties which an instance of this class should exhibit. Consequently, instances which fulfill the conditions can be classified by an OWL reasoner as members of such a defined class. Different third-party reasoners and rule engines have been examined in order to interpret and execute the SWRL rules and perform the required classification.

At first, KAON2 [237] was investigated, since it supports reasoning over OWL and SWRL. However, KAON2 implements a pure backward-chaining algorithm, which is designed for query answering; i.e. only facts necessary for answering one specific question are generated. It does not support the calculation of all facts based on a given knowledge base. That means it is not possible to trigger a forward-chaining reasoning, determining all facts that can be inferred from the given knowledge base [237]. However, this is necessary to generate polymorph instances containing all properties of each concept definition from the domain ontologies between the semantic bridge is applied. Furthermore, the SWRL support of the Pellet reasoner [238] has been evaluated, which however at time of investigation did not include required built-ins [239] such as the *makeOWLThing*-built-in [240]. However, this built-in is necessary e.g. if a semantic bridge defines a mapping between concepts where the target concept is defined on a finer granularity level then the source concept. In this case new OWL individuals have to be created to accommodate the additional level of structure. Due to the lack of support of such specific built-ins, the Pellet SWRL support is not sufficient.

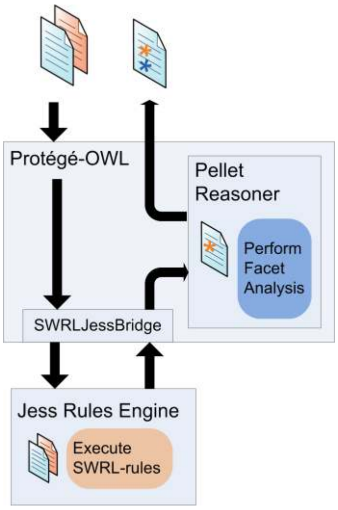


Figure 6-3 Realization of Semantic Mediation Mechanism

A third and finally adopted approach is the use of the Protégé-OWL Framework [241] in combination with Pellet and the Jess Rule Engine [242]. Protégé is an open source tool for managing and manipulating OWL. It provides a direct connection to Pellet for performing OWL-DL reasoning applied for facet classification. Since Pellet does not support all the SWRL-built-in-libraries as discussed above, the Jess rule engine is used for this purpose. The right hand Figure 6-3 illustrates this realization of the semantic mediation mechanism. Protégé is utilized as the top-layer framework that coordinates the communication between the other frameworks. It is responsible for reading, importing and managing all ontological facts. While Pellet is directly integrated into the Protégé framework, Jess is an independent component by itself. Therefore, all the facts that are necessary for executing the SWRL-rules have to be transferred to the rule engine via the SWRL-JessBridge [243]. The available methods and the syntax for handling the rules are explained in [244]. Since rules operate on individuals exclusively, proxy OWL individuals have to be created for all ontology concepts involved in the semantic bridge. The proxy individuals simulate the actual instances of information entities that will be provided during process execution. After the semantic bride is executed, the now polymorph proxy individuals can be visualized in the concept tree of the ontology browser and as well directly in the BPMN model as illustrated in Figure 6-4 below:

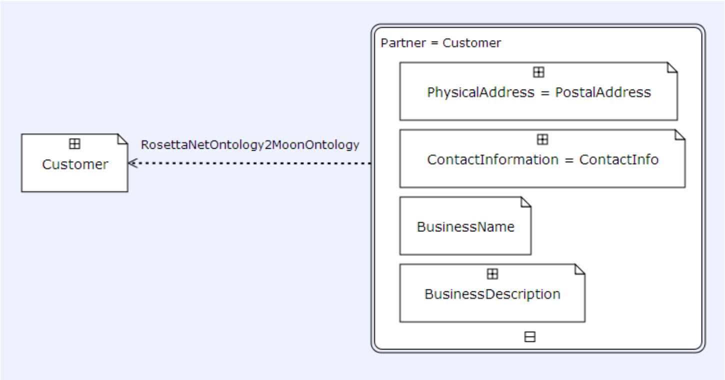
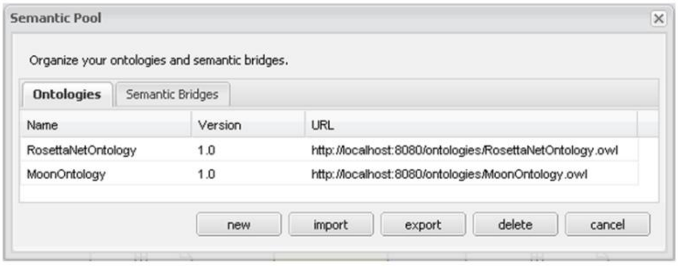


Figure 6-4 Polymorph Information Entities embedded in BPMN

Moreover, based on the polymorph proxy individuals the matching of semantically corresponding information entities across business domains can be performed. Iterating over the involved information entities in the process model by taking into account recursively sub-elements, the concept types can be directly compared. Matching information entities are highlighted and presented to the process expert. Furthermore, in an analog manner to the evolution of information models the tool provides advancement functionalities for the evolution of rule-based semantic bridges. Requirements for missing mapping rules between two concepts or as well for semantic bridges missing at all can be specified and are stored as textual comments to be addressed by domain experts. The technical realization of the visualization and advancement functionalities for semantic bridges are similarly realized as for the semantic annotations as described above.

Semantic Pool: On the one hand, the semantic pool plug-in provides a repository to handle and manage domain ontologies to be used during annotation of information flow. On the other hand, the analog functionality is provided to manage the used semantic bridges including dialogs for import, export, create and versioning operations required for the advancement functionality discussed above. After importing a domain ontology its URL is parsed on server-side and the tree-based representation for the client-side ontology browser is generated and stored. Thereby, the realization takes into account that the used ontologies and semantic bridges are persistently integrated into the data set of the business process model to restore them consistently when the business process is reloaded. The following Figure 6-5 shows the visualization of the semantic pool plug-in.



Organize your ontologies and semantic bridges.

Name	Version	URL
RosettaNetOntology	1.0	http://localhost:8080/ontologies/RosettaNetOntology.owl
MoonOntology	1.0	http://localhost:8080/ontologies/MoonOntology.owl

Figure 6-5 Realization of Semantic Pool

Semantic Web Framework: The Semantic Web framework has been chosen and utilized as discussed above in context of the realization of the semantic mediation mechanism. It includes the Protégé API framework combined with the Pellet OWL-DL reasoner and the Jess SWRL rule engine.

This section has focused on the most important aspects of the tool's implementation. A more detailed description of the tools technical realization including class diagrams, sample ontologies, test cases, lessons learned are provided in [193] and are discussed in the paper [245].

6.2.3 Scenario, Validation and Verification

This section covers the validation and verification of the developed prototype for mediated business process modeling. Based on a briefly outlined scenario a subsection about validation analyses how the developed prototype meets the objectives defined in the system requirements (cf. Section 6.2.1). A further subsection about verification deals with the question whether the prototype performs the mediated business modeling tasks correctly. An evaluation of the semantic mediation approach in general is given in Chapter 7.

The performed scenario is based on the "Purchase Order Mediation Scenario", which has been issued by the international Semantic Web Service Challenge (SWSC) [246]. It is highlighted from different perspectives according to the respective focus of each prototype in the toolkit. In this section the focus is put on the modeling of a cross-organizational business process including the design of information flow across heterogeneously defined information representations. The basic idea of the scenario is that a customer "Blue" wants to purchase goods from the manufacturer "Moon". However, the systems responsible for issuing a purchase order on the Blue side and for processing the order on the Moon side differ in terms of information representation and in terms of interaction patterns. I.e. the granularity and denotation of the data elements used on both sides varies, as does the order and granularity of operations, necessary to complete the processing of an order. The following Figure 6-6 Purchase Order Mediation Scenario Overview illustrates the scenario:

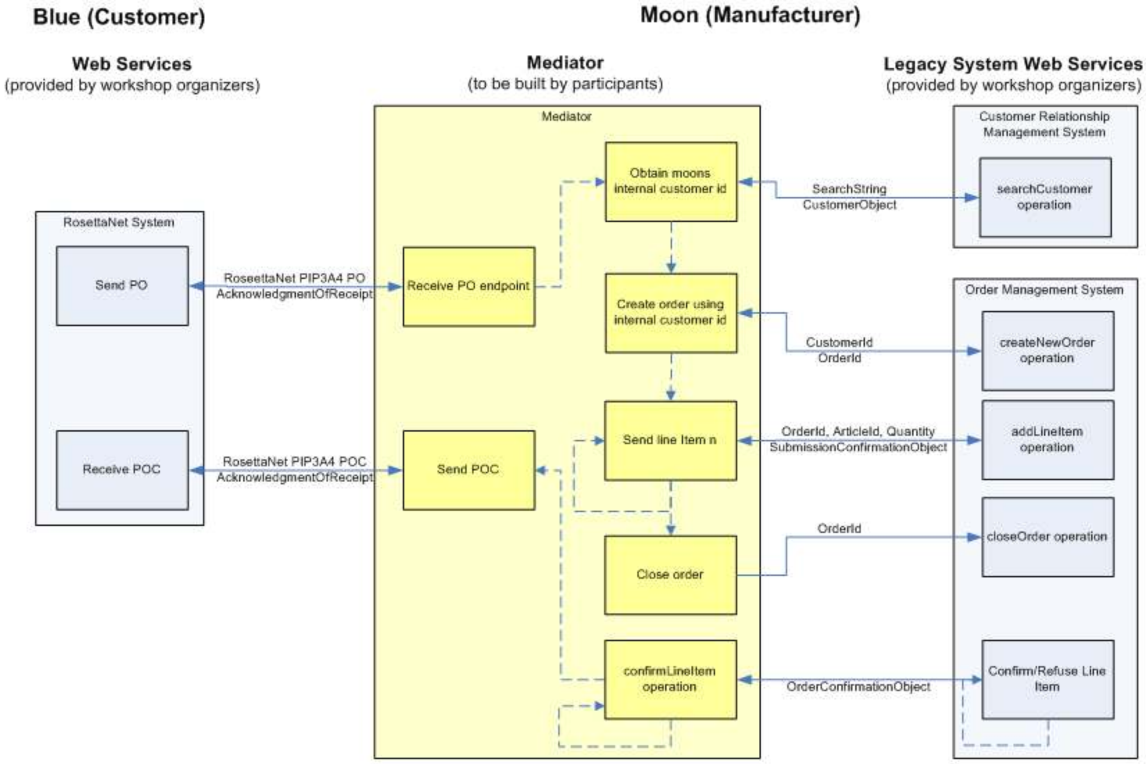


Figure 6-6 Purchase Order Mediation Scenario Overview [246]

The purchase order sent by the Blue system is based on an information model specified in the RosettaNet XML Schema standard, while the Moon system defines its own information model with a proprietary XML Schema format. Consequently, the challenge is to implement a mediator that should bridge the heterogeneities regarding the different information models and interaction patterns of the two systems. The business process between the two companies has been modeled with the developed prototype. Furthermore, the conceptual information flow has been designed based on developed domain ontologies “Blue” and “Moon”, which capture the different conceptualizations of the information models on an ontology level. Moreover, additionally developed semantic bridges between the Blue and the Moon domain ontology have been applied, in order to provide a transparent semantic mediation between the heterogeneous information models to the business process expert. The initial development of the domain ontologies corresponding to the provided XML-Schemas and the semantic bridges is further described in the context of the next prototype covering the systematic testing of semantic bridges. Furthermore, the advancement functionality for information models and semantic bridges of the mediated business process modeling tool has been used to complete the Blue and Moon domain ontologies and corresponding semantic bridges. Figure 6-7 below shows how the purchase order mediation scenario has been mapped to a business process model developed with the realized prototype:

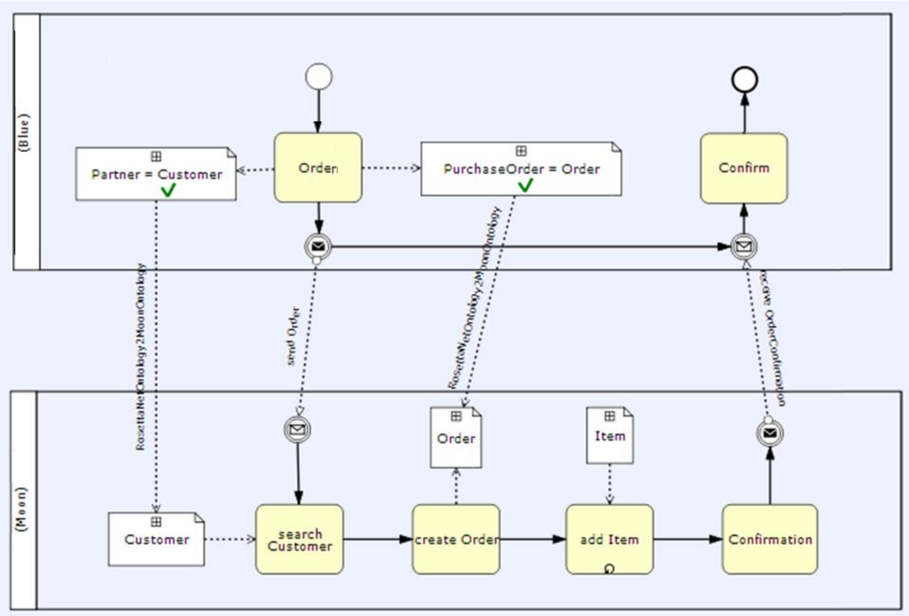


Figure 6-7 Scenario Performed with Mediated Business Process Modeling Prototype

Validation

The required functionality for the design of information flow on a non-technical conceptual level could be successfully provided: On the one hand by means of the ontology browser presenting the information model as a concept tree and on the other hand by extending the plain BPMN representation of information entities with concept annotations from a domain ontology. Furthermore, based on the underlying OWL-DL domain ontologies utilized for the expression of domain information models the link to their processing in the further SOA life-cycle could be ensured. I.e. the OWL-DL domain ontologies can be reused, in order to build the additional semantic layer for Web service enrichment and Web service composition, which can be mapped to existing XML-based infrastructures and hence contribute to improved business IT alignment. The semantic mediation mechanism has been realized by means of SWRL-based forward-chaining rules and the facet analysis classification of an OWL-DL reasoner. Once loaded to the semantic pool and accordingly applied with the matching functionality, the semantic bridges enable a seamless design of cross-organizational information flow, whereas semantic heterogeneities are kept transparently in the background. Finally, the functionality for business process-driven evolution of existing information models has been realized within the ontology browser and dialogs for the extension or completion of semantic bridges have been integrated. Therefore, it can be stated that the developed prototype for mediated business process modeling meets the objectives as defined in its system requirements (cf. Section 6.2.1).

Verification

Guided by the scenario several tests were run to ensure that the prototype mainly performs correctly and stable in the expected behavior. In particular, a set of defined use cases covering the main functionalities have been addressed. The tested use cases include: manage ontologies and semantic bridges in the semantic pool, create new ontology, define requirements for a new semantic bridge, annotate information entity with a concept, display annotated information entity, edit annotated information entity, link corresponding information entities with a semantic bridge, suggest semantically matching information entities, display semantic bridge and finally edit semantic bridge. Additionally, during the development of the prototype several unit tests were run to check the correct implementation of the various methods. Further details of the performed scenario are provided in [193].

To summarize, it can be stated that the prototype generally performs correctly. And taken into account as well the validation discussion, it can be concluded that the implemented prototype demonstrates successfully how the mediated business process modeling step of the semantic mediation methodology can be instantiated as a proof-of-concept.

6.3 Semantic Bridge Testing Tool

The testing of semantic bridges constitutes the second step of the developed semantic mediation methodology. While tools for the creation of SWRL-based ontology mappings exist, the systematic testing of such mappings has not yet been addressed. Hence, a prototypical instantiation within the semantic mediation toolkit has been developed that focuses on the visual design and execution of tests for rule-based semantic bridges between domain ontologies. This section is structured analog to the previous one starting with a summary of the system requirements, over the description of the design and realization of the system architecture and its components to its application in terms of a scenario for the tool's validation and verification.

6.3.1 System Requirements

The goals for the methodology step of semantic bridge testing together with its functional architecture have been already discussed in Section 5.6. They can be referred to as the system requirements of the envisioned prototypical instantiation. They are recalled and summarized as follows:

- Provide functionality to define test cases for semantic bridges, which include exemplarily instances of concepts from the source ontology as test inputs and expected polymorph instances as expected test outputs (expressed according to source and target ontology).
- Integrate the functionality to execute the defined semantic bridge test cases under the required execution conditions (in terms of availability of source and target ontology).
- Provide test reporting functionality which evaluates the semantic bridge test cases based on a gap analysis between intended results defined in the test case and actual results received after semantic bridge execution. Furthermore, a calculation should be performed that quantifies the correctness degree of a semantic bridge. On the one hand, this evaluation should include a comparison of how many test cases have succeeded vs. how many have failed. And on the other hand, the coverage of the whole test with regard to the overall scope of a semantic bridge should be measured.

The functional architecture as discussed in Section 5.7.2 is based on the Model-View-Controller (MVC) approach including: model components representing the ontologies, semantic bridges and test projects, the GUI for domain information model experts and modelers of business processes or service compositions and finally the controller components handling the interactions as well as providing the actual business logic for the semantic bridge test execution. Based on this functional perspective the following section refines this functional architecture to the tool's system architecture, whereas each component is described in terms of its concrete technical instantiation.

6.3.2 Design and Realization

Analog to the other prototypes of the semantic mediation toolkit the semantic bridge testing tool has been iteratively implemented according to the customized Agile Unified Process (AUP). This process has ensured high quality and enough flexibility to choose between different realization options. The MVC architecture style is reflected in structuring the tool into subsystems organized in the three main packages: model, controller and GUI. The following Figure 6-8 presents the designed system architecture:

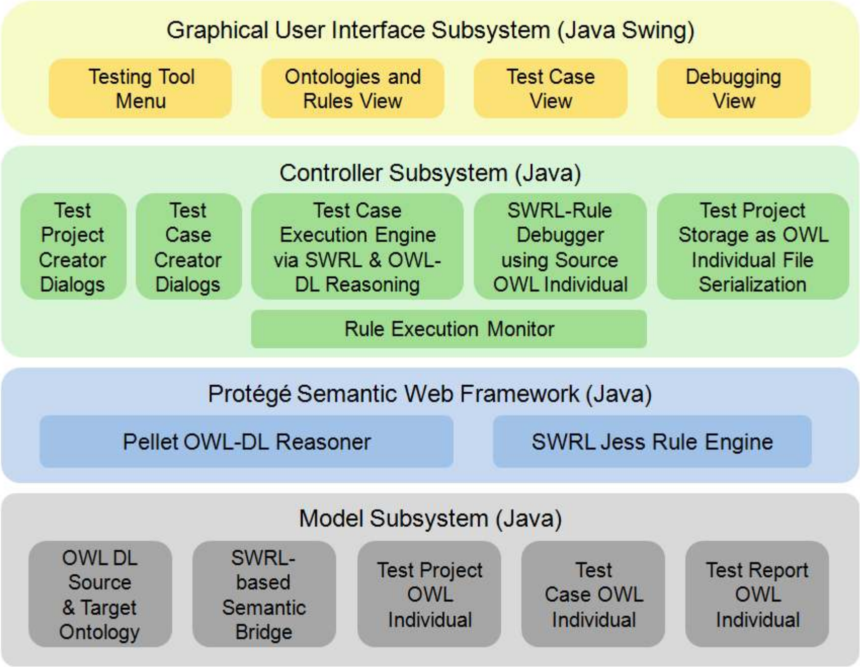


Figure 6-8 System Architecture of Semantic Bridge Testing Tool

Model Subsystem: The model subsystem manages the assets on which the testing tool operates. It is realized in Java following an object-oriented approach. Firstly, it contains a component that maintains the source and target domain ontologies of the semantic bridge under test. As the testing tool is part of the semantic mediation toolkit, the supported language for specifying domain ontologies is OWL-DL according to the other prototypes.

A further component encapsulates a representation of the semantic bridge to be tested, which is based on SWRL forward-chaining rules. Moreover, the internal representation of the test project and its containing test cases are realized as OWL individuals, whereas a dedicated ontology describing the required information has been developed. The right hand Figure 6-9 shows the required properties of a test project and test cases. The test project representation by means of an ontology has been chosen for two reasons: Firstly, the required technology for handling and storing the OWL files in a persistent form is already in use within the prototype and thus convenient to reuse. And secondly and more important, it provides a suitable form for exchange of test projects among domain information model creators and users.

Figure 6-9 Test Project Ontology

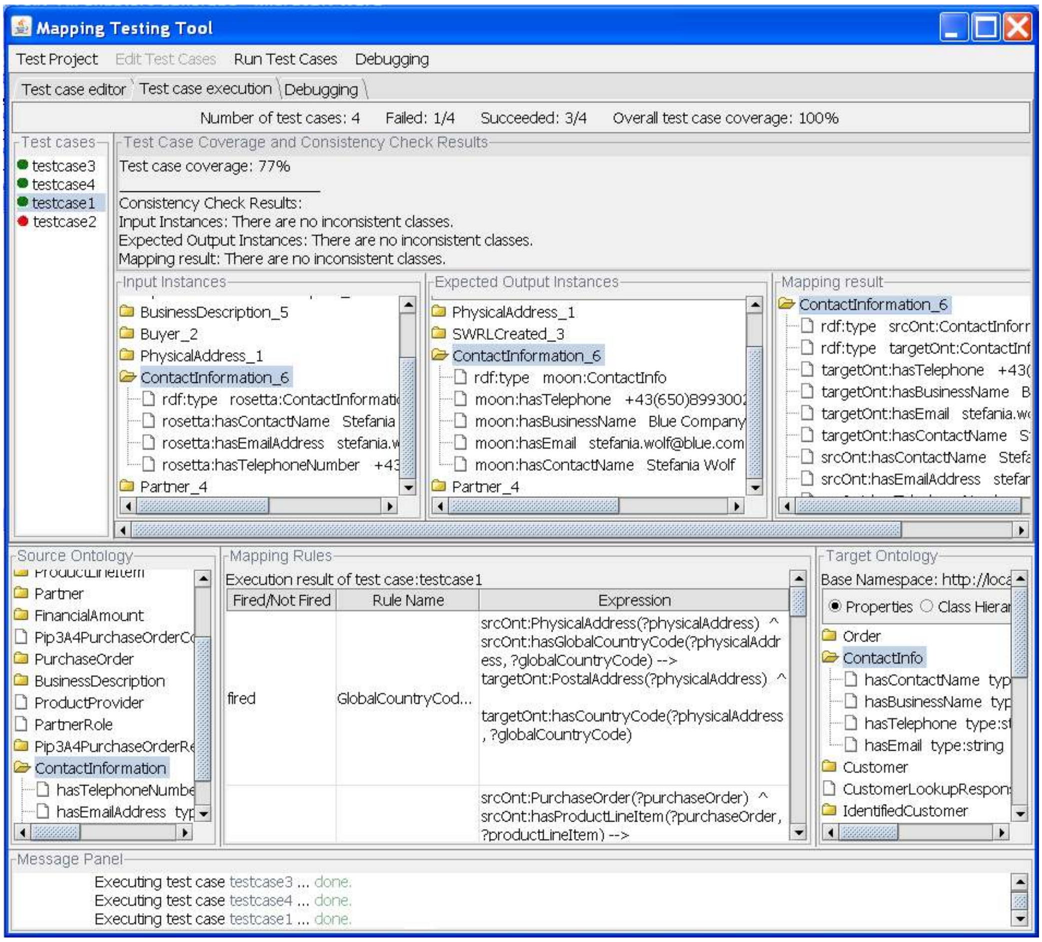
Analogically, the test report is represented as an OWL individual. The test report contains information whether a respective test case has succeeded or failed. If a test case has failed, the test report also lists statements in the expected test output which could not be found or inferred in the mapping result and hence caused a test case to fail. Furthermore, the report for each test case contains information regarding rules that have been fired. Finally, information about test case coverage of a particular test case and the overall test coverage is provided (cf. Section 5.7.1).

Semantic Web Framework: The Semantic Web framework utilized in the prototype for test case execution is the same as it has been exploited for the semantic mediation mechanism as described in context of the prototype of mediated business process modeling (cf. Section 6.2.2). This includes the object-oriented representations of ontologies and semantic bridges as utilized in the model subsystem.

Controller Subsystem: The controller subsystem contains the actual business logic of the testing tool, which is as well implemented in Java. Firstly, it provides the interaction logic in terms of dialogs for creating test projects and included test cases. The core component is the test case execution engine, which reads the test project description and performs the individual test cases based on the already discussed semantic mediation mechanism applying the semantic bridge's mapping rules (cf. Section 6.2.2). During test case execution the rule execution monitor determines which mapping rules have been fired. Therefore, specific axioms are temporarily added to each rule head and thus can be identified after rule execution. This information is required to prepare the test report. To compare a mapping result with expected test output instances, an inferred OWL model of the mapping result is derived with the objective to consider not only asserted but also inferred knowledge during comparison. Then, the test case execution engine checks whether each statement contained in the expected test output instances is also contained in the inferred mapping result model. If all the statements are present, the test case can be considered as succeeded.

The purpose of the rule debugger is to debug the mapping rules by means of executing only a selected set of rules. In this process, a concept individual of the source ontology, which has to be provided by the user, is utilized as the input. Similar to the test case execution engine the rule debugger uses the implemented semantic mediation mechanism to execute a set of mapping rules and the rule execution monitor to determine which rules have been fired. Finally, a component manages the test project storage and loading based on serialization of the OWL representation of test projects and included test cases according to the description in the context of the model subsystem.

GUI Subsystem: The GUI subsystem realizes the visual presentation relevant for editing and execution of test cases and rule debugging. The entry point for the functionality is organized in the testing tool menu. In the center, the ontologies and rules panels are located, which provide an overview of the rules constituting the semantic bridge and its source and target ontology. The following Figure 6-10 shows a screenshot of the GUI of the developed testing tool for semantic bridges, which is based on Java Swing:



Number of test cases: 4 Failed: 1/4 Succeeded: 3/4 Overall test case coverage: 100%

● testcase3
● testcase4
● testcase1
● testcase2

Test case coverage: 77%

Consistency Check Results:
Input Instances: There are no inconsistent classes.
Expected Output Instances: There are no inconsistent classes.
Mapping result: There are no inconsistent classes.

BusinessDescription_5
Buyer_2
PhysicalAddress_1
ContactInformation_6
rdf:type rosetta:ContactInformati
rosetta:hasContactName Stefania
rosetta:hasEmailAddress stefania.w
rosetta:hasTelephoneNumber +43
Partner_4

PhysicalAddress_1
SWRLCreated_3
ContactInformation_6
rdf:type moon:ContactInfo
moon:hasTelephone +43(650)899300
moon:hasBusinessName Blue Company
moon:hasEmail stefania.wolf@blue.com
moon:hasContactName Stefania Wolf
Partner_4

ContactInformation_6
rdf:type srcOnt:ContactInforr
rdf:type targetOnt:ContactInf
targetOnt:hasTelephone +43(
targetOnt:hasBusinessName B
targetOnt:hasEmail stefania.wo
targetOnt:hasContactName S
srcOnt:hasContactName Stefa
srcOnt:hasEmailAddress stefar

ProductLineItem
Partner
FinancialAmount
Pip3A4PurchaseOrderCc
PurchaseOrder
BusinessDescription
ProductProvider
PartnerRole
Pip3A4PurchaseOrderRe
ContactInformation
hasTelephoneNumbe
hasEmailAddress typ

Execution result of test case:testcase1

Fired/Not Fired	Rule Name	Expression
fired	GlobalCountryCod...	srcOnt:PhysicalAddress(?physicalAddress) ^ srcOnt:hasGlobalCountryCode(?physicalAddress, ?globalCountryCode) --> targetOnt:PostalAddress(?physicalAddress) ^ targetOnt:hasCountryCode(?physicalAddress, ?globalCountryCode)
		srcOnt:PurchaseOrder(?purchaseOrder) ^ srcOnt:hasProductLineItem(?purchaseOrder, ?productLineItem) -->

Base Namespace: http://loca
Order
ContactInfo
hasContactName typ
hasBusinessName typ
hasTelephone type:st
hasEmail type:string
Customer
CustomerLookupRespon
IdentifiedCustomer

```
Executing test case testcase3 ... done.
Executing test case testcase4 ... done.
Executing test case testcase1 ... done.
```

Figure 6-10 GUI of Semantic Bridge Testing Tool

The lower left hand side shows the source ontology view, the lower center the mapping rules and the lower right hand side the target ontology. The message panel at the bottom informs about the tool's actions performed during test case creation, execution and rules debugging.

Furthermore, the functionalities for the visual design of test cases and their combination to test projects have been implemented in a test case editor tab. The input instances are either defined by the domain expert or generated automatically in terms of dummy instances by the testing tool. The upper side of Figure 6-10 shows the test case view after test execution as a comparison between input instances, expected output instances and actual mapping results. Furthermore, the test case coverage and consistency check results are presented. Similarly, the debugging view is realized, whereas the domain expert can select the semantic mapping rules to be executed while observing the mapping results.

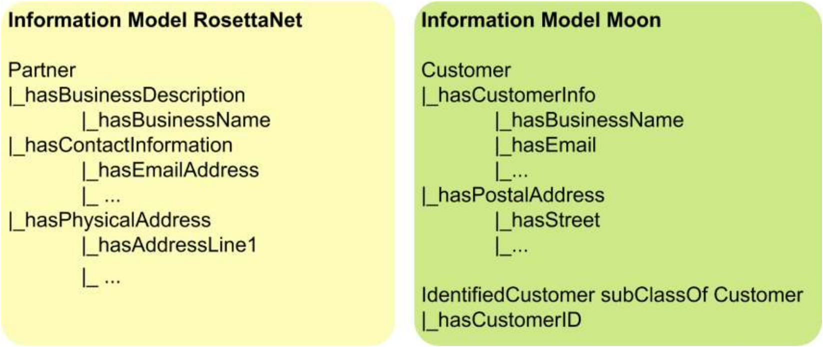
Finally, the tool presents a generated test report that summarizes the test results, consistency checks and the test case coverage. The latter is an indicator used to quantify the expressiveness of the performed test case as it has been mentioned above (cf. Section 5.7.1). The test report serves as a basis for conclusions regarding correctness of the semantic bridge or identification of defects within its underlying mapping rules.

More details on the design, system architecture and the implementation of this proof-of-concept including class diagrams, sample ontologies, test cases and lessons learned are provided in [225].

6.3.3 Scenario, Validation and Verification

In order to demonstrate the prototype of the mapping testing tool and provide a means for its validation and verification, it has been applied to the “Purchase Order Scenario” already presented in context of the prototype for mediated process modeling. This time the scenario is highlighted from the perspective of the two heterogeneous domain ontologies and the testing of the semantic bridge between them. Again, first a validation part recalls the originally defined system requirements (cf. Section 6.3.1) and applies them as evaluation criteria. Secondly, a verification part deals with the question, whether the prototype performs the tasks for semantic bridge testing correctly.

As already mentioned the two domain ontologies “Blue” and “Moon” have been developed, in order to shift the information models provided as XML Schemas onto the higher semantically expressive conceptual level. For the purpose of providing a better understanding for the performed testing of the semantic bridge, the following Figure 6-11 illustrates an excerpt of the two heterogeneous ontologies representing the different information models of the Blue system (RosettaNet) and the Moon system (legacy):



Information Model RosettaNet

Partner
|_hasBusinessDescription
    |_hasBusinessName
|_hasContactInformation
    |_hasEmailAddress
    |_ ...
|_hasPhysicalAddress
    |_hasAddressLine1
    |_ ...

Information Model Moon

Customer
|_hasCustomerInfo
    |_hasBusinessName
    |_hasEmail
    |_...
|_hasPostalAddress
    |_hasStreet
    |_...

IdentifiedCustomer subClassOf Customer
|_hasCustomerID

Figure 6-11 Heterogeneous Domain Ontologies "Blue" and "Moon"

The information models differ in their semantic sub-graph. As the concept *Partner* in the RosettaNet ontology is defined in terms of three object properties, a semantically corresponding concept *Customer* in the Moon ontology just features two object properties. The two concepts contain the same information, however defined at a different level of granularity. The domain ontologies have been designed according to the methodology step of domain information model development based on the tools outlined in the existing work part (cf. Section 5.4.2). In particular, the Protégé ontology editor has been applied. Furthermore, also the functionality for information model evolution provided by the developed mediated business process modeling prototype (cf. Section 6.2) has been taken into account to complete the information model aligned to the business process requirements.

Obviously, the *Partner* and *Customer* concepts presented above cannot be exchanged between communicating partners by default, although they represent the same conceptual idea. In order to mediate between these differently represented concepts, a semantic bridge between the two heterogeneous ontologies has been developed according to the methodology step of semantic bridge definition (cf. Section 5.6). The following screenshot Figure 6-12 shows an excerpt of mapping rules created with the Snoggle mapping tool presented in Section 5.6.2.

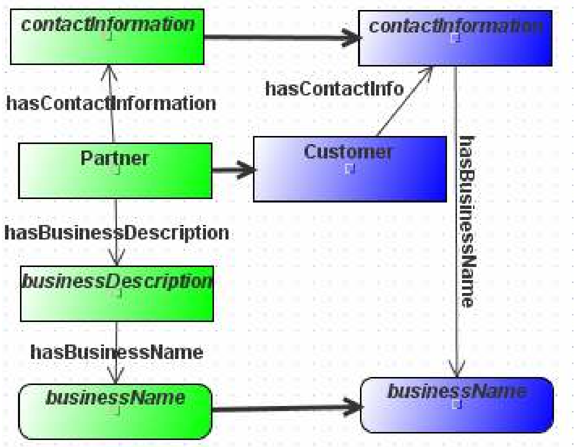
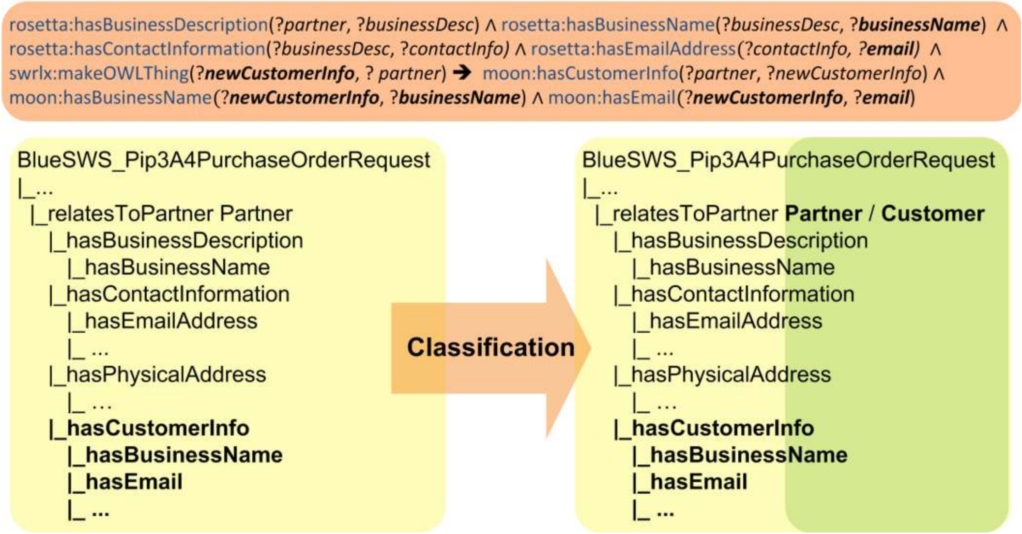


Figure 6-12 Example Mapping Rules Created with Snoggle Mapping Tool

By applying the semantic bridge rules, an instance of type *Partner* is furnished with additional properties e.g. with *hasCustomerInfo* combining the values of the *BusinessDescription* and the *ContactInformation* properties *hasBusinessName* and *hasEmailaddress*. Having the class definitions on hand, a reasoner is now able to classify the instance as a member of the defined class *Customer*, since all required properties (including *hasCustomerInfo*) are present. The following Figure 6-13 illustrates the underlying logic of the mapping rules in a so called human-readable syntax realizing this part of the semantic bridge:



rosetta:hasBusinessDescription(?*partner*, ?*businessDesc*) ∧ rosetta:hasBusinessName(?*businessDesc*, ?***businessName***) ∧ rosetta:hasContactInformation(?*businessDesc*, ?*contactInfo)* ∧ rosetta:hasEmailAddress(?*contactInfo*, ?***email****)* ∧ swrlx:makeOWLThing(?***newCustomerInfo***, ? *partner*) ➔ moon:hasCustomerInfo(?*partner*, ?*newCustomerInfo*) ∧ moon:hasBusinessName(?***newCustomerInfo***, ?***businessName***) ∧ moon:hasEmail(?***newCustomerInfo***, ?***email***)

```
BlueSWS_Pip3A4PurchaseOrderRequest
|_...
 |_relatesToPartner Partner
   |_hasBusinessDescription
     |_hasBusinessName
   |_hasContactInformation
     |_hasEmailAddress
     |_ ...
   |_hasPhysicalAddress
     |_ ...
   |_hasCustomerInfo
     |_hasBusinessName
     |_hasEmail
     |_ ...
```

```
BlueSWS_Pip3A4PurchaseOrderRequest
|_...
 |_relatesToPartner Partner / Customer
   |_hasBusinessDescription
     |_hasBusinessName
   |_hasContactInformation
     |_hasEmailAddress
     |_ ...
   |_hasPhysicalAddress
     |_ ...
   |_hasCustomerInfo
     |_hasBusinessName
     |_hasEmail
     |_ ...
```

Figure 6-13 Semantic Bridge and Polymorph Classification Example

In order to test this semantic bridge, a test project with the following three test cases has been defined and executed:

- Test Case 1 is used for testing the mapping rules between the concepts *Partner*, *ContactInformation*, *PhysicalAddress* and BusinessDescription of the Blue ontology and their corresponding concepts in the Moon ontology.
- Test Case 2 is used for testing the mapping between the same concepts; however provided input instances of the concept *ContactInformation* are not featured with the property *hasEmailAddress*.

- Test Case 3 also tests the mapping between the above mentioned concepts, however an instance of concept *PhysicalAddress* is not featured with the property *hasHouseNr*.

During test case execution Test Case 1 and Test Case 3 are supposed to succeed and Test Case 2 is supposed to fail, due to the missing but required property *hasEmailAddress* in the provided test input instance. However, the failing of Test Case 2 does not point to a defect in the defined semantic bridge but rather has been purposeful designed, in order to provoke a non-succeeding test case in an otherwise completely correct set of mapping rules. The already presented Figure 6-10 above shows a screenshot after test case execution including the three test cases presented above. More details on the performed scenario including complete ontology descriptions, semantic bridges, test case definitions and test reports are provided in [247].

Validation

The overall goal has been to increase trust in the quality of semantic bridges by supporting the evaluation of correctness of their underlying ontology mapping. Accordingly, three system requirements have been derived, which coverage is discussed in the following: The developed prototype provides the required functionality to define test cases for semantic bridges. The exemplary test input instances and expected output instances can be either provided by the user or are automatically generated in terms of so called dummy instances. Furthermore, the developed semantic mediation mechanism could be successfully reused, in order to execute the defined test cases. Finally, the envisioned test reporting functionality could be provided as a prototypical instantiation. The quantification of the quality of a semantic bridge in terms of comparison of succeeding test cases vs. overall test cases is calculated. Furthermore, test coverage with regard to the overall scope of a semantic bridge is derived by the tool and successfully tested in the presented scenario. It is important to recall that the absolute determination of correctness of a semantic bridge cannot be provided by the tool and has also not been targeted due to the limitations of testing in general as discussed in Section 5.7.1. However, a successful test case execution in combination with a high indicator for test case coverage contributes to increase trust in the correctness of the created mapping. Moreover, the debugging mode of the tool helps to eliminate errors in the semantic bridge development phase. Consequently, it can be stated that the developed prototype for semantic bridge testing meets the objectives as defined in its system requirements (cf. Section 6.3.1).

Verification

According to the described scenario several test were run to ensure that the prototype mainly performs correctly and stable in the expected behavior. In particular, the prototype was continuously applied during the definition of the semantic bridge between the “Blue” and “Moon” domain ontologies. Additionally, analog to the previous prototype, several unit tests were run during the development iterations to check the correct implementation of the various methods.

Thus, it can be stated that the prototype for testing of semantic bridges generally performs correctly. Taken as well into account the validation results, it can be concluded that the implemented prototype demonstrates successfully how the semantic bridge testing step of the semantic mediation methodology can be instantiated as a proof-of-concept.

6.4 Mediated Service Composition Tool

This section is dedicated to the prototypical instantiation of the methodology step of mediated service composition (cf. Section 5.9). It has been argued that although several approaches targeting the integration of Semantic Web technologies into Web service composition exist, their focus is put mainly on the automation of the composition process, however leaving aside the challenge of heterogeneous information models describing the involved Web services (cf. Section 3.4.3 and Section 5.9.3). The concept of mediated service composition has been explained in detail along with its required semantic components and how they should be used. In order to proof this methodology step, a prototypical composition tool has been developed, which is described in this section. The section is structured according to the ones before, starting with a summary of the system requirements, over the description of the design and realization of the system architecture and its components to its application in terms of a scenario for the tool's validation and verification.

6.4.1 System Requirements

The main goals of the methodology step of service composition together with its functional architecture discussed in Section 5.7 can be taken as the system requirements of the envisioned tool. In order to recall them and provide a consistent understanding for the following sections they are summarized as follows:

- Provide functionality for the visual design of compositions of semantically enriched Web services, whereas the focus should be put on the design of information flow, i.e. the assignment of input and output parameters between the services.
- During the design process the tool should exploit the semantically enriched Web service descriptions for service selection based on semantic match making. I.e. the visual tool should recommend services, which input and output parameters can be assigned semantically sound to each other.
- Furthermore, provided semantic bridges should be utilized to mediate between the different domain conceptualizations describing the Semantic Web services. The semantic mediation should be handled transparently for the service composer to overcome technical transformation coding and reduce integration efforts.
- Finally, the tool should serialize the designed composition to a Semantic Web service orchestration plan based on the industry standard BPEL. The generated orchestration plan should be executable in available BPEL process engines, while preserving the capabilities of the exploited ontology-based technology such as polymorphism etc. (cf. Section 4.8).

The functional architecture as discussed in Section 5.9.2 is based on the following components: semantic bridging engine, matching engine, visual modeling tool and deployment engine and supported by a Semantic Web framework. They process the artifacts including domain ontologies, Semantic Web service description and semantic bridges, in order to produce an executable orchestration plan. In the following this functional architecture is refined to the tool's system architecture, whereas each component is described in terms of its concrete technical instantiation.

6.4.2 Design and Realization

Again the customized Agile Unified Process (AUP) has been applied for the development phase. Furthermore, the architecture reflects as well the MCV style. However, for this prototype this approach is not highlighted furthermore but still remains as the underlying functional architecture style. More central for this prototype are the various artifacts processed by the tool, which are therefore focused in the system architecture following the information viewpoint (cf. Section 3.5.2). Accordingly, the system architecture for the mediated composition tool is illustrated in the following Figure 6-14. It provides an overview on the applied technologies for the different components of the prototype.

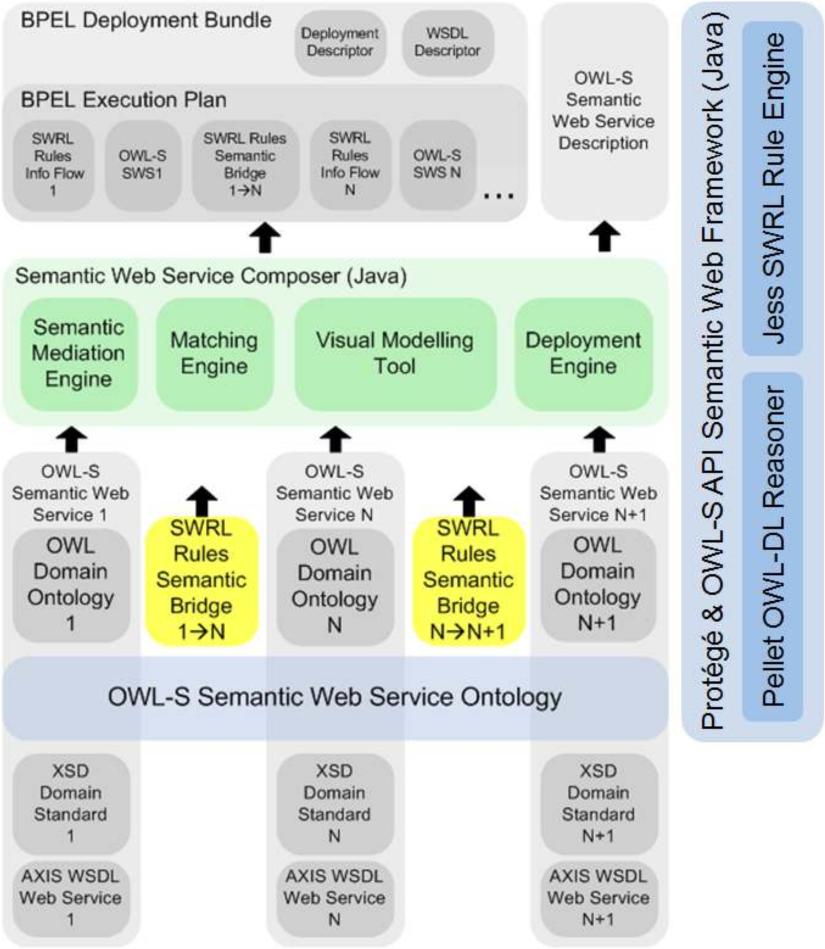


Figure 6-14 System Architecture of Mediated Service Composition Tool

The subsequent paragraphs describe the components in detail and explain how they are related to each other:

Semantic Web Services: The tool requires the input of Semantic Web services, in order to compose them to an orchestration. Therefore, a set of exemplary Semantic Web service has been developed, which are further described in context of the performed scenario (cf. Section 6.4.3). The realization of the Semantic Web services has been achieved by defining OWL-S service descriptions and ground them to underlying traditional WSDL-based Web services as described in the existing work section of the methodology step of semantic service enrichment (cf. Section 5.8.2). The development of therein referenced domain ontologies and corresponding domain XML Schemas standards, which specify the input and output parameters in the WSDL descriptions, is explained as well in context of the scenario (cf. Section 6.4.3). Especially, XSLT transformations need to be developed, in order to realize the lifting and lowering between the different meta-data models of OWL-S and WSDL. For the development and provision of the underlying WSDL-based Web services, the Apache AXIS framework was used to map the Web service interfaces to the actual implementation of the service in terms of a Java components.

As outlined in Section 3.4.2, there exist several approaches for realizing Semantic Web services. In this work OWL-S Version 1.1 has been chosen, because OWL-S provides several advantages for the purpose of the presented approach. OWL-S has been the first of the submissions to the W3C in the context of Semantic Web services and many researchers regard OWL-S as the most mature Semantic Web service technology [108]. Compared to other approaches such as WSMO or SWSF, OWL-S is a light-weight approach. WSMO and SWSF concentrate on goal-based dynamic plan creation known from artificial intelligence research, which has turned out to be less suitable for the approach targeted in this work (cf. Section 3.4.3). Also it has been considered to move from OWL-S-based Semantic Web services to the newer light-weight approach of SAWSDL (cf. Section 3.4.2), which recently was adopted as a W3C recommendation. There is growing support in terms of available tools to create and parse WSDL files annotated with the extension attributes defined by SAWSDL. Moreover, it is already used in several research projects. However, at the time of this prototypical realization no available framework or API exists, that supports the complete process from parsing of an SAWSDL file to the execution of the Web services including the lowering and lifting of the inputs and outputs of the service. Therefore, SAWSDL could not be considered to be used in this work.

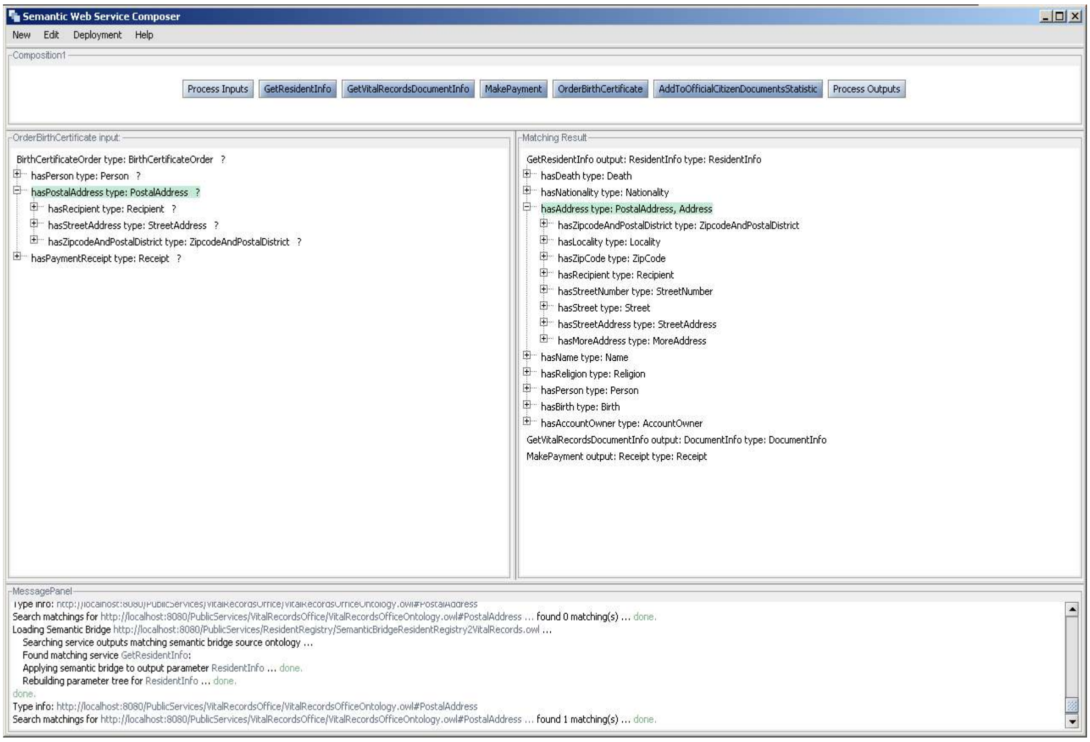
Semantic Bridges: The semantic bridges have been realized by means of SWRL forward-chaining rules as already elaborated in context of the previous prototypes (cf. Section 6.2.2). The serialization of the SWRL-based semantic bridges are OWL files, whereas a dedicated OWL ontology has been defined in the SWRL standardization process, which describes the content of the rules including body and head in terms of ontology concepts (cf. Section 3.3.2).

Semantic Web Framework: The Semantic Web framework utilized in the prototype is the same as it has been exploited for the semantic mediation mechanism as described in context of the prototype for mediated business process modeling (cf. Section 6.2.2). In the mediated service composition tool the framework has been extended with the Java-based OWL-S API [223]. It enables to parse and generate OWL-S-compliant Semantic Web service descriptions. Moreover, the OWL-S API also supports the execution of Semantic Web services including lifting and grounding, which is exploited for the mediated business process tool (cf. Section 6.5). Together, the framework is utilized for the handling of the Semantic Web services, the semantic bridges, the application of the semantic mediation mechanism and the matching mechanism discussed below.

Semantic Mediation Engine: The semantic mediation engine performs the SWRL rule-based mappings described in the semantic bridges. The engine is realized by reusing and integrating the already described instantiation of the semantic mediation mechanism (cf. Section 6.2.2). Applied to the composition task, this means that output parameters of Semantic Web services expressed in the source ontology are additionally expressed in the target ontology as polymorph representations. Thus, the service composer can assign them to inputs of subsequent Semantic Web services within the composition regardless of heterogeneous representations of their service parameters. Again, it has been taken into account that rules operate on OWL concept instances called individuals by binding variables to them. However, the service parameters are described by OWL concepts, which hinders that the semantic bridges can be directly work on them. Therefore, analog to the annotated information entities in the mediated process modeling tool (cf. Section 6.2.2), firstly proxy individuals are generated for the Semantic Web service outputs based on the concept definitions describing them.

Matching Engine: The matching engine performs reasoning over Semantic Web service parameters, which enables the composition tool to provide recommendations for suitable information flow assignments between the involved services. It exploits the semantically described relationships between concepts described in the utilized domain ontologies, such as inheritance or equality. As well the matching engine considers the polymorphism based on facet analysis classification and the transparently applied semantic bridges. In particular, the matching mechanism checks, if any concept C_{OUTPUT} describing the part of an output parameter of a service A fits to the concept C_{INPUT} describing the part of an input parameter of a service B. If they fit, then this matching represents an assignment candidate for semantically sound information flow between service A and B. The question whether two concepts fit, depends on their inferred relation to each other. If concept C_{OUTPUT} is a subconcept or defined as equal to the required concept C_{INPUT}, then these concepts fit. Furthermore, if a semantic bridge has been defined between two concepts they also fit, because due to inferred rules and facet classification they share the same type (cf. Section 4.8.4). The subclass relation ensures that all properties are inherited from a superclass. The equal relation ensures that the same or equal properties are available, so that the required properties can be inferred from the equal properties. And for mediated concepts the semantic bridges infer the required properties through semantic bridge rules. Thus, it is ensured, that the output described by concept C_{OUTPUT} provides all the properties required by concept C_{INPUT}. This can be considered as the core requirement for a proper assignment as the guarantied properties ensure that the forwarded data can be processed correctly.

Visual Modeling Tool: The visual modeling tool provides a Java Swing-based graphical user interface to model the information flow in Semantic Web service sequences. Due to the nature of a prototype, it has been focused on the information flow design and more sophisticated control flow functionalities going beyond simple service sequences have not been realized. The top panel of the visual modeling tool presents an overview of the current state of the designed service composition and serves as a navigation bar. Selecting a service, its input and output parameters are displayed in a parameter split pane in the middle. The following Figure 6-15 shows a screenshot of the prototypical tool captured during the matching process:



```
BirthCertificateOrder type: BirthCertificateOrder  ?
⊞ hasPerson type: Person  ?
⊟ hasPostalAddress type: PostalAddress  ?
    ⊞ hasRecipient type: Recipient  ?
    ⊞ hasStreetAddress type: StreetAddress  ?
    ⊞ hasZipcodeAndPostalDistrict type: ZipcodeAndPostalDistrict  ?
⊞ hasPaymentReceipt type: Receipt  ?
```

```
GetResidentInfo output: ResidentInfo type: ResidentInfo
⊞ hasDeath type: Death
⊞ hasNationality type: Nationality
⊟ hasAddress type: PostalAddress, Address
    ⊞ hasZipcodeAndPostalDistrict type: ZipcodeAndPostalDistrict
    ⊞ hasLocality type: Locality
    ⊞ hasZipCode type: ZipCode
    ⊞ hasRecipient type: Recipient
    ⊞ hasStreetNumber type: StreetNumber
    ⊞ hasStreet type: Street
    ⊞ hasStreetAddress type: StreetAddress
    ⊞ hasMoreAddress type: MoreAddress
⊞ hasName type: Name
⊞ hasReligion type: Religion
⊞ hasPerson type: Person
⊞ hasBirth type: Birth
⊞ hasAccountOwner type: AccountOwner
GetVitalRecordsDocumentInfo output: DocumentInfo type: DocumentInfo
MakePayment output: Receipt type: Receipt
```

```
Type info: http://localhost:8080/PublicServices/VitalRecordsOffice/VitalRecordsOfficeOntology.owl#PostalAddress
Search matchings for http://localhost:8080/PublicServices/VitalRecordsOffice/VitalRecordsOfficeOntology.owl#PostalAddress ... found 0 matching(s) ... done.
Loading Semantic Bridge http://localhost:8080/PublicServices/ResidentRegistry/SemanticBridgeResidentRegistry2VitalRecords.owl ...
  Searching service outputs matching semantic bridge source ontology ...
  Found matching service GetResidentInfo:
  Applying semantic bridge to output parameter ResidentInfo ... done.
  Rebuilding parameter tree for ResidentInfo ... done.
done.
Type info: http://localhost:8080/PublicServices/VitalRecordsOffice/VitalRecordsOfficeOntology.owl#PostalAddress
Search matchings for http://localhost:8080/PublicServices/VitalRecordsOffice/VitalRecordsOfficeOntology.owl#PostalAddress ... found 1 matching(s) ... done.
```

Figure 6-15 GUI of Mediated Service Composition Tool

Based on the automatically provided assignment recommendations, which are highlighted in green color, the service composer defines the appropriate information flow between the services according to the overall process logic. The same view is used to define the overall inputs and outputs of the designed composite service. Based on the assignments an information flow description can be formulated, which is integrated into the execution plan of the service composition. In order to remain consistently on the conceptual level followed in the Semantic Web service descriptions and the rule-based semantic bridges, the information flow is as well expressed by applying description logics. This means that SWRL-based forward-chaining rules specify how output instances of a service A are asserted to the input of a service B. Accordingly, on the one hand, semantic bridges and as well the information flows within a composition are implemented as forward-chaining rules. This way, information flow can be expressed as a rule, where the body of the rule describes the involved parameters and the head, which is inferred, describes the parameters after the information flow has been assigned. Finally, the bottom of the visual composition tool provides a message panel, which informs the user about the performed tasks.

Deployment Engine: Apart from the composition support, the tool is capable of exporting the defined service composition as an BPEL orchestration plan, which can be executed as a composite service in a process engine. The deployment engine is responsible for serializing the composition including the mapping rules of the semantic bridges, the rule-based information flow definitions and the Semantic Web service calls. However, due to the higher abstraction level during the composition design (based on the ontology meta-data model), the composed process cannot be directly expressed in a BPEL orchestration plan (based on the XML Schema meta-data model). Especially, the semantic bridges and the information flow descriptions expressed in description logic rules cannot be directly mapped to BPEL constructs. Therefore, an extension mechanism of BPEL has been exploited to incorporate these semantics-based features. The details of this extension mechanism and its reflection in an appropriate BPEL engine are discussed in the following prototype targeting the mediated business process execution (cf. Section 6.5). Furthermore, the tool generates additional necessary artifacts to construct a BPEL deployment bundle including an engine-specific deployment descriptor and a WSDL descriptor. This WSDL file describes the structure of the input and output messages of the designed process, in order to access it with conventional Web service technologies as required for a BPEL process. Furthermore, a Semantic Web service description is generated, which is grounded to the previously constructed WSDL file. Thus, the composition is finalized and available in an SOA landscape as a newly created composite Semantic Web service.

More details on the design, system architecture and the implementation of this proof-of-concept including class diagrams, sample Semantic Web services, test cases and lessons learned are provided in [225] and are discussed in the paper [8].

6.4.3 Scenario, Validation and Verification

Analog to the previous prototypes a scenario has been performed, in order to assess whether the aimed system requirements of the mediated service composition tool could be met and if they have been implemented correctly. This time a different scenario has been performed. Due to the nature of the tool as a proof-of-concept, it supports only the most relevant functionality for mediated service composition. Therefore, the focus is put on the design of information flow and only basic control flow specifications, namely sequences are supported. In order to cover the previously addressed “Purchase Order Scenario” more control flow functionality would be required such as the definition of loops or the evaluation of conditions. Nevertheless, a more suitable scenario for the assessment of the developed tool has been designed demonstrating a sequential cross-organizational process from the eGovernment domain.

In this eGovernment scenario the process of applying online for a birth certificate has to be implemented in terms of a Web service composition. In the application process various authorities from different domains are involved. The process includes a service for handling the payment of the birth certificate fee, a resident registry service for checking the citizen input for consistence, a vital records office responsible for issuing the birth certificate and a statistical office, to which the vital records office reports its activities.

These services are supposed to provide standard XML-based Web service interfaces including well-defined message sets. In fact, in various countries national interoperability frameworks impose XML Schema and Web service interfaces for exchanging data between administrations to increase interoperability. An example for such a national effort is the Danish eGovernment initiative, which focuses on the definition and promotion of XML domain data structures [248]. A key achievement of the initiative is the “InfoStructureBase” [249], a shared repository for XML-based schemas of eGovernment services. In other countries such as Germany, its federal structure causes a less centralized approach. Nevertheless, there are domain and state-specific initiatives, which work under a shared organizational framework named OSCI-XÖV [250]. However, interoperability is just ensured within domain boundaries, e.g. through OSCI/XMeld (XML Schema exchange standard between registration offices) or OSCI/XJustiz (XML Schema exchange standard for legal authorities). In cross-organizational eGovernment processes (such as addressed in the scenario below) services of various public agencies from different domains are involved. In such scenarios the lack of semantic interoperability across-domains results in enormous integration efforts.

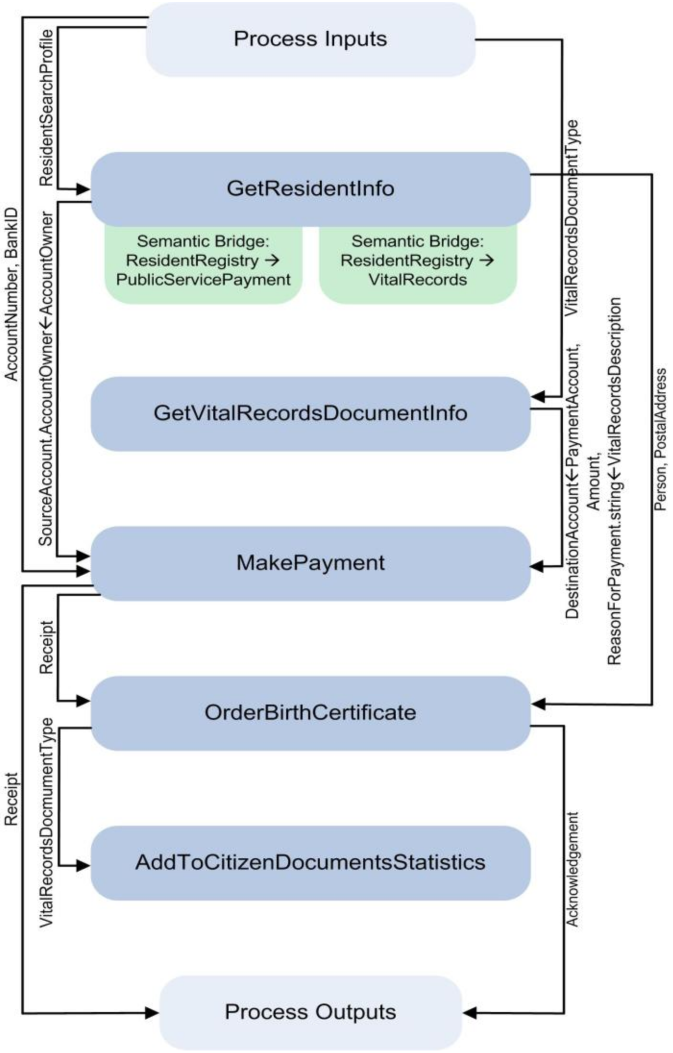


Figure 6-16 eGovernment Scenario for Mediated Service Composition

Accordingly, in the developed scenario different XML Schemas have been applied to describe the WSDL-based Web services. The domain information models have been expressed in terms of ontologies to benefit from the developed semantic mediation mechanism. Therefore, the Web services have been semantically annotated with concepts from the OWL-S upper ontology and the designed domain ontologies. Furthermore, SWRL-based semantic bridges have been developed according to the previous methodology steps. The left hand Figure 6-16 illustrates the applied semantic bridges and the designed information flow of the eGovernment scenario. For instance the domain standard employed by the resident registry uses a different information representation for names and addresses than that used by the vital records office. While in the one domain an address might be a complex type consisting of different attributes for given name, surname, street, street number, etc.; in the other domain standard the address concept might be modeled as a complex type that contains just one single attribute for street and street number all together. According to this process description a composite eGovernment service has been composed as presented in the screenshot Figure 6-15 above. Finally, a BPEL-based orchestration plan has been generated and published in an extended open source-based process execution engine, which is further discussed in the following Section 6.5. More details on the performed scenario including ontology descriptions of the eGovernment domain standards, semantic bridges and test case definitions are provided in [247].

Validation

The overall goal has been to ease the information flow design in Web service compositions spanning heterogeneous conceptualized domains. In particular, five system requirements have been derived, which extent of coverage is discussed in the following.

The basic functionality for the visual design of information flow in Web service compositions has met the expectations, whereas this does not include the provision of complex control flow support. In particular, the expression of information flow, i.e. the assignment of input and output parameters, could be achieved by utilizing description logic based rules. Thus, the whole composition design could remain purely on the conceptual level. Furthermore, the developed semantic mediation mechanism could be successfully integration in the service composition process, while keeping its application transparent to the service composer as described in the scenario. Based on the semantic mediation as well the required matching mechanism for service selection and composition support has been realized. The realization of the final requirement of exporting the designed composite service to a semantically extended BPEL engine has been fulfilled. However, the successful execution of the generated execution plan and its validation remains to be shown in the next section discussing the realization of the mediated process execution tool.

Verification

Analog to the previous verification part several tests have been run to ensure that the prototype mainly performs correctly and stable in the expected behavior. In particular, the WSDL-based Web services representing the eGovernment services have been tested by specifically developed test clients. Similar, Semantic Web service test clients have been developed that call each eGovernment Web service using its OWL-S description and the OWL-S API for direct invocation. Moreover, during the development of the composition tool and execution engine several unit tests have been run to check the correct implementation of the various methods. In order to ensure that the prototype does not only work with the specifically for the scenario designed Semantic Web services, it has also been tested with test-oriented Semantic Web services provided in the Mindswap project [223].

Thus, it can be stated that the prototype for mediated service composition generally performs correctly. Taken as well into account the validation results, it can be concluded that the implemented prototype demonstrates successfully how the mediated service composition step of the semantic mediation methodology can be instantiated as a proof-of-concept.

6.5 Meditated Process Execution Tool

This section is dedicated to the final tool of the semantic mediation toolkit, which instantiates the methodology step of mediated process execution (cf. Section 5.10). In order to close the cycle of the presented methodology, it remains to demonstrate how modeled business processes, which then have been instantiated by a composition of semantically enriched Web services, can be finally executed. The section is structured analog to the previous, starting with a summary of the system requirements, over the design and realization description of the system architecture and its components to its application in terms of a scenario for the tool's validation and verification.

6.5.1 System Requirements and Challenges

Analog to the previous prototype descriptions, the goals of the methodology step of mediated business process execution together with its functional architecture (cf. Section 5.10) are recalled, in order to list the system requirements of the aimed prototype. The general goal has been to incorporate components providing the Semantic Web technology-based functionalities into BPEL process integration middleware (BPEL process engine). Thereby, in particular the reflection of the different meta-data models (ontology vs. XML) on the runtime level is required. The Semantic Web technology-based components to be incorporated into BPEL include the following four aspects:

- Semantics-based information flow – Enable the execution of information flow definitions, which are specified in terms of description logic based rules that work on serializations of OWL ontology concept instances.
- Semantic mediation mechanism – Integrate the developed semantic mediation mechanism to enable the execution of information flow across services with heterogeneous ontological conceptualizations of service parameters.
- Semantic Web service invocation – Remain on the higher conceptual abstraction level to the latest possible point, i.e. breaking down the abstraction level just before service invocation in terms of mapping Semantic Web service calls to traditional WSDL-based Web service calls.
- Semantic Web service interface to the semantically-extended BPEL orchestration – Handle instances of ontology concepts as process inputs and outputs in terms of a Semantic Web service interface on top of the WSDL interface of the BPEL process.

In order to provide a practice-oriented and usable solution, also time consumption for the reasoning during execution of semantic bridges and information flow rules has to be considered, which is often a weakness of Semantic Web technology.

In order to achieve these goals, the functional architecture is based on an extension mechanism for BPEL process engines to incorporate the above summarized semantics-based functionalities. In the following the functional architecture is refined to the system architecture of the mediated process execution engine, whereas each component is described in terms of its concrete technical instantiation.

6.5.2 Design and Realization

Analog to the previous prototype realizations again the customized Agile Unified Process (AUP) has been applied during development. The principle idea of the realization has been to exploit the XML-based RDF-serialization of OWL concept instances (cf. Section 3.3.2). As the information flow needs to be embedded in the BPEL process description, such serializations are stored in BPEL variables, which are designed to hold XML instances. Whenever one of the above outlined semantic enhancements is required to provide processing on the higher abstraction level, a call for external functionality is delegated by the extension evaluator to dedicated external engines performing this task. Such an external engine parses the dedicated BPEL variable and loads the stored information into an OWL model, in order to process the information on the higher semantically expressive abstraction level. Thus, the system architecture follows a straight-forward approach by providing for each semantic extension one dedicated external engine, which are incorporated though a common BPEL extension mechanism based on XPath function calls. Accordingly, the system architecture for the mediated process execution engine is illustrated in the following Figure 6-17:

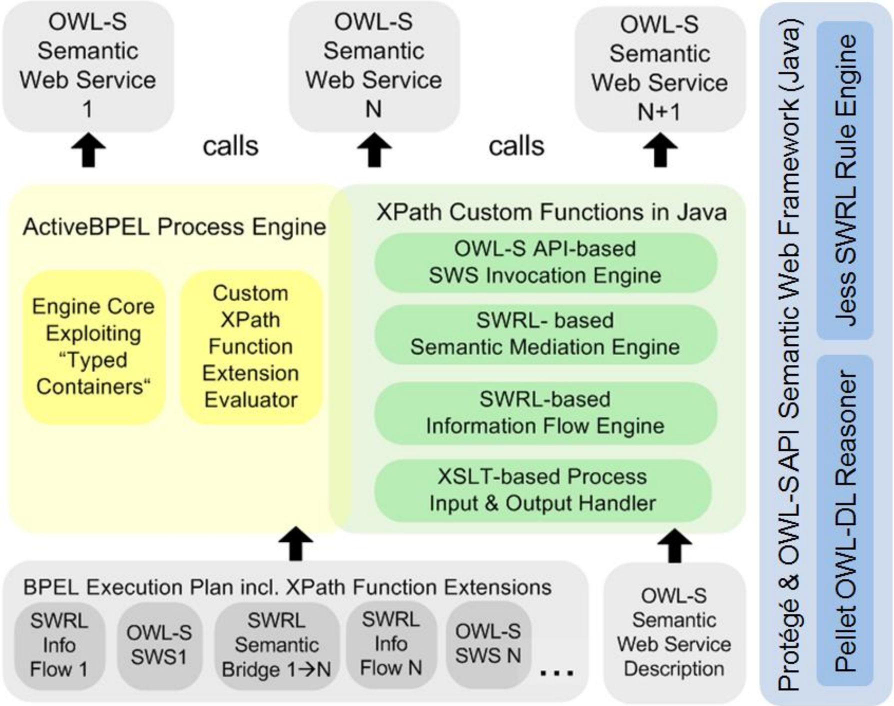
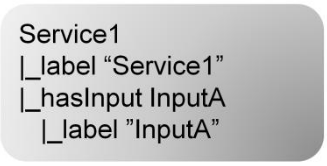


Figure 6-17 System Architecture of Mediated Process Execution Engine

The subsequent paragraphs describe the individual components and explain how they are related to each other:

BPEL Process Engine: The realization of the prototype is based on the existing ActiveBPEL [251] process execution engine. The BPEL-engine requires a Web container to run in. The recommended container for ActiveBPEL is the Apache Tomcat Web Server. In order to enable the aimed mediated process execution, mainly the handling of ontology concept instances in BPEL and a solution for an extension mechanism for the semantic enhancements have to be targeted. The ActiveBPEL engine has been chosen since it is open source and supports defined extension points by implementing so called "custom functions". However, the concept and developments can be applied to any BPEL engine supporting such defined extension points. As an alternative, the first investigated approach for the extension mechanism has been the delegation to external components via Web services using the standard <invoke> activity of the BPEL language. The advantage of this approach is that no modifications or extensions have to be made to a BPEL engine and that all BPEL compliant execution engines could execute the resulting process. But a drawback is the mixing of "functional Web services" that contribute to the business logic with „technical Web services" that would perform supporting tasks such as mediation and information flow. Hence, the integration of semantic bridges, semantic information flow etc. via standard Web service calls has not been chosen as the preferred solution. The more suitable approach is to use the extendibility of XPath to integrate semantic components into BPEL. XPath is supported in BPEL engines to support XML processing, whereas XPath custom functions are designed to externalize certain complex processing tasks to dedicated components. The advantage of this approach is that no modifications to the BPEL engine have to be made since the most popular BPEL engines such as ActiveBPEL, Apache ODE [252] and Oracle BPEL Process Manager [253] support the integration of custom XPath functions ([254], [255], [256]). Additionally, portability issues are minimized, since custom XPath functions can be installed into all above mentioned BPEL engines using standardized installation procedures.

As it has been already outlined above, the handling of ontology concept instances in BPEL is based on their serialization format. Accordingly, the most straight-forward solution is to type all BPEL variables as *xsd:anyType* and to assign OWL individuals in terms of RDF/XML-serializations to them. Moreover, in order to be able to define SWRL information flow rules, that move parameters between service inputs and outputs, it is necessary to prepare the variables. For this purpose so called typed containers define a graph-structure of statements as an anchor to which the actual parameters can be attached. This provides defined access points on which the SWRL rules can be performed. The right hand Figure 6-18 illustrates an empty typed container for a service input.



```
Service1
|_label "Service1"
|_hasInput InputA
   |_label "InputA"
```

Figure 6-18 Typed Container in BPEL Variable

In future it can be assumed, that a semantically enhanced version of BPEL might natively support ontology concepts to be used for typing of variables. Nevertheless, the implementation of a respective semantic BPEL engine is out of scope of this thesis. Rather the aim is to use standard extensions of the BPEL specification to integrate the additional semantic functionality.

In fact, storing OWL individuals in terms of RDF/XML-serializations in BPEL variables goes well along with the decision to use custom XPath functions to incorporate semantic concepts into BPEL. Thus, BPEL variables can be used in BPEL expressions, which are used to call the custom XPath functions. Hence, each semantic component can gain easy access to the required OWL individuals and each semantic component can be realized as a separate Java component integrated via the universal Java XPath engine [257].

Input and Output Handler: A deployed BPEL process is as a whole handled like a traditional Web service, whereas its interface is provided by an XML Schema-based WSDL definition created during deployment. Thus, the XSD instances of the process input have to be lifted onto the ontology level. As the lifting and lowering mechanism based on XSL transformations is well addressed in the OWL-S API providing the grounding (lifting and lowering) for Semantic Web service calls, the same functionality is reused for the process input lifting and process output lowering. The process inputs and analog the process outputs are passed to an external Java component based on the OWL-S API via the XPath custom function mechanism.

Information Flow Engine: Having lifted the abstraction level of the service parameter representation, the information flow rules can be applied, which assign the process input parts to the respective service inputs and between service outputs to following service inputs within the orchestration. It is important to note that the information flow rules have already been integrated into the generated BPEL process description during mediated service composition based on the information flow assignments. Consequently, the information flow rules are embedded within the BPEL process storing them in terms of their RDF/XML serialization in variables. The execution of a SWRL rule-based information flow is again performed as an XPath custom function call in a BPEL assignment expression. The engine's internal expression evaluator determines the responsible external Java component and delegates the function call together with the variables holding the SWRL rules, the source parameters and the target parameter.

Within the external component the RDF/XML-serialized service parameters including the anchor structure (cf. Figure 6-18) are loaded into a new Protégé-OWL model allowing further semantic processing. Internally, all service inputs and outputs are handled in different OWL models but are merged temporarily to apply the SWRL rules and thus perform the information flow between services. The application of SWRL rules for the information flow is realized with a similar mechanism as already explained for the semantic mediation engine (cf. Section 6.2.2). Finally, the resulting OWL model is reduced to contain only the statements belonging to the respective service input. Finally, the OWL model can be serialized to its RDF/XML representation before writing it back to the BPEL process into a variable holding the inputs for the targeted service.

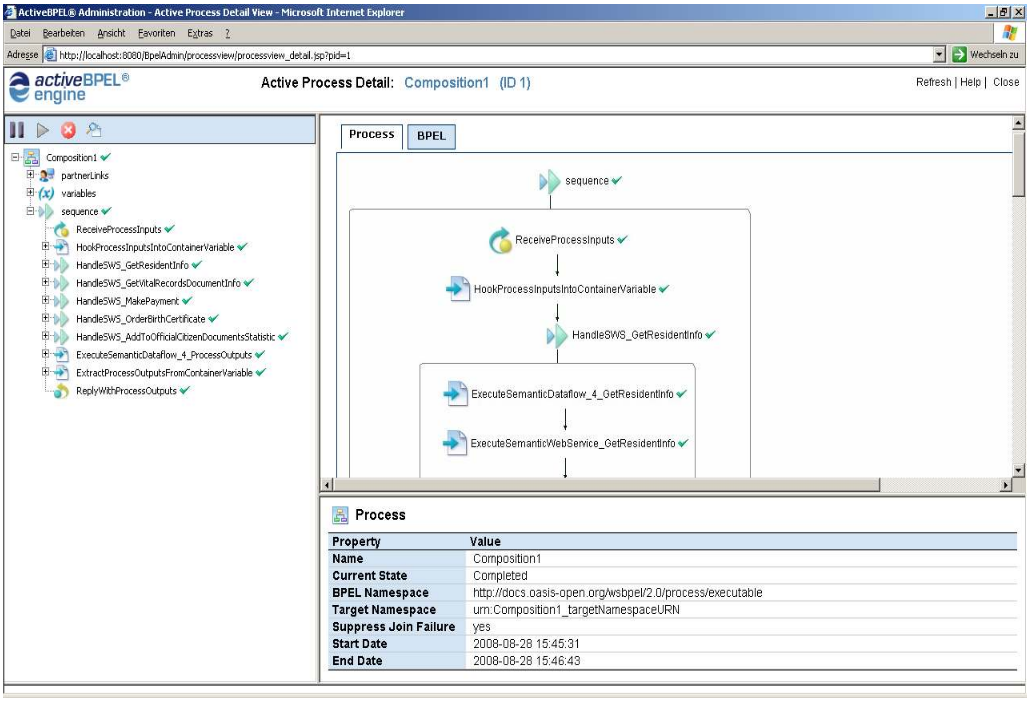
Semantic Mediation Engine: In order to execute the mediation between the heterogeneous conceptualizations used for the description of service parameters, the application of semantic bridges is delegated to the external semantic mediation engine. Therefore, the SWRL rules for the semantic mediation are embedded into the BPEL process in an analog manner as described for the information flow rules above. Again, the XPath custom functions are exploited to delegate to an external Java component implementing the semantic mediation engine. It is realized by means of a combination of selected Semantic Web frameworks based on the same semantic mediation mechanism and its implementation as described in the previous prototypes (cf. Section 6.2.2).

In this context of process execution, it should be noted that the current OWL-DL semantics bring one particular shortcoming for the application to Semantic Web services. The OWL-DL semantics are based on the open-world assumption (cf. Section 3.3.3) as the language is designed for the World Wide Web. However, in the context of Semantic Web service composition all information required for reasoning are available during processing time. Accordingly, closed-world semantics would be more suitable. For example if a class C is defined as the set of individuals having exactly one certain property p in terms of *minCardinality = 1* and *maxCardinality = 1*, then due to open-world semantics an individual featuring this certain property p cannot be classified to class C . The open-world semantics-based reasoner does not know if there maybe exists a second occurrence of p for this individual, so that the restriction would become unsatisfied. This behavior was not expected but has been detected during the implementation phase of the prototypes. However, it is important to express exact cardinality constraints on service parameter properties to correspond to the constraints of type definitions in XML Schema domain standards describing WSDL-based Web services. In order to overcome this weakness, the feature of functional properties within OWL-DL is exploited. By declaring the property p as functional the reasoner knows that p exists exactly once. Thus, even with open-world semantics the individual (service parameter) can be classified to class C. Hence, basic cardinality restrictions are supported and the facet analysis classification applied in the semantic mediation mechanism reflects the requirements which originate from the targeted mapping to underlying traditional XML Schema-based Web services.

Semantic Web Service Invocation Engine: After the representations of service parameters have been mediated by semantic bridges and all required input parameters have been assigned according to the defined information flow, a Semantic Web service can be invoked. Analog to the previous components, the Semantic Web service call is delegated via the XPath extension mechanism to an external Java component which instantiates the Semantic Web service invocation engine. It applies the OWL-S API to get programmatic access to read and execute OWL-S service descriptions. Internally, service specific XSL transformations have to be performed, in order to ground the Semantic Web service to an XML Schema-based Web service (cf. Section 5.8.2). In this process, it has to be taken into account that XSLT does not work on the same meta-data model as OWL and thus can only operate on the OWL serializations. However, the natively provided RDF/XML-ABBREV serialization by the OWL-S API implementation does not allow to exploit the full potential of polymorphism as necessary for further processing of service parameter with semantic bridges. When a polymorph individual is serialized using the RDF/XML-ABBREV format, one of the types it holds is non-deterministically selected and the last fragment of the types URI is taken for the opening tag for this individual's XML serialization. The other types are expressed by separate <rdf:type.../> sub elements. This varying structure complicates the development of the service specific XSLT code dramatically. To overcome this weakness, the OWL-S API, which is provided as on open source project, was adjusted. Now internally the basic RDF/XML serialization is applied. This means that all types are represented equally as sub-elements, which allows to define straighter XSL transformations. Hence, the mapping from polymorph OWL serializations to single typed XML Schema instances can be achieved in terms of XSLT rules that match exactly the type which has been defined in the OWL-S input description. The processing of the Web service results is less complicated, as the received message parts correspond to the data model XSLT was designed for. XSLT rules can easily match the XML Schema instances and fill predefined skeletons of serializations of OWL individuals.

It has turned out, that the application of XSL transformations for Semantic Web service groundings as intended in OWL-S provides only a sub-optimal solution for bridging the gap between the abstraction levels of the meta-data models of Semantic Web services and WSDL-based Web services. To overcome the complicated XSLT processing on OWL serializations, a more powerful language should be applied. This language should be similar to XSLT in terms of providing means to produce XML Schema based instances but it should as well be able to understand and work on the triple model of OWL. However, as long as such a language is not established and well supported (cf. Section 3.4.3) the presented mechanism provides a solid solution based on widely accepted and standardized languages.

The four above discussed semantic extensions can be found in the following Figure 6-19 showing a process execution report of the extended ActiveBPEL engine. It illustrates the execution of a semantically enriched BPEL process, which has been generated by the mediated service composition tool according to the eGovernment scenario discussed in Section 6.4.3. The semantic extensions for process input handling, semantic information flow and invocation of Semantic Web services can be seen in the process graph shown in the center:



Property	Value
Name	Composition1
Current State	Completed
BPEL Namespace	http://docs.oasis-open.org/wsbpel/2.0/process/executable
Target Namespace	urn:Composition1_targetNamespaceURN
Suppress Join Failure	yes
Start Date	2008-08-28 15:45:31
End Date	2008-08-28 15:46:43

Figure 6-19 Mediated Process Execution in Semantically Enhanced Process Engine

More details on the design, system architecture and the implementation of this proof-of-concept including class diagrams, further samples of semantically enriched BPEL processes, test cases and lessons learned are provided in [225] and are discussed in the paper [185].

6.5.3 Scenario, Validation and Verification

Analog to the previous prototypes, a scenario has been performed to assess whether the aimed system requirements of the mediated process execution engine could be met and if they have been implemented correctly. This time again the “Purchase Order Mediation Scenario” already introduced and modeled in context of the mediated business process modeling prototype (cf. Section 6.2.3) and the semantic bridge testing prototype (cf. Section 6.3.3) has been exploited. Having the capabilities of the mediated process execution engine at hand, the full mediation scenario has been as well performed on the runtime level.

Besides applying semantic bridges to mediate between different information models used by the interacting systems Blue and Moon, the control flow capabilities of BPEL are used to align as well their heterogeneous interaction patterns. I.e. the granularity and order of operations of the two systems in the scenario differs. For example the Blue system issues a RosettaNet-based purchase order message containing all relevant information such as contact information for the shipping as well as requested items and their requested quantities in one block. On the other side, the Moon system expects this information to be sent separately: First the customer information, then a general order request and - upon acceptance of the order in general - each ordered item one by one. The following Figure 6-20 Purchase Order Mediation Scenario shows the full mediation scenario including the interaction patterns. Furthermore, it highlights where the developed semantic extensions address the key challenges of the scenario:

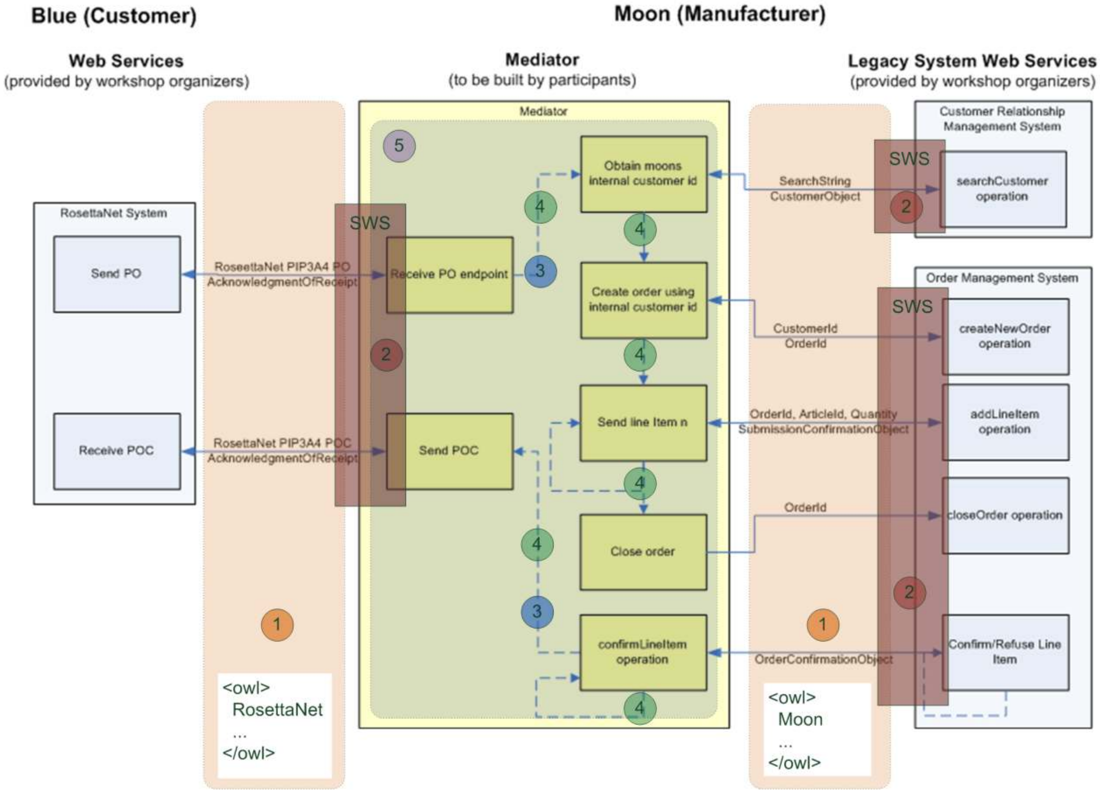


Figure 6-20 Purchase Order Mediation Scenario and Semantic Extensions

The following paragraphs discuss the semantic extensions (1) – (5). This includes as well references to the previously presented prototypes, as finally they all together contribute to the solution of the scenario.

(1) Information Model Shift to Domain Ontologies – In order to shift the different information models to a higher abstraction level and address semantic interoperability on the more adequate conceptual level for this challenge, the two XML Schema-based information models (RosettaNet and Moon) have been lifted to domain ontologies. This part of the scenario solution has been already discussed in detail in context of the semantic bridges testing prototype and its application to the scenario (cf. Section 6.3.2).

(2) Semantically enriched Web Services – By using the technologies described in Section 5.8.2 the WSDL-based Web services provided in the scenario are annotated with concepts from the developed domain ontologies, in order to get OWL-S Semantic Web services. For instance, in the scenario a Semantic Web service for the Moon CRM has been developed, which expects the defined class *Customer* which – among others – defines the property *hasBusinessName*. This property is used as the search criteria for the customer lookup. If a customer with the given name is found, an *IdentifiedCustomer* OWL individual is returned containing all customer attributes supplied by the CRM system. The corresponding domain ontology containing these concepts has been already discussed in Section 6.3.3. Furthermore, OWL-S-based services for creating an order and filling it stepwise with items have been deployed on top of the existing Web services. Finally, the lifting and lowering definitions for converting the incoming and outgoing XSD instances to OWL instances have been defined.

(3) Semantic Mediation – Dedicated semantic bridges for the scenario have been defined and tested (cf. Section 6.3.3). Based on the semantic bridges and the externalized semantic mediation engine called from the BPEL process, the heterogeneous conceptualizations could be aligned and prepared for sound information flow.

(4) Information Flow on Conceptual Level – In order to express the information flow on the ontology-based conceptual level within the BPEL process, as well SWRL rules are used similarly to the approach applied for semantic bridges (cf. Section 6.5.2). For this scenario the information flow rules have been defined manually. They move the necessary input parameters to the respective variables reserved for each service in the BPEL process. For instance, the overall process input message contains a *Partner* instance as part of a *PurchaseOrder* instance. After applying a semantic bridge this *Partner* instance is polymorph and as well classified as a *Customer* according to the concept definition in the Moon ontology. In order to prepare the service call which registers the *Customer* in the Moon CRM system, the *Customer* instance is separated from the variable storing the overall *PurchaseOrder* instance by applying an information flow rule that transfers the *Customer* instance into the designated input variable for the CRM service. In an analog manner the other service input variables are prepared by applying accordingly defined information flow rules.

(5) BPEL Process for Mediating Interaction Patterns – The BPEL process can be seen as the glue which holds the before discussed components together. Furthermore, as already outlined above, the BPEL process is utilized to harmonize the interaction patterns, i.e. the different granularity levels in service calls of both systems. In the scenario the *OrderItems* provided by the Blue system are aggregated in a single incoming call. However, the Moon system requires fine granular calls of the *AddLineItem* Semantic Web service, i.e. one service call for each single LineItem. Therefore, a for-each loop available as a BPEL language construct is used to split of the aggregated *OrderItem*s and invoke the

AddLineItem Semantic Web service one by one. For this scenario the BPEL process has been designed using the ActiveBPEL process designer. However, the consistent usage of the ontology level provides the foundation for further semi-automatic tool support in the design phase by means of matching functionalities etc. as presented with the mediated service composer prototype in Section 6.4.3. However, for the composer prototype it has been out of scope to develop comprehensive support for control flow including loops and conditions which would be necessary to completely design the purchase-order scenario.

Validation

The overall goal has been to incorporate the semantic mediation mechanism into the industry standard BPEL and corresponding execution engines, in order to demonstrate the practicability of the developed approach. The four aimed semantic enhancements as outlined in the requirements Section 6.5.1 could be integrated in a BPEL process engine making use of the XPath custom function mechanism and the RDF/XML serialization format of OWL instances. Semantics-based information flow could be realized by exploiting SWRL-based rules that work consistently on the higher conceptual abstraction level. Furthermore, the semantic mediation mechanism as well utilized in the previous prototypes has been successfully integrated. The BPEL process engine could be enabled to invoke OWL-S based Semantic Web services by customizing the OWL-S API and grounding WSDL-based Web services with XSL transformations. Finally, by exploiting the same technologies as used for the grounding in a reverse manner, as well a Semantic Web service interface to the semantically-extended BPEL process could be provided.

Regarding time consumption the following runtime measurement has been performed during scenario execution: On a test platform based on a VMware image running on a Intel Core Duo 1.33 GHz CPU with 2 GB RAM the execution of the whole scenario including repeated classification, semantic bridging, information flow inference and Semantic Web service invocations takes less than 5 seconds on the server side, where the execution engine runs, and around 11 seconds including the time consumption for the simulation of the Blue system calling the deployed process. This good performance could be achieved by avoiding reasoning processes on big data sets including all involved statements of service parameters, semantic bridges and information flow rules all at once. Rather, reasoning has been applied in a very focused manner by means of individual reasoning tasks concentrating just on particular variables within the BPEL process that contain only service parameters of one service or just one semantic bridge etc. Thus, the overall time consumption can be regarded as reasonable.

Verification

Consistently to the verification measures applied to the previous prototypes, several tests guided by the scenario have been run to ensure that the prototype mainly performs correctly and stable in the expected behavior. Besides the above presented "Purchase Order Scenario" as well the service composition from the eGovernment scenario presented in Section 6.4.3 has been deployed and successfully executed in the extended ActiveBPEL engine as highlighted in Figure 6-19. Additionally, during the development of the composition tool and execution engine several unit tests have been run to check the correct implementation of the various methods.

Thus, it can be stated that the prototype for mediated process execution generally performs correctly. Taken as well into account the validation results, it can be concluded that the implemented prototype demonstrates successfully how the mediated process execution step of the semantic mediation methodology can be instantiated as a proof-of-concept.

6.6 Usage and Extension of the Semantic Mediation Toolkit

For further extension of the prototypes the source code of the toolkit is provided in four open source projects under the GNU General Public License (GPL). The projects are hosted on the BerliOS platform [258] of the Fraunhofer Institute for Open Communication Systems:

- the Mediated Business Process Modeling Tool is available at [259];
- the Semantic Bridge Testing Tool is available at [260];
- the Mediated Service Composition Tool is available at [261];
- and the Mediated Process Execution Tool is available at [262].

The source code is documented using Javadoc. Furthermore, the above listed project locations provide documentations for each prototype including detailed installation guides and usage descriptions with stepwise walkthroughs for the above presented scenarios.

6.7 Summary and Reflection

The chapter has presented the developed semantic mediation toolkit. The principle idea has been to provide prototypical instantiations of certain key steps of the semantic mediation methodology discussed in the previous Chapter 5. Some steps of the methodology could be well covered with existing work and respective tools. Therefore, it has been abstained from implementing such functionalities redundantly. Rather, it has been focused on these methodology steps which cannot be performed adequately with existing work. Therefore, the methodology steps of mediated business process modeling, semantic bridge testing, mediated service composition and process execution have been addressed with newly develop prototypes. They complete the semantic mediation toolkit along with the already existing tools for domain ontology development, semantic bridge definition and semantic service enrichment.

The chapter has started by discussing the realization of the mediated business process modeling prototype. Firstly, the goals for mediated business process modeling from the semantic mediation methodology have been briefly recalled and refined to four system requirements. Subsequently, the design and realization has been described, whereas the tool's system architecture has been presented and its building blocks have been explained. The tool has been described as an extension of a client-server based business process modeling tool. Therein the developed semantic mediation mechanism has been integrated by applying Semantic Web technology frameworks including an ontology reasoner and a rule engine working on W3C standardized description logic languages. In particular, the implementation of the semantic mediation mechanism, i.e. the realization and processing of semantic bridges, has been discussed in detail. Finally, in order to validate and verify the implemented prototype a scenario has been performed, namely the "Purchase Order Mediation Scenario" defined in context of the international Semantic Web Service Challenge. Therefore, the business process between two heterogeneous systems - Blue and Moon - and corresponding information flow across heterogeneous conceptualizations has been modeled with the realized prototype. Consequently, the system requirements have been mapped to the results of the performed scenario and it has been concluded that the envisioned goals could be demonstrated within the scope of a proof-of-concept. The essence of the concept and the prototypical implementation of the mediated business process modeler applied to the “Purchase Order Scenario” have been published in [245].

The second section has been dedicated to the realization of the semantic bridge testing prototype. As the underlying ontology mapping of a semantic bridge is a complex and error-prone process, users and developers need to be enabled to systematically test and determine the quality of semantic bridges. Therefore, the core system requirements have been derived and listed including: functionality for semantic bridge test case definition, functionality for test case execution and reporting functionality. The functional architecture discussed in Section 5.7.2 has been refined to a concrete system architecture following the model-view-control (MVC) pattern. The realization of the individual components has been explained, especially the calculation of test case coverage in order to quantify the test result. The validation and verification of the prototype has been examined by applying the testing tool to the semantic bridges developed for the “Purchase Order Scenario” discussed above. Thereby, as well the development of the domain ontologies “Blue” and “Moon” have been described. As the targeted goals for semantic bridge testing could be met, it can be stated that the purpose of a proof-of-concept implementation is fulfilled.

The next described prototypical instantiation of the semantic mediation toolkit has targeted the methodology step of mediated service composition. The purpose of the composition tool is to demonstrate how the task of integrating Web services described by heterogeneous ontologies can be facilitated by semantic matchmaking and incorporation of semantic bridges, which are expressed in the Semantic Web Rule Language (SWRL). The tool is capable of loading and composing sequences of Web services that are semantically described by means of the OWL-S standard. The output is a semantically extended BPEL process orchestration plan, which contains the sequence-based control flow and rule-based information flow between the Semantic Web services. The section has started with a recall and refinement of system requirements followed by a description of the design and realization of the tool in terms of a stepwise elaborated system architecture. The evaluation of the prototype was again performed in terms of a scenario, however this time a cross-organizational eGovernment scenario has been applied. According to the validation and verification results, it can be concluded that the implemented prototype provides an adequate proof-of-concept for mediated service composition. The essence of the concept and prototypical implementation of the Semantic Web service composer applied to the eGovernment scenario has been published in [8].

Finally, the realization of the mediated process execution engine has been discussed. The developed prototype is based on the existing ActiveBPEL process execution engine. The ActiveBPEL engine was chosen, since it is open source and supports defined extension points by implementing so called “custom functions”. However, the concept and developments can be applied to any BPEL engine supporting this standardized extension mechanism. Firstly, the system requirements have been specified followed by a detailed discussion of the derived system architecture. This includes the semantic extensions for the BPEL engine and particularly the development of components for executing semantic information flows and Semantic Web services. With the capabilities for mediated process execution on hand, the complete “Purchase Order Mediation Scenario” could be performed. Thereby, the whole process execution remains on the ontology-based conceptual level and is just broken down at the latest possible point in terms of grounding OWL-S Semantic Web service calls to traditional WSDL-based Web services. According to the successful scenario execution, the validation and verification of the implemented prototype can be considered as a valid proof-of-concept for the mediated process execution approach. The essence of the concept and prototypical implementation of the mediated process execution engine applied to the “Purchase Order Scenario” has been published in [185].

At the end of the chapter, a brief section has presented information about the extension and usage of the four developed prototypes, including references to a Web-based development platform where the corresponding open source projects are hosted and from where the realized semantic mediation toolkit can be downloaded.

Chapter 7

Evaluation and Case Study of an Exemplary Distributed Organization

This chapter targets the evaluation of the presented work. Having discussed the validation and verification of the semantic mediation toolkit with regard to the system requirements of the respective prototypes in the previous chapter, this chapter aims at evaluating the overall approach of the developed concept for semantic mediation between loosely coupled information models in SOA. The chapter starts by presenting the evaluation methodology, which focuses on a qualitative approach based on a case study of an exemplary distributed organization. Accordingly, the subsequent section discusses this case study, which outlines how the German Chambers of Industry and Commerce have addressed the challenge of internal and external semantic interoperability. In particular, it is analyzed how the in this work proposed solution for semantic mediation and its corresponding artifacts (models, methodologies, prototypes) can optimize effectiveness and efficiency in this process. Based on this analysis, the coverage of goals and the confirmation of the research hypothesis are discussed, whereas the main idea is to map the outcome of the case study to the originally set conceptual goals and the claims of the research hypothesis. Finally, a summary and reflection of the evaluation chapter is provided.

7.1 Evaluation Methodology

The evaluation is performed in two steps: Firstly, the semantic interoperability activities of the German Chambers of Industry and Commerce are compared in terms of a gap analysis with the developed approach for semantic mediation. Thus, the comparison is used to outline the potential of the developed concept, methodology and toolkit. Secondly, the outcome of this analysis is mapped to the originally set conceptual goals from Section 4.2 and to the claims of the research hypothesis in Section 1.3.1.

In order to fit for the evaluation, the case study has to target a large-scale SOA landscape which covers multiple organizations with independent IT management. Taken this into account, the exemplary character of the German Chambers of Industry and Commerce is given, because firstly: it is a decentralized organization; and secondly: it has rolled out an organization-wide SOA covering 80 institutionally-independent chambers, which are served by four different IT service providers. Moreover, the Chambers are core actors in cross-organizational eGovernment processes. For instance, the administrative process for the registration of a new business involves besides the Chambers various further public administrations including the trading supervision department, statistical offices and fiscal authorities etc. depending on the business case. As the IT-based communication and interaction between these various involved institutions is iteratively migrated to service-oriented approaches, the case study of the Chambers provides as well the required attribute of a large-scale SOA landscape.

However, due to the prototypical nature of the semantic mediation toolkit, it is out of scope of this evaluation to apply the toolkit directly to operational processes of the Chambers and its business partners. Rather, a qualitative analysis is performed comparing the Chambers achievements and challenges with the currently applied practices and technologies for achieving semantic interoperability with the in this research presented approach. The case study has been carried out in context of a research transfer project from 2007 to 2010 of the Fraunhofer Institute FOKUS supporting the introduction of a service-oriented architecture to the German Chamber of Industry and Commerce.

Based on the outcomes of the case study the fulfillment of the conceptual goals and claims in the research hypothesis are assessed. According to the applied research methodology of design research, the main artifacts developed in this work are recalled to structure the evaluation. They address the core steps of design research: awareness of a problem, conceptual suggestion and development (cf. Section 1.3.1). The corresponding main artifacts are:

- Framework for Semantic Interoperability in SOA (cf. Section 2.5)
- Concept for Semantic Mediation between Loosely Coupled Information Models (cf. Section 4.8)
- Semantic Mediation Methodology and Toolkit (cf. Section 5.3 and 6.1)

Furthermore, general implications and derived conditions are discussed, which relate to the fulfillment of the conceptual goals and the research hypothesis.

7.2 The German Chambers of Commerce and its eGovernment Context

In Germany the Chambers of Industry and Commerce [263] are a public institution with self-administration under the inspectorate of the state ministry of economy. All German companies registered in Germany, with the exception of handicraft businesses, the free professions and farms, are required by law to join a chamber. Accordingly, the Chambers represent approx. 3.6 million companies. The main duties of the Chambers of Industry and Commerce are [263]:

- Sovereign functions entrusted to business by government to perform upon its own responsibility
- Services to member companies
- Representation of companies in political decision making

The German Chambers of Industry and Commerce are a decentralized organization with 80 chambers across Germany. The Chambers are legally independent entities with one central umbrella association DIHK [263]. The decentralized character of the Chambers organization has direct implications on their IT landscape. The 80 chambers have four different general IT service providers leading to redundant IT applications for major Chamber business processes with partially decentralized hosted instantiations and data storage. Besides this geographical fragmentation along chamber districts as well fragmentation along the business departments has been part of the “as-is” state of the Chambers IT landscape. Below Figure 7-1 illustrates the historically grown isolation of IT applications across the major business departments. Most IT projects and applications have been isolated across business departments, whereas only basic IT services such as intra- and internet, email, etc. have been provided as shared services. These redundancies in IT project management, development, operation and maintenance have caused low cost-effectiveness and significant integration problems with external partners due to the internal heterogeneity.

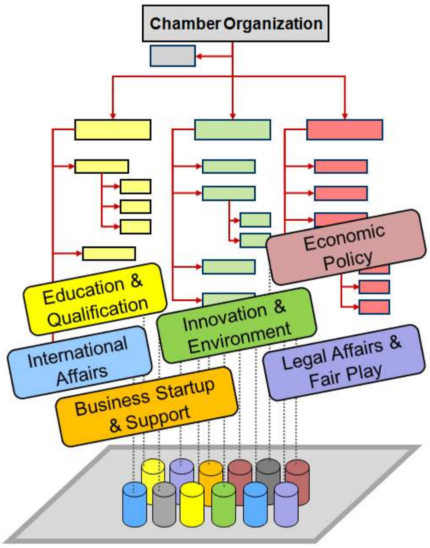


Figure 7-1 Task-oriented Isolated IT Applications

As this is a typical situation for a large distributed organization, where local entities mainly operate within their own scope of responsibility, the heterogeneous and isolated IT landscape can be explained historically. However, the Chambers organization as a whole had to meet new requirements posed by European legislation requiring certain organization-wide IT applications. For example, according to its main duty to provide sovereign functions, the Chambers organization was entrusted to provide a central Web-based registry for insurance broker, i.e. agents who want to sell insurance contracts to private clients. The background of this European consumer protection legislation is that it is sometimes difficult for customers to know if an insurance broker is serious and to know if there have been any legal complaints regarding him in the past. Besides this registry for insurance broker of the legal affairs and fair play department a set of further organization-wide registries have to be provided. This includes for instance a registry for companies to register their amount of produced packaging material from the innovation and environment department. Another example is a Germany-wide apprenticeships registry, in order to face demographic change and provide companies a focused channel to qualified trainees. This registry is not motivated by any legislation but corresponds to the second duty of the Chambers organization to provide their member companies with attractive services.

It turned out that the status-quo situation of geographical and departmental IT fragmentation could not meet the new requirements for cross-organizational eGovernment projects and consequently an organization-wide IT integration approach was targeted. In order to anticipate the legally-bound decentralized structure of the Chambers organization, an SOA-based integration infrastructure has been favored to enable eGovernment processes across Chamber districts and departments.

7.2.1 The Chambers Service Bus and Service Hub

The introduced service-oriented architecture consists of two main infrastructures:

- The Chambers Service Bus for internal process-oriented IT integration within the chambers domain
- The Chambers Service Hub acts as a single point of contact for secured IT-based communication with external partners

The following Figure 7-2 provides an overview of the Chambers SOA-based integration infrastructures:

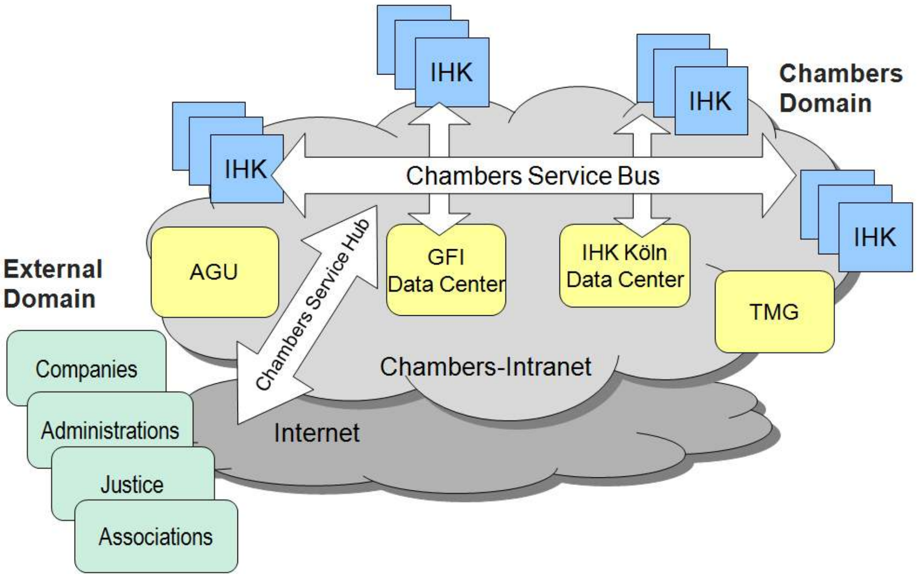


Figure 7-2 Overview of Chambers Service-Oriented Architecture

The Chambers Service Bus connects 80 local chambers (IHK), which are served by four main IT service providers (AGU, GFI, IHK Köln and TMG). The IT applications are partly hosted in external data centers (GFI data center and IHK Köln data center) or the IT service providers host the IT applications locally at the individual chamber. Despite this decentralization on the application level the Chambers share a common intranet, which ensures a secure and trusted communication layer within the Chambers domain. Of major interest are the so called master data systems, which provide customer relationship management such as functionality to process the information of member companies of the local chambers. These master data systems provide the foundation for most additional IT applications of each chamber. Accordingly, in order to enable cross-organizational integrated processes, the various decentralized master data systems had to be enriched with Web service adapters to be connected to the Chambers Service Bus. The Chambers Service Bus has been realized by means of an industry product-based enterprise service bus [264]. Furthermore, the Chamber Service Bus contains a BPEL-process engine (cf. Section 3.3.2) that enables the processing of Web service orchestrations between decentralized IT applications that materialize chamber-wide process chains.

The Chambers Service Hub provides a single point of contact for the integration with external eGovernment partners such as administrations, associations, justice or companies. As this communication takes place over the public internet, the Chambers Service Hub is based on a secured OSCI intermediary [265], which provides a German eGovernment specific transport protocol stack for secure communication. Incoming service messages are received over the Chambers Service Hub and are then further routed over the Chambers Service Bus to the particular chamber and vice versa for outgoing service communications.

In order to establish the SOA of the Chambers, many technical and organizational challenges had and still have to be addressed. For instance, the IT service providers have to adopt the still relatively new Web service technology. Furthermore, their different business models have to be aligned to ensure sustainable competition while keeping the required cooperation. Besides the technical and organizational dimension, as well semantic challenges have to be addressed. As this work focuses on semantic interoperability, the following section presents how the Chambers organization has targeted the aim for seamless cross-organizational process integration despite the fact that the different IT service providers and external partners use heterogeneous information models for their applications and the communication between them.

7.2.2 The Data Conference Working Group

One major challenge for achieving semantic interoperability across the Chambers organization is the fact that the communication partners are autonomous with no legally binding management authority. Therefore, any commitment depends on consensus and collaboration. However, a properly functioning SOA requires well defined service interfaces not only on the technical, protocol and syntactic level but also on the semantic level.

In order to achieve semantic interoperability between the different IT applications connected via the Chambers Service Bus, the chambers central umbrella association DIHK has established the so called data conference working group as part of an established eGovernment project office. The aim of the data conference has been to develop a process-oriented chambers-wide XML-based data exchange standard, whereby all four IT service providers have been involved supported by consultation and moderation of the Fraunhofer Institute FOKUS. The idea has been to focus on a Chambers internal XML standard for message exchange instead of enforcing one consistent information model to be followed by all applications. The advantage of this approach is that the existing internal data structures of the different IT applications of the four IT service providers can remain unchanged. They just have to provide Web service adapters providing interfaces according to the new established message exchange standard to communicate across chamber borders.

Consequently, the data conference working group has been defined as an organizational entity with a methodology and toolbox for the development and evolution of cross-chamber data exchange standards named XIHK. The name XIHK is aligned to the name of German eGovernment exchange standards XÖV. The relation between XIHK and the XÖV standards will be discussed later on in Section 7.2.3. The major task of the data conference is to support demand-driven cross-chamber IT projects with expertise in data modeling on the one hand and ensuring the incorporation of existing XIHK data models within new IT projects on the other hand. The following Figure 7-3 illustrates the basic idea of the data conference working group:

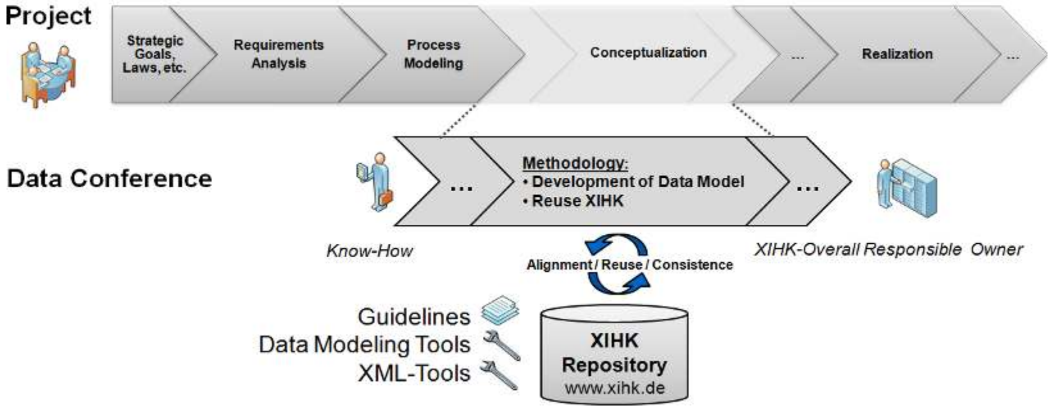


Figure 7-3 General Approach of Data Conference Working Group

The data conference works demand-driven, i.e. its expertise is called from a concrete eGovernment project of the chambers organization (e.g. the provision of a new cross-chamber register). After the first project phases of the new project are finished including requirements analysis and process modeling, the data conference works on the required information models in parallel to the conceptualization phase of the initiating project. According to the identified requirements and derived from the process models, required information or data entities are identified. In order to foster reuse of existing XIHK data models, which are stored and managed centrally in the so called XIHK repository, a designated XIHK overall responsible owner has been established who has been selected among the IT service providers. In order to work in a consistent way, the data conference applies a defined set of guidelines, data modeling tools and XML tools. The general methodology describing how these assets come into play is presented in the following Figure 7-4 and briefly explained in the subsequent paragraphs. Furthermore, the essence of the methodology of the chambers data conference has been published in [188].

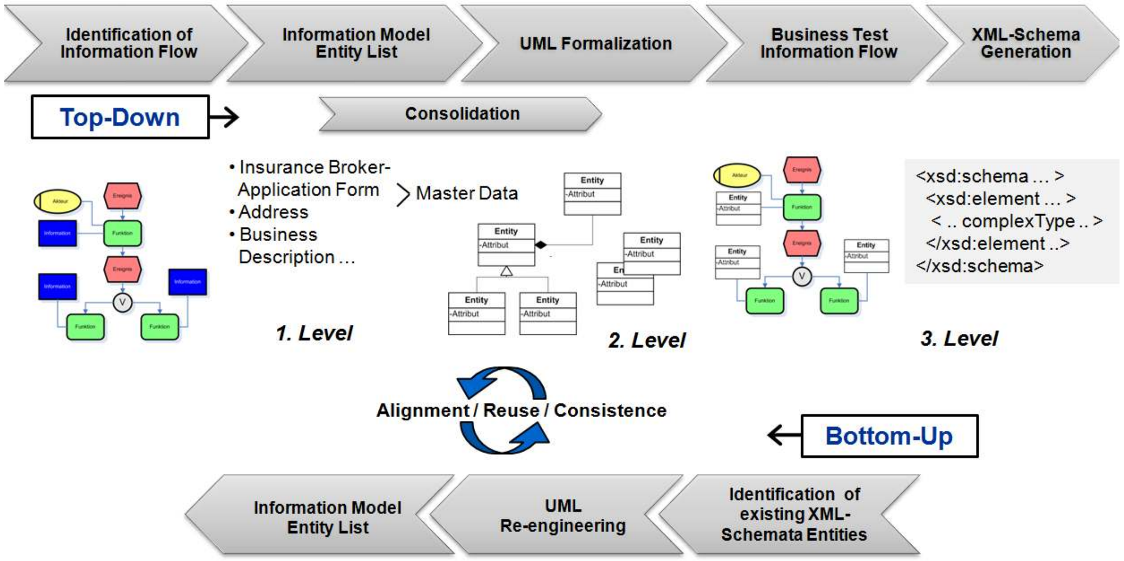


Figure 7-4 Data Conference Methodology

The data conference methodology follows a combined top-down and bottom-up approach. The idea is to derive the requirements of information models for new individual IT applications and services from their process models and align them to existing information models. In the first step of the top-down stream, an information model entity list is derived from a process model to be used by business experts responsible for the concrete IT project. In a second step, the information entities are consolidated and then are formalized in UML entity class diagrams. In order to ensure that the formalized information entities are suitable and complete for the desired IT application, a so called business test of the information flow is performed. I.e. the corresponding information flows in the process models are traversed by using the defined UML information entities. In the third and final step of the top-down stream the concrete XML schemas are generated based on the UML models.

Reuse and overall consistence of the XIHK information models is ensured by means of the bottom-up stream of the data conference methodology. In a first bottom-up step existing schemas of major IT applications are formulated as XML Schema entities, in order to be available for the alignment with the new requirements. For this purpose additionally a UML view and entity list view on the existing information entities is provided. Thus, as well the business expert with a non-technical background is enabled to identify existing information entities for reuse.

All three views or abstractions levels (entity list, UML and XML) of the information models for message exchange are stored in the chamber-wide XIHK repository. Furthermore, a lifecycle-model for the information models along with a versioning framework has been developed, in order to ensure sound evolution of the internal XIHK message exchange standard. This includes as well service models and process models and is part of an overall SOA Governance activity of the Chambers organization. However, due to the focus on semantic interoperability within this case study, these aspects are not further elaborated in this work.

7.2.3 Achievements and Ongoing Challenges

Based on the data conference methodology and its application to a first set of cross-chamber IT projects such as the above discussed eGovernment register, a chamber-wide XML-based data exchange standard could be established. Similar to the in this work developed framework for semantic interoperability (cf. Section 2.5) the XIHK information model reflects different abstraction levels for its representation. Even though the Chambers approach does not utilize ontologies for the conceptual level, the combination of entity lists and UML models allow non-technical business experts to understand the information models and use them in the process analysis and conceptualization of new IT applications.

Furthermore, a general goal targeted with the SOA introduction to follow a more process-oriented approach in the design and realization of IT applications has been anticipated in the data conference. The top-down and bottom-up approach starts with process models to identify required information entities and thus provides a starting point for improved alignment between business and IT perspectives.

However, it turned out that in practice only incomplete representations on the three abstraction levels of the information models have been maintained, due to shortcomings of the applied technologies which require significant manual development efforts. In consequence, mostly only the XML Schema representation exists and is continuously maintained. This leads to inconsistencies between the different abstraction levels and thus limits the aimed advantages for business IT alignment.

The overall concept of loose coupling on the semantic level (cf. Section 4.6) has been reflected insofar, that the developed Chamber-wide message exchange standard XIHK remains independent from the internal information models within the decentralized IT applications. Each IT service provider provides XIHK-conform service adapters and performs the semantic mapping to its internal representation internally. Accordingly, it can also be stated that with regard to the abstraction levels the semantic integration remains on the logical level or even physical level as the mappings are performed between the XML level and the internal application specific representation.

On the other hand, compared to the main strategies for achieving semantic interoperability discussed in Section 4.4.1, the Chambers approach follows rather the standardization strategy than the mediation strategy between loosely coupled coexisting domain information models. Within the chambers organization only one shared XIHK standard for message exchange exists. The feasibility and effectiveness of this approach can be linked to the existence of the central umbrella association and the data conference working group of its eGovernment project office. These organizational conditions could enable the required degree of consensus among the independent stakeholders and thus establish the XIHK standard within the chambers.

Nevertheless, the mediation strategy between loosely coupled coexisting domain information models is well reflected when it comes to semantic integration with external partners. The chambers XIHK standard has been developed and defined completely independent of other information model standards in the eGovernment domain. In Germany, the XÖV [266] standards define XML Schemas for electronic message exchange in the public sector as already presented in the eGovernment scenario for the evaluation of the mediated service composition prototype (cf. Section 6.4.3). As mentioned there, XÖV consist of several standards for particular domains in eGovernment such as XGewerbe for business registration, XMeld for resident registration or XPersonenstand for vital records, e.g. birth certificates, death records, marriage licenses, etc. Furthermore, in order to achieve semantic interoperability in cross-domain processes, i.e. between the different XÖV standards, the XÖV AG Datenkonferenz [267] has suggested the so called XÖV core components [268]. The XÖV core components specifiy information entities such as *Name* and *Address*, which should be consistently reused in the various XÖV standards. However, it turned out that due to the complexity in the consensus finding process and due to heterogeneous requirements in the different domains, the feasibility for reuse of core components is limited (cf. Section 4.4). For instance the core component *Name* cannot be consistently defined and reused for XMeld and XPersonenstand because of contradicting legislative regulations [269]. In the legislation for resident registration, the surname has to be split obligatory into his subparts for electronic transmission. However, in the respective legislation for vital records, the whole surname has to be represented in a single string and cannot be represented in its subparts because for that purpose the existing IT applications of this domain would require costly reengineering [270].

Based on these experiences, the chambers data conference has decided to stay independent from any XÖV standards and not derive the Chambers XIHK standard from e.g. XGewerbe or the XÖV core components. Furthermore, this strategy has allowed the chambers to establish an efficient standardization process (cf. Section 7.2.2) without interfering into the complex organizational structures of the XÖV committees. Consequently, the data conference has anticipated the limited feasibility of cross-domain standardization and has adopted the mediation strategy based on loosely coupled domain information models, whereas XIHK represents such a domain information model.

According to the develop concept of semantic mediation, central mediation services have been planned to provide semantic integration with external business process partners in the larger eGovernment context. This mediation service should extend the Chambers service hub as the single point of contact for external electronic communication (cf. Section 7.2.1). It should provide a message translation from the XIHK standard to the relevant XÖV standards such as XGewerbe for business registration in which process the Chambers organization plays an active role. The following Figure 7-5 illustrates the idea of the mediation services:

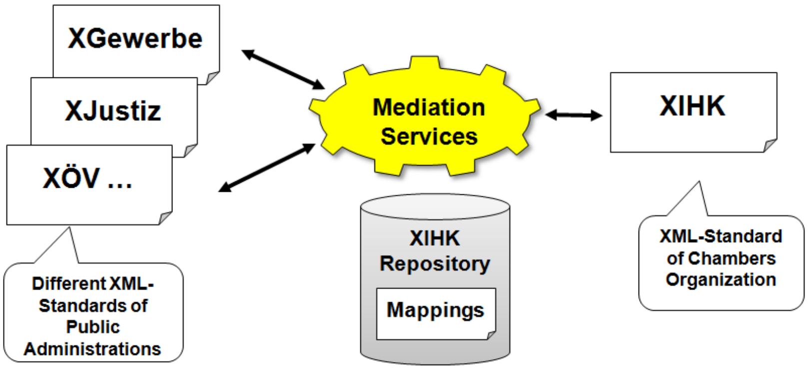


Figure 7-5 Planned Mediation Services of Chambers Service Hub

The mediation services are based on mappings between the different XML standards, e.g. realized by means of XSL transformations (cf. Section 3.2.3). They are stored centrally in the XIHK repository together with the XIHK information models.

However, the mediation services are not yet realized due to high costs for their realization and investment protection of existing adapters within decentralized IT applications. Thus, it can be stated that the semantic integration with external partners is successfully performed according to the approach of loose coupling between information models but not on domain level. The mapping between e.g. XGewerbe and XIHK is not targeted once by means of organization-wide mediation services but redundantly within existing decentralized IT applications. I.e. receiving XGewerbe messages are forwarded through the Chambers service hub and service bus to the particular Chamber and mapped to the internal information model and then are translated to XIHK messages when further communication with other Chambers is required. This can be understood for historical reasons as existing application-internal adapters have been in place and its consolidation requires a migration roadmap, which is a second step after the general SOA introduction.

Having discussed above the achievements of the Chambers but also their ongoing challenges with regard to effective and efficient semantic interoperability, the following section points out how a more complete adoption of the semantic mediation approach developed in this work could contribute to overcome the identified shortcomings.

7.2.4 Potential of the Semantic Mediation Approach

The potential of the semantic mediation approach is exemplary demonstrated by traversing through the semantic mediation methodology steps (cf. Chapter 5) and the respective semantic mediation toolkit (cf. Chapter 6) to discuss their impact and potential advantages for the Chambers organization and the semantic integration with its external partners.

The first step of the semantic mediation methodology is the step of domain ontology development (cf. Section 5.4). The Chambers organization has already developed their own chambers-wide information model XIHK, which is distinct from information models of other eGovernment domains (XÖV). However, as argued in the previous Section 7.2.3, the XML-based XIHK information model remains on the logical abstraction level instead of the conceptual level as targeted by domain ontologies. The Chambers data conference effort to establish in parallel a UML representation and textual information entity lists to be used on the conceptual level could not be achieved completely and consistently (cf. Section 7.2.3). Taken this into account, the development of domain ontologies for the Chambers domain and the discussed eGovernment domains provides a consistent conceptual level of the domain information models. The explicit conceptualization improves the consensus finding and standardization process within the domains and eases the expression of semantic bridges between the different domains (cf. Section 4.8.5). Moreover, due to the explicit and formal nature of description logic based ontologies – such as OWL ontologies as utilized in the semantic mediation toolkit – the domain ontologies can be consistently linked to underlying XML-based information representations such as the existing XIHK and XÖV standards. All in all, the development of domain ontologies establishes the additional semantic layer that provides the foundation for the further steps of the semantic mediation methodology.

The second step of the methodology is concerned with mediated business process modeling (cf. Section 5.5). The data conference of the Chambers organization already adopted a process-oriented approach to requirements engineering and design of new IT applications and corresponding information models (cf. Section 7.2.2). However, no consistent incorporation of existing conceptual information models during process modeling could be ensured. Therefore, applying the mediated business process modeling approach provides advantages regarding consistent reuse and evolution of existing domain information models. The chambers domain ontology could be directly used by non-technical business experts during process modeling for the design of information flows and thus eases the alignment between business and IT perspectives. Furthermore, the ontology-based approach allows to address semantic integration with external partners already on the conceptual level. By means of semantic bridges heterogeneous information representations in e.g. XGewerbe and XIHK could be kept transparent, which reduces complexity and efforts for process modeling in cross-organizational scenarios.

The methodology steps of semantic bridge definition (cf. Section 5.6) and testing (cf. Section 5.7) provide the means to realize the planned mediation services of the Chambers service hub. The declarative rule-based approach to express the mappings could reduce the development efforts in contrast to the planned but however not adopted technical complex XSL transformations. Furthermore, maintenance efforts which occur with changing versions of XIHK and XÖV could be reduced. Current redundantly maintained mappings in the decentralized IT applications could be replaced by semantic bridges between corresponding domain ontologies, which bears the already discussed advantage of addressing semantic integration declaratively on the domain level rather than redundantly and technically more complex on the application level.

In order to reap the benefits of the additional semantic layer and actually apply the semantic mediation mechanism, the so called grounding to the existing underlying XML-based Web services need to be ensured (cf. Section 5.8). This step of the semantic mediation approach does not directly provide any advantages for the Chambers case. In contrast it requires further efforts to define the liftings and lowerings between the ontology concepts and the corresponding XIHK elements for each service parameters. However, the step is required in order to enable the overall semantic mediation approach.

With regard to the mediated service composition step (cf. Section 5.9), the good starting point of the Chambers SOA initiative towards process-orientation for improved business-IT alignment could be further fostered. The during business process modeling utilized and further derived XIHK ontology concepts could be directly reused for service composition, in order to design the BPEL processes which instantiate the originally modeled business processes. Here the full benefits of the semantic mediation approach become an important factor. Firstly, the mediated service composition tool provides semantics-based matching functionality between corresponding service parameters and thus eases the design process in general. And secondly, it transparently integrates the semantic mediation mechanism based on the Chamber-wide deployed semantic bridges. Consequently, the whole service composition can be performed on the conceptual level, whereas heterogeneities between XIHK and XÖV information models are kept transparent to the service composer. Thus, no more technical transformation code has to be implemented within cross-organizational BPEL processes and high level Chamber business processes can be mapped to concrete IT infrastructures with less manual efforts.

Finally, the designed service compositions need to be interpreted during runtime in order to execute the actual business processes. Thereby, the additional semantic layer can be processed by the mediated business process execution engine (cf. Section 6.5) that enables the information processing on the conceptual level along with the semantic mediation mechanism during runtime.

The analysis of the Chambers case study has shown that the developed semantic mediation approach provides the potential for improving semantic integration within the distributed Chambers organization and particularly with external business partners. As the highest potential could be identified in cross-domain integration between XIHK and the various XÖV domain information standards, the next section provides an additional perspective with a larger scope.

7.2.5 Network Effect

As the overall goal of this work is to provide an effective and efficient approach to reduce complexity in semantic integration, it has to be analyzed if the subsumption of the proposed methodology steps, which are complex themselves, altogether indeed conduce the overall goal. Considering the various proposed activities, it has to be stated that firstly they produce additional efforts. These include the shifting of existing XML Schema-based domain standards to the ontology level as well as the additional description of WSDL-based Web services by wrapping Semantic Web service descriptions. Furthermore, semantic bridges have to be defined between all domains that are involved in cross-domain processes. And finally, the appropriate tools for utilizing all these additional artifacts have to be further developed to a mature product state, in order to exploit them for semantic integration support.

Considering again the Chambers case and the particular integration scenario with XGewerbe, a conventional approach with XML-based Web service composition would maybe provide a solution inducing less effort. By manually integrating the involved Web services, the development of e.g. XPath expressions for the data flow definition and e.g. XSL transformations for alignment of different representation formats would be probably less expensive than to develop the various semantic artifacts, including domain ontologies, semantic bridges and Semantic Web service descriptions.

However, the whole potential of the presented approach becomes only visible on a large scale. Once the additional semantic artifacts are available, a network effect exceeds the initial costs. The Chambers and XÖV-based services of public administrations are not only involved within one particular scenario but are integrated in various eGovernment processes. Once the Semantic Web service descriptions have been established, they are reused and thus the average costs per each particular scenario is marginal. Similar, this holds for the domain standards, where additionally domain ontologies have to be established as well as the semantic bridges that enable the required mediation between them. These artifacts are reused by all services from one domain and thus their creation efforts become reasonable. The core benefit of this approach lies in the fact that semantic interoperability is addressed on the level of domain standards instead of addressing it on application level. One-time ontology mediation substitutes n-time adaptation of parameter formats during process integration. Thereby, the high abstraction level of the information model provided by ontologies eases the maintainability of the semantic bridges.

Consequently, the full potential of the developed approach evolves when applying it to a larger scope, for instance not only to the distributed Chambers organization but to the overall XÖV landscape. Accordingly, domain ontologies would be required for each XÖV standard, including XGewerbe, XMeld, etc. (cf. Figure 7-5). However, the additional development of domain ontologies replacing the whole set of XÖV standards seems unrealistic especially with regard to technological path dependency. Nevertheless, when looking at the design process of XÖV standards as outlined in the XÖV-Framework [271], it includes a top-down approach similar to the Chambers data conference starting with UML models for conceptualizing the domain information models. Based on these UML models, which are restricted to well-defined UML design templates, the actual XÖV XML Schemas are generated. Therefore, one starting point for the adoption of the presented approach in this larger scope could be the exploitation of the UML models. Notably, they correspond to the conceptual abstraction level for information models and allow to generate as well domain ontologies besides the XML Schemas and to provide as well lifting and lowering transformation code in this process. Thus, the conceptual language features such as expressive relations between information entities, generalization and polymorphism could be preserved in the ontology representation. And consequently they could be exploited for semantic mediation on the conceptual level between the heterogeneous XÖV domains.

Based on the analysis above, it can be stated that the presented approach for semantic mediation has the potential to contribute substantially to the reduction of complexity in semantic integration. Furthermore, in order to finalize the evaluation the following section systematically refers back to the originally set individual goals for the developed approach and discusses the research hypothesis and the extent of its coverage.

7.3 Coverage of Goals and Confirmation of Research Hypothesis

In order to discuss whether and to which extends the conceptual goals could be attained, this second part of the evaluation systematically maps the outcomes of the analysis above to the individual goals set in Section 4.2 and then interprets the results with regard to the formulated research hypothesis and its confirmation.

7.3.1 Coverage of Conceptual Goals

The first conceptual goal has targeted the reduction of complexity in semantic integration by separation of technical issues from business issues. As outlined in the Chambers case and its larger scope in the XÖV landscape, the lifting of the semantic integration task to the conceptual level by means of semantic mediation between domain ontologies addresses this goal. Separation of concerns is improved as information flows can be designed on an abstraction level suitable for business experts as demonstrated with the mediated business process modeling prototype (cf. Section 6.2). Moreover, the transparent incorporation of declarative semantic bridges enables process experts to concentrate on the cross-organizational business logic without restrictions caused by heterogeneous information representations. As domain experts provide the semantic bridges between different domains from an overall domain perspective, the individual process expert is not required to have in parallel detailed insight into information models from other domains but can focus on his familiar domain perspective. Independent from business and conceptual tasks, the technical tasks such as the Semantic Web service enrichment in terms of liftings and lowerings can be performed by service developers, whereas the required transformations only concern the local information model. Consequently, the aim for separation of concerns could be achieved.

The second goal has been the reflection of limited feasibility of semantic standards in cross-organizational environments caused by differences in business requirements and organizational boundaries. The analyzed case study has demonstrated this requirement and it could be shown that the developed concept of loosely coupled information models (cf. Section 4.6.3) directly addresses this goal. In particular, autonomy and independent evolution of heterogeneous information models is anticipated and builds the foundation of the developed mediation mechanism.

A further goal has been to overcome the status quo of complicated and highly technical transformation coding for bridging heterogeneous information representations in semantic integration scenarios. As demonstrated with the semantic mediation toolkit, the semantic bridges are not only used during process modeling but as well during its instantiations in terms of Semantic Web service composition. Heterogeneous service interfaces are transparently mediated as service parameter descriptions are transformed to polymorph representations which correspond to the conceptualizations of different domains. Thus, the semantic integration is shifted on the domain level, instead of implementing it recurrently within each service composition realizing a concrete business process.

Another goal has targeted the expressiveness of the semantic mediation mechanism. As differences in information model representations can be complex, semantic bridges should be able to cover that complexity in terms of completeness but should as well remain easy to maintain by domain experts. With regard to this goal it can be stated that the implementation of the semantic bridges, which is based on description logic rules, provides a computational complete solution. Any semantic mappings including one-to-one, one-to-many or many-to-many between corresponding concepts can be covered as well as any transformation of underlying semantic sub-graphs. Furthermore, granularity differences (such as showcased in Section 4.8.4 with regard to e.g. *Street* and *StreetNumber* as separate fields or in contrast *StreetAddress* containing the two values in a combined field) can be mapped. This is enabled by integrating so called built-in functions which provide procedures such as string concatenation or unit transformation, etc. within the declarative rules. However, this completeness only covers mappings and transformations which are based on information contained explicitly or implicitly within the concepts to be mapped or static information such as unit transformations which can be embedded in built-ins. Other mappings which require external information such as the translation of zip codes to location names, which can also cause semantic integration problems, are not yet covered by the developed semantic mediation mechanism. However, this question is further discussed in the open issues part of the conclusion in Chapter 8, where starting points for possible extensions of the semantic mediation mechanism are discussed.

The final two goals have focused on industry suitability and technological path dependency. On the one hand, the developed conceptual approach should remain consistent to best practice SOA methodologies. As described in Section 5.3 the semantic mediation methodology has been strictly aligned to the SOA life cycle, starting with business analysis leading to process models, followed by service composition and process execution. In particular, no goal-based planning approaches have been integrated into the semantic mediation mechanism, as its suitability for achieving semantic interoperability in heterogeneous environments has been identified as less suitable (cf. Section 3.4.3).

On the other hand, a further goal has been the restriction to build upon existing concepts and standards of the World Wide Web and thus respect technological path dependency, especially with regard to Web service technologies which are the dominant instantiation of SOA. As described in Chapter 6, the semantic mediation toolkit provides its mediation functionality by means of an additional semantic layer (domain ontologies and semantic bridges) on top of existing XML-based Web service technology. Lifting and lowering mechanisms of Semantic Web service technology are used to connect the different abstraction layers between the semantics-based mediation layer and the existing traditional Web service technology. Consequently, it can be stated that technological path dependency has been respected.

According to the analysis above, it can be stated that the originally set conceptual goals could be almost completely covered. Minor aspects have been identified as open issues, which will be further discussed in the conclusion of the thesis, whereas starting points for further extensions and future work are pointed out. Finally, the confirmation of the research hypothesis is discussed in the next section.

7.3.2 Confirmation of Research Hypothesis

In order to confirm the hypothesis, this work has applied the methodology of design research in information systems as outlined in Section 1.3. Consequently, this section reviews the major artifacts produced in the methodology steps (awareness of the problem, suggestion, development and evaluation) to provide the argumentation why the hypothesis can be confirmed. To better structure this process the research hypothesis can be divided into three parts:

(1) The goal: … *to effectively and efficiently achieve semantic interoperability in large-scale cross-organizational service-oriented architectures…*

(2) The concept to achieve the goal: … *the principle of loose coupling can be applied to information models based on a flexible semantic mediation mechanism …*

(3) And the technology to instantiate the concept: ... *using Semantic Web technology for autonomous management and integration of domain-specific information models in terms of self-contained ontologies.*

In order to ensure that the goal (1) has been specified and scoped correctly, a systematic analysis of the problem area of semantic interoperability in SOA has been performed in Chapter 2 with special focus on large-scale cross-organizational environments. The main outcome has been a framework with its integral part describing the semantic interoperability gap between heterogeneous information models on different abstraction levels. This framework could be applied to analyze the state-of-the-art (cf. Chapter 3) and evaluate existing approaches with regard to their effectiveness and efficiency and to outline limitations and directions for the later presented concept.

The concept (2) of semantic mediation between loosely coupled information models (cf. Chapter 4) has been derived from the analysis of limited effectiveness with regard to feasibility and practicability of semantic standardization in large-scale service-oriented environments (cf. Section 4.4 and 4.5). This has led to the identification and transfer of key characteristics of loose coupling (cf. Section 4.6) to information models with its integral part of semantic mediation on the conceptual level by means of rule-based semantic bridges (cf. Section 4.8). To point out the advantages regarding effectiveness and efficiency, the developed concept has been compared to alternative approaches. With regard to an identified inherent trade-off between effectiveness and efficiency (cf. Section 4.7), it could be shown that the developed concept provides an optimized balance within this trade-off and thus meets the required adverbs (*effectively and efficiently*) in the goal part (1) of the research hypothesis.

Furthermore, it has been shown how Semantic Web technology and its specific features such as polymorphism, facet analysis classification and declarative rule-based entity manipulation (cf. Section 4.8) can be exploited to enable the semantic mediation mechanism. In order to instantiate the concept (3), the semantic mediation mechanism has been implemented based on independent OWL domain ontologies and SWRL rule-based semantic bridges. The mechanism has been integrated into several prototypes (cf. Chapter 6) covering the SOA lifecycle according to a developed semantic mediation methodology (cf. Chapter 5), ranging from business process modeling, over service composition to run-time process execution.

Combining the results from the case study-oriented evaluation (cf. Section 7.2) including the qualitative analysis regarding the coverage of the conceptual goals (cf. Section 7.3.1) and the three steps of the analytical review regarding the research hypothesis, it can be concluded that the research hypothesis can be confirmed. Identified remaining open issues and future work including possible further extensions are discussed in the conclusion of this work in Chapter 8.

7.4 Summary and Reflection

This chapter has presented the evaluation of the developed approach for semantic mediation between loosely coupled information models in SOA. As a quantitative evaluation would be out of scope for this thesis, the evaluation has followed a qualitative approach. The evaluation has been structured in two parts: Firstly, a case study about the semantic interoperability activities of the distributed organization of the German Chambers of Industry and Commerce has been carried out and compared in terms of a gap analysis with the developed approach for semantic mediation. Secondly, the outcome of this analysis has been mapped to the originally set conceptual goals from Section 4.2 and to the claims of the research hypothesis of this work in Section 1.3.1.

Firstly, the organizational background and IT landscape of the German Chambers of Industry and Commerce with its 80 decentralized independent entities served by four different IT service providers has been presented. In particular, the Chambers service bus as the major internal integration infrastructure has been discussed together with the Chambers service hub providing a single point of contact for electronic interaction with external partners.

The main focus of the case study then has been put on the Chambers data conference working group, which has developed a methodology for and an instantiation of a chamber-wide XML-based data exchange standard (XIHK). This corresponds to the in this work developed concept of loose coupling on the semantic level insofar, that the developed Chamber-wide message exchange standard XIHK remains independent from the internal information representations within the decentralized IT applications. Moreover, the Chambers XIHK information model has been designed independently from external information models in the eGovernment domain, namely the XÖV exchange standards. This has ensured an optimal reflection of Chamber requirements including its evolution without any organizational dependencies to external entities. Shortcomings could be identified regarding the consistent and complete representation of the XIHK information model on different abstraction levels, which has constrained the envisioned gains for alignment between business and IT perspectives. Furthermore, the semantic integration between the XIHK information model and the external XÖV information models has not been realized on the domain level as planned but still redundantly on the application level within decentralized IT applications.

In order to outline the potential of the developed semantic mediation methodology and the toolkit, it has been shown how they can be mapped to the Chambers context. It has been pointed out how the additional semantic layer on top of the existing XML-based Web service technology can reduce complexity in process integration and enable semantic mediation on domain level. Furthermore, a network effect has been identified which determines the full potential of the developed approach. Therefore, the scope has been enlarged beyond the applications within the Chambers organization to the overall XÖV landscape. In this larger scope, it could be shown how the ontology and semantic bridge-based approach could be integrated into the existing XÖV framework to improve semantic interoperability between the various XÖV standards and corresponding domains.

The second part of the evaluation has focused on the coverage of the originally set conceptual goals and the confirmation of the research hypothesis of this work. The conceptual goals have been recapitulated and mapped to the results of the case study and they have been discussed with regard to the developed concept for semantic mediation, the semantic mediation methodology and the implemented prototypical toolkit. It has been concluded that the conceptual goals could be accomplished successfully with minor open issues concerning e.g. the integration of external information within semantic bridges to support sophisticated semantic integration challenges.

Finally, the confirmation of the research hypothesis of this work has been discussed. Based on a three-step analysis, the hypothesis has been reviewed, whereas the previous evaluation results and the major artifacts produced in this work have been considered. Special emphasizes has been given to the aimed characteristics *effectiveness* and *efficiency* of the proposed approach, whereas the main argumentations of the previous chapters have been recalled. Structured according to the applied research methodology of design research the major artifacts include:

- the framework of semantic interoperability in SOA and a systematic state-of-the-art analysis as the analytical step of understanding the problem domain and providing awareness of the problem;
- the concept of semantic mediation between loosely coupled information models as the conceptual suggestion step;
- the instantiation of the concept by means of the semantic mediation methodology and the prototypical semantic mediation toolkit as the development step;
- and finally the evaluation in terms of the case study of the Chambers organization as an exemplarily distributed organization including the enlarged scope covering as well its external process partners in the eGovernment domain.

Consequently, an argumentation based on a qualitative analysis for the confirmation of the research hypothesis could be provided. Moreover, professional reviews of the published papers ([8], [185], [188], [245], [272], [273], [282]) have confirmed the conceptual and technical quality, originality and impact of the developed artifacts.

Chapter 8

Conclusion and Outlook

This chapter concludes the thesis. The aim of this work has been to develop an effective and efficient approach for semantic interoperability in large-scale service-oriented architectures based on semantic mediation between loosely coupled information models. The motivation has been to reflect that the dominant semantic integration approach of developing a single information model spanning multiple organizational domains has failed. The guiding idea has been to transfer the principle of loose coupling to the semantic level and consider semantic mediation between heterogeneous conceptualizations not as a necessary evil but as a silver bullet to tackle the challenge of semantic interoperability with a more flexible information architecture pattern. Furthermore, the goal has been to show how emerging semantic technologies can contribute to the instantiation of this concept based on their capabilities to explicitly express semantics. The following summarizes the central findings of this work and points out its scientific contributions. Finally, the evolution of the work is outlined, whereas open issues, potential advancements and future work and priorities in this area are discussed.

8.1 Summary and Main Contributions

In order to provide a concise overview of the thesis, the following recapitulates the line of argument and depicts the central aspects of the developed artifacts in a condensed manner.

The thesis has started with an examination of the general research context of semantic interoperability in the focused domain of service-oriented architectures. The outcome has been a framework (cf. Chapter 2) which has differentiated four major abstraction levels for the representation of information models: the initial conceptual idea in the mind, the conceptual model, the logical model and the underlying physical model in each IT system. Thereby, the so called semantic interoperability gap increases which each lower abstraction level.

The framework has been used as a common ground for comparison in a systematic state-of-the-art analysis (cf. Chapter 3) of existing approaches and technologies for achieving semantic interoperability in SOA. Firstly, traditional Web services along with the existing XML-based technology stack have been discussed followed by an evaluation describing their capabilities and limitations. After outlining the need for formally defined semantics of Web service descriptions, an intermediate step introducing the core concepts and technologies of the Semantic Web initiative has been provided. Then, it has been described how these concepts can be applied to Web services in terms of so called Semantic Web services. Furthermore, relevant related areas have been presented such as semantic information integration in distributed database systems and distributed object-oriented systems. Additionally, these traditional approaches have been related to a detailed analysis of ontology-based strategies for semantic integration including approaches where multiple ontologies and ontology mapping approaches are involved.

Based on the problem identification and the analysis of the state-of-the-art, Chapter 4 has presented the developed concept of semantic mediation between loosely coupled information models in SOA. Firstly, the general idea of the concept has been outlined in order to provide a condensed overview of the central aspects. These include mainly the shift from monolithic to loosely coupled information models combined with the approach to address semantic integration on a higher abstraction level for information representation. Thereby, the notion of a shift on a higher abstraction level has been elaborated along two dimensions. On the one hand, semantic integration is shifted from the schema or structure-based logical abstraction level to the conceptual abstraction level. And on the other hand, semantic mediation is addressed on domain level instead of recurrently on application or process level.

In the following the general idea has been refined: At first, the limitations of semantic standardization in large-scale service-oriented architectures with multiple organizational independent stakeholders have been pointed out. Then the underlying reason for context-dependency of information models has been deeper analyzed by referring to a model theoretic approach, namely the model of conception. Based on these findings, the transfer of the principle of loose coupling to the semantic level has been discussed and a specification of loosely coupled information models has been provided. According to the research hypothesis claiming for an effective and efficient approach for the semantic interoperability problem in SOA, the developed conceptual solution has been reflected including a proposed balance between these as competing identified sub-goals. Furthermore, the semantic mediation mechanism as the enabling part of the concept of loosely coupled information models has been specified based on Semantic Web concepts in terms of ontologies and description logic rules.

The main conceptual conclusions can be summarized as follows:

- Semantic interoperability can be addressed on different abstraction levels for information representation, whereas the semantic interoperability gap increases with each lower abstraction level.
- Traditional XML-based Web service approaches for semantic interoperability require high technical complexity, as they address the semantic interoperability gap on the lower logical or physical abstraction level and are applied recursively on the application or process level instead of on the domain level.
- Semantic Web service approaches provide the advantage to address the semantic interoperability gap on the higher conceptual level; however the dominant approaches are based on the idea of a common ontology that aims to cover exhaustively different organizational domains, which has limited feasibility in practice.
- Goal-based Semantic Web services approaches do not ease the semantic interoperability problem in heterogeneous environments, as different conceptualizations of service pre- and postconditions cause further semantic heterogeneity to overcome. Furthermore, the targeted automated planning of service compositions does not match to best practice SOA approaches, where control should remain with the human process expert.
- The underlying reason for the failure of common ontologies as an effective means for achieving semantic interoperability lies in the context dependency of information models. Different organizational domains have different requirements on information models to serve best for intra-domain information exchange, which results in limitations for semantic standardization across domains.
- To provide a flexible mediation between independent domain information models the principle of loose coupling can be transferred to the semantic level in terms of loosely

coupled information models expressed on the conceptual level. Thereby, a balance between effectiveness and efficiency can be ensured when semantic mediation is applied on domain level.

- Semantic Web concepts and technologies including features such as polymorphism, facet analysis classification and declarative entity manipulation can be exploited to enable the semantic mediation mechanism based on domain ontologies and description logic rules.

In order to instantiate the developed concept and provide practical evidence for the formulated research hypothesis, Chapter 5 has presented the semantic mediation methodology, which shows how the concept can be applied to the SOA life-cycle. Firstly, the basic steps of the SOA life-cycle have been recapitulated and an actors model including roles and responsibilities of stakeholders relevant for semantic mediation has been derived. The following actors could be identified: domain information model expert, business process expert, service developer, service composer and service or process consumer. Then the individual methodology steps have been discussed with regard to their goals, the tasks within each step and the required functionalities for performing them. The following seven methodology steps have been derived:

- Domain Ontology Development
- Mediated Business Process Modeling
- Semantic Bridge Definition
- Semantic Bridge Testing
- Semantic Service Enrichment
- Mediated Service Composition
- Mediated Business Process Execution

In order to prepare an experimental confirmation of key steps of the methodology in terms of a prototypical toolkit for semantic mediation in SOA, as well a high-level view on the functional architecture for each methodology step has been derived. Some steps of the methodology could be well covered with existing work and respective tools. Therefore, it has been abstained from implementing such functionalities redundantly. Rather, it has been focused on these methodology steps which cannot be performed adequately with existing tools. Accordingly, for the steps: domain ontology development, semantic bridge definition and semantic service enrichment existing work has been presented, whereas for the methodology steps: mediated business process modeling, semantic bridge testing, mediated service composition and mediated business process execution a dedicated functional architecture has been elaborated.

Based on the semantic mediation methodology and the functional architectures, Chapter 6 then has presented the developed toolkit, which demonstrates how the semantic mediation mechanism can be incorporated into key steps of the SOA life-cycle. The prototype for mediated business process modeling has demonstrated how business process experts can be enabled to design cross-organizational information flow explicitly on the conceptual level, whereas different domain conceptualizations are kept transparent and are mediated in the background. Additionally, it allows to derive and to identify further requirements for the used information models directly during process modeling. The prototype for semantic bridge testing has addressed the challenge that the underlying ontology mapping is a complex and error-prone tasks, so that users and developers need to be enabled to systematically test and determine the quality of semantic bridges. The purpose of the prototype for mediated service composition, which instantiates the previously modeled business processes, has been to demonstrate how the task of composing Web services described by heterogeneous ontologies can be facilitated in terms of semantic matchmaking and incorporation of rule-based semantic bridges. Finally, the prototype for mediated process execution has demonstrated how existing industry-proven process engines can be extended to process the semantic mediation mechanism during runtime. Thereby, the whole process execution remains on the ontology-based conceptual level and is just broken down at the latest possible point in terms of grounding Semantic Web service calls to underlying traditional XML-based Web services. The four prototypes have been described based on system requirements derived from the semantic mediation methodology followed by an outline of the design and realization in terms of system architectures. Finally, the developed prototypes have been validated and verified by applying them to integration scenarios including two cross-organizational eBusiness and eGovernment processes, which have been highlighted from different perspectives.

Finally, the evaluation of the developed approach for semantic mediation has been presented in Chapter 7. The evaluation has been structured in two parts: Firstly, a case study about an exemplarily distributed organization and its semantic interoperability activities has been carried out. This included namely an analysis of the service bus and the data conference of the German Chambers of Industry and Commerce, which then have been compared in terms of a gap analysis with the developed approach for semantic mediation. Secondly, the outcome of this analysis has been mapped to the originally set conceptual goals in Chapter 4 and to the claims of the research hypothesis.

The case study has outlined how several aspects of the approach of loosely coupled information models have been applied in practice and which benefits could be generated. Furthermore, shortcomings and missing conceptual and technological aspects have been identified and it has been shown how a complete application of the developed semantic mediation methodology and toolkit can provide further improvements. Moreover, it turned out that due to a network effect, the full potential of the approach becomes visible when mapping it to a larger scope. Consequently, the application of the approach to a larger context – exemplarily the German eGovernment landscape – has been discussed and the potential to ensure semantic interoperability across multiple organizational domains could be pointed out.

Finally, the research hypothesis has been reviewed based on a three step analysis, whereas the previous evaluation and the major artifacts produced in this work have been considered. Based on the evaluation and the conceptual argumentation in Chapter 4, the claimed research hypothesis could be confirmed.

Consequently, the main contributions of this work can be listed as follows:

- the framework of semantic interoperability in SOA mapped to an overview and evaluation of existing approaches for bridging the semantic interoperability gap;
- the concept of semantic mediation between loosely coupled information models in SOA;
- the semantic mediation mechanism on domain level based on ontologies and description logic rules;
- the semantic mediation methodology and the semantic mediation toolkit for the SOA life-cycle.

Finally, it should be noted that the central aspects of the contributions have been presented and published at international research conferences and workshops including ([8], [185], [188], [245], [272], [273], [282]).

8.2 Evolution and Outlook

Having presented a summary of the thesis and a conclusion of the main contributions, this section addresses the further evolution of it and provides an outlook on potential advancements and future work.

Starting from the presented approach, additional questions arise. For example, it has to be ensured that all stakeholders in cross-organizational business processes covering multiple domains have access to the required assets, including process models, information models and particularly the corresponding semantic bridges. The adoption of the concept of loose coupling on the semantic level prevents from the limitations of cross-domain commitment to an overall ontology and thus minimizes less practical central structures. However, it still requires a certain central organizational framework or kind of central clearinghouse to support the stakeholders in the process of providing and sharing the semantic assets. Such a clearinghouse should include a repository to publish the various business process models, domain information models and semantic bridges. It should provide means to categorize them with retrievable and thus expressive semantics and provide methods for versioning and quality assurance to ensure their sound evolution. For example in the eGovernment domain, the European Union has established the semantic interoperability center SEMIC.EU [274] including an assets repository. However, the focus is put on cross-border semantic interoperability within particular government branches and less on cross-domain challenges. Moreover, the explicit mission of SEMIC.EU is to promote reuse and harmonization of data formats and semantic assets [275]. This includes the aim to establish so called pivot assets for core information entities to be consistently reused in pan-European eGovernment processes, such as the recently published draft of a core person model [281]. Consequently, it still follows rather the monolithic information model approach and the concept of loose coupling on the semantic level and the provision and exchange of semantic bridges is not yet well reflected. Thus, the incorporation of the developed approach in this work into existing semantic clearinghouse platforms provides a potential field for future work.

Another organizational perspective includes the question regarding the scope of domains. The developed concept addresses semantic mediation on domain level but it leaves it open to the specific integration scenarios which coverage the domain ontologies should have. Generally, it has been argued that the scope of a domain ontology and the respective feasible semantic standardization is limited and depends on certain organizational structures such as the existence of an umbrella association etc. However, to derive concrete lower and upper bounds for adequate domain ontology coverage and to identify the relevant factors for it requires dedicated empirical studies in different organizational scenarios. From a macro-perspective this includes the question about the adequate granularity of a network of loosely coupled domain ontologies in an IT ecosystem of multiple organizations. Future research should address these issues with the goal to develop a certain heuristic on how to map the structure of relations between organizations or organizational entities to an adequate set of domain ontologies.

Besides these organizational issues, further functional advancements should be addressed in future work. One point has already been discussed within the evaluation and concerns the integration of external information into the semantic mediation mechanism. Semantic heterogeneities may include different representations of information such as zip codes vs. location names or representing a person with more or less detailed information about e.g. name and place of birth etc. These differences cannot be semantically bridged without considering further external information. A starting point for this advancement is to enable the rule-based semantic bridges to incorporate function calls, which then provide access to the required external information or functionality. The built-in functions already applied to provide procedures within the declarative rules such as string concatenation to overcome certain granularity differences could be reused as an entry point for external function calls. The generic extension then would consider the incorporation of Semantic Web service calls that process information on the same abstraction level as used within the description logic based rules and which then provide access to any kind of external functionality or information.

Regarding the general range of functionality of the semantic mediation toolkit, it has to be stated that it just provides the basic functions to demonstrate how the approach can be incorporated into the SOA life-cycle as a proof-of-concept. Thus, in order to evolve towards mature industry products, the prototypes have to be extended substantially or integrated into existing products to provide the required comprehensive functionality. E.g. the mediated service composer has just provided certain control flow and information flow features allowing the design of relatively simple BPEL-based service orchestrations. For more complex business processes as well the full set of advanced BPEL constructs including scopes, compensation handling and events would become necessary.

When discussing the general readiness of the approach for wide industry adoption, it has to be distinguished between the concept of loosely coupled information models in SOA as a more effective and efficient information architecture pattern and the proposed instantiation based on evolving Semantic Web technologies. As shown in the case study of the German Chambers of Industry and Commerce, it is already possible to apply the concept of loosely coupled information models with state-of-the-art traditional XML-based technologies to a certain extent. However, the full potential is just achieved by mapping it to Semantic Web technologies as for instance complicated technical transformation coding is substituted by the declarative rule-based approach. As Semantic Web technologies are still an emerging technology, most provided frameworks and APIs are still academic-driven and at so called beta stage and thus cannot yet be considered as technically stable and mature. In fact, during development and testing of the semantic mediation toolkit several problems and bugs did occur. While most of the reported bugs could be solved by contacting the developers of the provided frameworks (e.g. the Protégé-API for ontology handling), as well workarounds had to be implemented for some problems. To minimize these drawbacks and due to the raised claim of being not only of theoretical but also of practical relevance, the presented approach relies strongly on existing Web standards. This includes both: State-of-the-art XML-based technologies such as BPEL, XSLT, XPath and XML Schema on the one hand and Semantic Web technologies such as OWL, OWL-S and SWRL rules on the other hand. The frameworks and APIs supporting these existing standards can be regarded as the most mature including available tool support and stability compared to others. Nevertheless, the evolution of these standards and tools is still in progress and future versions of the developed semantic mediation toolkit should consider this evolution.

In this process, the future will show to which extent the Semantic Web technologies exploited for the additional semantic layer on top of traditional Web service technology will emerge to mature industry standards. An alternative path could be as well a kind of convergence. In this perspective, the identified benefits of the underlying language concepts of Semantic Web technologies such as polymorphism or facet analysis classification will have an impact in terms of an infusion into the evolution of the XML-based standards itself. For instance in [276] it is discussed how XML languages can be extended to support a polymorphic type system such as subtyping, inference of types, etc. However, irrespective of which path will be adopted, the gap between the two technology layers will be at the focus of further evolution. For instance a new recommendation for Semantic Web service descriptions namely SAWSDL [277] has recently been published by the W3C, which directly integrates semantic annotations into WSDL-based

Web services descriptions. Even though solid tool support is still missing and thus the older and more mature W3C recommendation OWL-S has been favored for the prototypical toolkit, the lightweight approach of SAWSDL seems to be promising and suitable to foster industry adoption. Another candidate for observation is XSPARQL [278], which promises to improve the translation between the traditional XML and the semantics-oriented RDF world. This is particularly relevant for grounding of Semantic Web services to existing traditional Web services. Currently, the developed semantic mediation toolkit performs the required lifting and lowering translations in terms of XSL transformations, which process the ontology annotations on the level of their XML serialization. XSPARQL promises a more adequate way as it is designed to understand both meta-models and to provide powerful means for transformation between the two in any direction. Consequently, the evolution of the semantic mediation toolkit should consider these advancements to further ease the provision of the additional semantic layer and thus facilitate industry adoption.

In the long run these activities should contribute to the vision of performing cross-organizational IT system integration in a seamless manner with less complexity. In this process, established SOA concepts and standards combined and advanced with powerful emerging Semantic Web technologies pave the way for more effective and efficient semantic interoperability enabled by semantic mediation between loosely coupled information models.

Bibliography

[1] Bass and Lee 2003. Building a Business Case for Enterprise Integration. Accenture Research & Insights, www.accenture.com/xdoc/en/services/insights/building.pdf.

[2] Medjahed, B., et al. 2003. Business-to-business interactions: issues and enabling technologies. The VLDB Journal, vol. 12, no. 1, pp. 59-85.

[3] European Commission IDABC 2004. The European Interoperability Framework (EIF) Version 1.0.

[4] Erl, T. 2005. Book: Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall.

[5] Vetere, G. and Lenzerini, M. 2005. Models for Semantic Interoperability in Service-oriented Architectures. IBM Syst. J.

[6] Doan, A., Noy, N. F. and Halevy, A. Y. 2004. Introduction to the Special Issue on Semantic Integration. SIGMOD Rec. vol. 33, no. 4, pp. 11-13.

[7] Noy, N. F. 2004. Semantic integration: a survey of ontology-based approaches. SIGMOD Rec. vol. 33, no. 4, pp. 65-70.

[8] Barnickel, N., Fluegge, M. and Schmidt, K. 2006. Interoperability in eGovernment through Cross-Ontology Semantic Web Service Composition. Proceedings Workshop Semantic Web for eGovernment of 3rd European Semantic Web Conference.

[9] Simon, H.A. 1996. Book: The Sciences of the Artificial. 3rd edn. MIT Press.

[10] March, S. T. and Smith, G. F. 1995. Design and natural science research on information technology. Decis. Support Syst. vol.15, no. 4, pp. 251-266.

[11] Vaishnavi, V. and Kuechler, W. 2004. Design Research in Information Systems, http://desrist.org/design-research-in-information-systems/

[12] OASIS WSBPEL TC: Web Services Business Process Execution Language Version 2.0. Committee Specification, 31 January 2007, http://docs.oasis-open.org/wsbpel/2.0/CS01/wsbpel-v2.0-CS01.html

[13] Miller, P. 2000. Interoperability. What is it and Why should I want it? Ariadne Magazine, Issue 24, University of Bath UK

[14] European Commission IDABC. 2008. Draft Document as basis for European Interoperability Framework 2.0.

[15] Sheth, P. 1999. Book: Interoperating Geographic Information Systems, Chapter 2 Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics. Kluwer Academic Publishers Group

[16] Uschold, M. and Gruninger, M. 1996. Ontologies: principles, methods, and applications. Knowledge Engineering Review, vol. 11, no. 2, pp. 93–155.

[17] Morris, C. W. 1971. Writings on the General Theory of Sign, Approaches to Semiotics. Vol. 16.

[18] Pokraev, S., Reichert, M. et al. 2005. Semantic and Pragmatic Interoperability: A Model for Understanding. Proceedings of the Open Interop Workshop on Enterprise Modelling and Ontologies for Interoperability.

[19] World Wide Web Consortium 2002. Web Services Activity, www.w3.org/2002/ws/

[20] Veltman, K. 2001. Syntactic and Semantic Interoperability: New Approaches to Knowledge and the Semantic Web, The New Review of Information Networking, vol. 7.

[21] Eagleton, T. 1996. Book: Literary Theory: An Introduction. Chapter 3: Structuralism and Semiotics, University of Minnesota Press.

[22] Bell, D. 2004. UML basics: The class diagram. An introduction to structure diagrams in UML 2. IBM Technical Library, www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/

[23] Chen, P. 1976. The Entity-Relationship Model: Towards a Unified View of Data, ACM Transactions on Database Systems. ACM-Press, p. 9-36.

[24] Miller, J. and Mukerji, J. 2003. MDA Guide Version 1.0.1, OMG.

[25] Object Management Group (OMG), Model Driven Architecture, www.omg.org/mda

[26] Bieberstein et al. 2006. Book: Service-Oriented Architecture (SOA) Compass. Pearson plc, pp. 1-19.

[27] Object Management Group. 1993. The common object request broker: Architecture and specification. Revision 1.1/Omg Document No 91.12.1

[28] DeMichiel, L.G et al. 2002. Enterprise JavaBeans TM Specification, Sun Microsystems, Inc.

[29] Olsen, G. 2006. From COM to Common. Queue Journal, Vol. 4, No. 5, pp. 20-26.

[30] Feng, G., Wang, C, Li, H. 2005. Web Services Based Cross-Organizational Business Process Management. In Proc. of the seventh Asia-Pacific Web Conference - APWeb2005. Springer Berlin/Heidelberg, pp. 548-559.

[31] Gerneth, M., Hansen, H., Mahr, B. et al. 1995. A Framework for Telemedicine Services in Europe - An Essential Guide for Establishing Telemedicine Services, KIT-Report 123, Technical University Berlin.

[32] Arsanjani, A. 2004. Service-oriented modeling and architecture. IBM Technical Library, www.ibm.com/developerworks/library/ws-soa-design1/

[33] Gu, Q. and Lago, P. 2007. A stakeholder-driven service life cycle model for SOA. In Proc. of IW-SOSWE’07, pages 1–7.

[34] McBride, G. 2007. The Role of SOA Quality Management in SOA Service Lifecycle Management, IBM Technical Library, www.ibm.com/developerworks/rational/library/mar07/mcbride

[35] Barstow, A., Hendler, J. et al. 2004. OWL Web Ontology Language for Services (OWL-S), W3C Submissions.

[36] Wikipedia article: Function (mathematics), http://en.wikipedia.org/wiki/Function_(mathematics), accessed 30/05/2010

[37] Fluegge, M. 2008. PhD Thesis: Adaptive Service Composition in a semantics-based Interoperability Infrastructure. Technical University Berlin.

[38] World Wide Web Consortium 2002. Web Services Activity, www.w3.org/2002/ws/

[39] The Integrity Center, Inc. 2006. Web Services FAQs. www.integctr.com/WebServices/FAQ105W.asp

[40] Berners Lee, T. et al. 2005. Uniform Resource Identifier. IETF RFC 3986.

[41] World Wide Web Consortium 2004. Web Services Architecture Requirements. W3C Working Group Note, www.w3.org/TR/wsa-reqs/

[42] Bettag, U. 2005. Article: Web Services. Informatiklexikon. Gesellschaft für Informatik, www.gi-ev.de/no_cache/service/informatiklexikon/informatiklexikon-detailansicht/meldung/web-services-95/

[43] Cervantes, H. and Hall, R.S. 2004. Autonomous Adaptation to Dynamic Availability Through A Service-Oriented Component Model. Proceedings of the International Conference on Software Engineering.

[44] IBM 2004. Web Services Transactions specifications, www.ibm.com/developerworks/library/ws-coor

[45] OASIS 2006. WS-Security 1.1 OASIS standard. Web Services Security (WSS) TC, www.oasis-open.org/committees/wss/

[46] World Wide Web Consortium 2007. SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). W3C Recommendation.

[47] Wide Web Consortium 2007. Web Service Description Language (WSDL) Version 2.0 Part 0: Primer. W3C Recommendation.

[48] IBM 2001. Web Services Flow Language (WSFL) Version 1.0.

[49] Microsoft 2001. XLANG, http://msdn.microsoft.com/en-us/library/aa577463(BTS.70).aspx

[50] OASIS 2007.Web Service Business Process Execution Language Version 2.0, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[51] World Wide Web Consortium 1999. XML Path Language (XPath) Version 1.0. W3C Recommendation.

[52] World Wide Web Consortium 1999. XSL Transformations (XSLT) Version 1.0 W3C Recommendation.

[53] Paolucci, M., Sycara, K. et al. 2003. Toward a Semantic Web e-Commerce. Proc. of 6th Int. Conf. on Business Information Systems.

[54] Klein, M. and Bernstein, A. 2004. Towards High-Precision Service Retrieval. IEEE Internet Computing, vol. 8, p. 30-36.

[55] Srivastava, B. and Koehler, J. 2004. Web Service Composition - Current Solutions and Open Problems. Proceedings Workshop on Planning for Web Services. ICAPS.

[56] Cover, R. 1998. XML and semantic transparency. Technology reports. OASIS cover pages, http://xml.coverpages.org/xmlAndSemantics.html

[57] World Wide Web Consortium 1998. Document Object Model Level 1. W3C Recommendation, http://www.w3.org/DOM/Activity

[58] Hitzler P., Kroetzsch M., Rudolph S. and Sure Y. 2008. Book: Semantic Web. Grundlagen. Springer, 2008. p.10.

[59] Baader, F., Horrocks, I. and Sattler, U. 2001. Description Logics for the Semantic Web. IEEE Data Engineering Bulletin, vol. 16 no. 25, pp. 4-9.

[60] Berners-Lee, T., Hendler, J. and Lassila, O. 2001. Journal Article: The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American, www.scientificamerican.com/article.cfm?id=the-semantic-web

[61] World Wide Web Consortium 2001. Semantic Web Activity, www.w3.org/2001/sw/

[62] World Wide Web Consortium 2004. Resource Description Framework (RDF), Concepts and Abstract Syntax. W3C Recommendation, www.w3.org/RDF/

[63] Dumbill, E. 2001. Building the Semantic Web. Knowledge Technologies Conference.

[64] Gruber, T. R. 1993. A translation approach to portable ontology specifications. Knowl. Acquis. 5, 2 (Jun. 1993), 199-220.

[65] Davies, J., Fensel, D. and Van Harmelen, F. 2003. Book: Towards the Semantic Web: Ontology driven Knowledge Management. Sussex: John Wiley and Sons Ltd.

[66] Fensel, D. 2003. Book: Ontologies: A Silver Bullet for Knowledge Management and E-Commerce. Springer, Berlin, Heidelberg, 2 edition.

[67] Baader, F., Horrocks, I. and Sattler, U. 2002. Description Logics for the Semantic Web, KI Jorunal, vol. 16, no. 4, pp. 57-59.

[68] Gödel, K. 1931. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I, Monatshefte für Mathematik und Physik, vol. 38, pp. 173-98.

[69] Donini, F. M. et al. 1991. The complexity of concept languages. Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning.

[70] Hart, L. and Emery, P. 2004. A Description Logic for Use as the ODM Core. AT&T Government Solutions, Inc.

[71] Stollberg, M. 2008. PhD Thesis: Scalable Semantic Web Service Discovery for Goal-Driven Service-Oriented Architecture. University of Innsbruck.

[72] de Bruijn, J. 2006. Logics for the Semantic Web. In Cardoses, J., Semantic Web: Theory, Tools and Applications. Idea Publishing Group.

[73] Horrocks, I., Li, L., Turi, D., and Bechhofer, S. 2004. The In- stance Store: Description Logic Reasoning with Large Numbers of Individuals. In Proc. of International Workshop on Description Logics (DL 2004).

[74] Motik, B., Shearer, R., and Horrocks, I. 2007. Optimized Reasoning in Description Logics using Hypertableaux. In Proc. of the 21st Conference on Automated Deduction (CADE-21).

[75] Hepp, M., de Leenheer, P., de Moor, A., and Sure, Y. 2007. Book: Ontology Management: Semantic Web, Semantic Web Services, and Business Applications. Semantic Web and Beyond. Springer.

[76] Gomez-Perez, A., Corcho, O., and Fernandez-Lopez, M. 2003. Ontological Engineering: With Examples from the Areas of Knowledge Management, E-Commerce and Semantic Web. Series of Advanced Information and Knowledge Processing. Springer.

[77] Harth, A. and Decker, S. 2005. Optimized Index Structures for Querying RDF from the Web. In Proc. of 3rd Latin American Web Congress.

[78] de Leenheer, P. and Mens, T. 2006. Chapter 5 Ontology Evolution: State of the Art and Future Directions. In Hepp, de Leenheer, de Moor, and Sure, editors, Ontology Management. Semantic Web, Semantic Web Services, and Business Applications. Springer.

[79] Noy, N. 2004. Semantic Integration: A Survey of Ontology-based Approaches. ACM SIGMOD Record, vol. 33, no. 4, pp. 65-70.

[80] Scharffe, F. and de Bruijn, J. 2005. A language to specify mappings between ontologies. In Proc. of the Internet Based Systems IEEE Conference (SITIS05).

[81] Davis, J., Studer, R., and Warren, P. 2006. Book: Semantic Web Technology. Trends and Research in Ontology-based System. Wiley & Sons

[82] Borkin, S. A. 1980. The Semantic Relation Data Model: Foundation for a User Interface. ICOD, pp. 47-64.

[83] World Wide Web Consortium 2004. RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation.

[84] Defense Advanced Research Projects Agency (DARPA). 2000. DARPA Agent Markup Language, www.daml.org

[85] Fensel, D. et al. 2001. OIL: an ontology infrastructure for the Semantic Web. IEEE Intelligent Systems, vol. 16, no. 2.

[86] Horrocks, I. et al. 2004. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission.

[87] World Wide Web Consortium. 2005. Rule Interchange Format (RIF) Working Group, www.w3.org/2005/rules/

[88] Bontas, E. and Mei, J. 2005. Reasoning Paradigms for SWRL-enabled Ontologies. Technical Report TR-B-05-01. Freie Universitaet Berlin.

[89] Horrocks, I. et al. 2005. Semantic Web Architecture: Stack or Two Towers?. Principles and Practice of Semantic Web Reasoning, pp. 37-41.

[90] Hobbs, J. R. and Pan, F. 2004. An Ontology of Time for the Semantic Web. ACM Transactions on Asian Language Information Processing, vol. 3, pp. 66-85.

[91] Chen, H., Fellah, S. and Bishr, Y. 2005. Rules for geospatial semantic web applications. W3C Workshop on Rule Languages for Interoperability.

[92] Sanchez, E. et al. 2006. Book: Fuzzy Logic and the Semantic Web, Elsevier Science.

[93] Stoilos, G. et al. 2006. Deliverable Analysis of Requirements for Further Language Extensions, EU-IST Network of Excellence Knowledge Web.

[94] World Wide Web Consortium 2009. OWL 2 Web Ontology Language Profiles. W3C Recommendation.

[95] Ruttenberg, A. et al. 2007. Advancing translational research with the Semantic Web. BMC Bioinformatics.

[96] Hyvönen, E. et al. 2007. Elements of a National Semantic Web Infrastructure. Proceedings of the First International Semantic Computing Conference (IEEE ICSC 2007).

[97] World Wide Web Consortium 2009. Semantic Web Service Interest Group, www.w3.org/2002/ws/swsig/

[98] McIlraith, S., Son, T. and Zeng, H. 2001. Semantic Web Services. IEEE Intelligent Systems. Special Issue on the Semantic Web, vol. 16, no. 2, pp.46-53.

[99] Gugliotta, A. 2006. Knowledge Modelling for Service-oriented Applications in the e-Government Domain. AAAI Spring Symposium Series Stanford University.

[100] Trapero, R. 2008. Deliverable Service Deployment and Management System, IST-FP6-Project Opuce.

[101] World Wide Web Consortium 2007. Semantic Annotations for WSDL and XML Schema, Terminology. W3C Recommendation.

[102] Martin, D. 2004. OWL-S: Semantic Markup for Web Services. W3C Member Submission.

[103] Lausen, H., Polleres, A., and Roman (eds.), D. 2005. Web Service Modeling Ontology (WSMO). W3C Member Submission.

[104] Battle, S., et al. 2005. Semantic Web Services Framework (SWSF). W3C Member Submission.

[105] Akkiraju, R. et al. 2005. Web Service Semantics - WSDL-S. W3C Member Submission.

[106] Farrell, J. and Lausen, H. 2007. Semantic Annotations for WSDL and XML Schema. W3C Recommendation.

[107] World Wide Web Consortium 2004. OWL Web Ontology Language for Services (OWLS). W3C Recommendation.

[108] Sirin, E. 2005. Tools for Semantic Web Services. Developers Day at the 2005 World Wide Web Conference, Japan.

[109] Horrocks, I. et al. 2004. SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission.

[110] de Bruijn, J et al. 2005. The Web Service Modeling Language WSML. Deliverable D16.1. WSML Working Group, www.wsmo.org/wsml

[111] Haller, A. et al. 2005. WSMX - a semantic service-oriented architecture, in Web services. In Proceedings of the IEEE international Conference on Web Services (ICWS 2005).

[112] Stollberg, M. and Norton, B. 2007. A Refined Goal Model for Semantic Web Services. Proceedings of the Second International Conference on Internet and Web Applications and Services (ICIW 2007).

[113] Johnson, T. and Eigenmann, R. 2006. Context-Sensitive Domain-Independent Algorithm Composition and Selection. ACM SIGPLAN Notices, vol. 41.

[114] Mocan, A. et al. 2005. WSMO Mediators. D29v0.1. WSMO Final Draft.

[115] Ankolekar, A. et al. 2004. OWL Web Ontology Language for Services (OWLS). OWL-S' Relationship to Selected Other Technologies, W3C Member Submission.

[116] Gruninger, M. and Menzel, C. 2003. The Process Specification Language (PSL) Theory and Applications. AI Magazine, vol. 24, no. 3, pp. 63-74.

[117] Akkiraju, R. et al. 2005. Web service Semantics – WSDL-S, W3C Member Submission.

[118] Kopecky, J. 2007. Semantic Annotations for WSDL and XML Schema. Presentation at W3C track of the WWW 2007 Conference.

[119] Polleres, A. 2004. AI Planning for Semantic Web Service Composition?, Survey of current approaches, challenges and open questions, Presentation, DERI, Innsbruck.

[120] Srivastava, B. and Koehler, J. 2003. Web Service Composition - Current Solutions and Open Problems. ICAPS Workshop on Planning for Web Services.

[121] Stollberg, M. 2008. PhD Thesis: Scalable Semantic Web Service Discovery for Goal-driven Service-Oriented Architectures. University of Innsbruck.

[122] Hendler, J. 2004. Composition as planning, W3C Mailing List Semantic Web Service Interest Group, http://lists.w3.org/Archives/Public/public-sws-ig/2004Feb/0001.html

[123] The Semantic Dialogues 2009. Progress Software Corporation, http://web.progress.com/de-de/semanticdialogues/

[124] Noy, N. F. 2004. Semantic integration: a survey of ontology-based approaches. SIGMOD Rec. vol. 33, no. 4, pp. 65-70.

[125] ITU-T, ISO/IEC 1995. Recommendation X.902. International Standard 10746-2. ODP Reference Model: Descriptive Model.

[126] Kimball, R. and Caserta, J. 2004. Book: The Data Warehouse ETL Toolkit. Chapter Introduction pp. xxxi, Wiley.

[127] Rahm, E. and Berstein, P. 2001. A survey of approaches to automatic schema matching. The VLDB Journal, vol. 10, pp.334-350.

[128] Kemper, A. and Eickler, A. 2004. Datenbanksysteme, Oldenburg, pp. 129-133.

[129] Cal, A. et al. 2002. On the Expressive Power of Data Integration Systems. Proceedings of the 21st International Conference on Conceptual Modeling, pp. 338-350.

[130] Naumann, F. 2006. Lecture Script: Informationsintegration. Humboldt Universität Berlin, www.ki.informatik.hu-berlin.de/wbi/ii/folien/InfoInt_02_Szenarien.ppt

[131] Vallecillo, A. 2001. RM-ODP: The ISO Reference Model for Open Distributed Processing, DINTEL Edition on Software Engineering, no. 3, pp. 69-99.

[132] Meyer, B. 1998. Book: Object-oriented software construction, Prentice Hall.

[133] Fluegge, M, Jaeger, M. C. 2006. Adding Value to E-Services: A Business-Oriented Model, in (Jie Lu et. al. Eds.): E-Service Intelligence - Methodologies, Applications, Technologies in Springer's book series on Computational Intelligence.

[134] Vallecillo, A. 2001. RM-ODP: The ISO Reference Model for Open Distributed Processing, DINTEL Edition on Software Engineering no. 3, pp. 69-99.

[135] Kutvonen, L. 1998. PhD thesis: Trading services in open distributed environments., Department of Computer Science. University of Helsinki.

[136] Müller, S. et. al. 1996. Global Trader Cooperation in Open Service Markets. Trends in Distributed Systems, pp.214-227.

[137] Uschold, M. and Gruninger, M. 1996. Ontologies: Principles, methods and applications. Knowledge Engineering Review, vol. 11, no. 2, pp. 93-155.

[138] van Harmelen, F. 2006. Semantic Web Research anno 2006: main streams, popular fallacies, current status and future challenges. Proceedings CIA-2006. Number 4149 in Lecture Notes in Artificial Intelligence.

[139] Wache, H. et al. 2001. Ontology-Based Integration of Information - A Survey of Existing Approaches. In Stuckenschmidt, H., ed., IJCAI-01 Workshop: Ontologies and Information Sharing, pp. 108-117.

[140] Ehrig, M.; Staab, S.: 2004. QOM - quick ontology mapping. In Proceedings of the 3rd International Semantic Web Conference, pp. 683-697.

[141] Euzenat, J. and Shavaiko, P. 2007. Chapter 5: Matching strategies, pp. 133, in Book: Ontology Matching, Springer.

[142] van Harmelen, F. 2006. Semantic Web Research anno 2006: main streams, popular fallacies, current status and future challenges. Proceedings CIA-2006. Number 4149 in Lecture Notes in Artificial Intelligence.

[143] Scharffe, F. 2006. Deliverable WP2: Ontology Management D2.6 Mapping Tool Background. European Commission Integrated Project: Data, Information, and Process Integration with Semantic Web Services (DIP).

[144] Euzenat, J. and Shavaiko, P. 2007. Chapter 11 Conclusion in Book: Ontology Matching. Springer, pp. 274.

[145] Calvanese, D., De Giacomo, G. and Lenzerini, M. 2001. A framework for ontology integration. In Proc. of the First Semantic Web Working Symposium, pp. 303-316.

[146] Maedche, A. et. al. 2002. Mafra - A mapping framework for distributed ontologies. Proceedings of the EKAW, vol. 2473 of Lecture Notes in Computer Science, Springer.

[147] Dou, D., McDermott, D. and Qi, P. 2004. Ontology Translation on the Semantic Web. LNCS Journal of Data Semantics, vol. II, LNCS 3360, pp.35-57.

[148] Zhao, G., Zheng J. and Meersman, R. 2003. An Architecture Framework of Ontology Development and Deployment. Proceedings of the E-Society IADIS International Conference.

[149] Kalfoglou, Y. and Robertson, D. 2000. Applying experienceware to support ontology deployment. In Proceedings of the International Conference on Software Engineering and Knowledge Engineering.

[150] Li, Y. et al. 2003. Beyond Ontology Construction; Ontology Services as Online Knowledge Sharing Communities. Lecture Notes in Computer Science, vol. 2870/2003, pp.469-483.

[151] Cali, A. et al. 2008. Tightly Integrated Probalistic Description Logic Programs for Representing Ontology Mappings. Proceedings of the 5th International Symposium on Foundations of Information and Knowledge Systems.

[152] Zapletal, M. et al. 2008. UN/CEFACT's Modeling Methodology. Vdm Verlag Dr. Müller.

[153] Damodaran, S. 2004. B2B Integration over the Internet with XML-RosettaNet - Successes and Challenges. In Proceedings 13th Int'l Conf. World Wide Web, pp. 188-195, ACM Press.

[154] Bosak, J. et al. 2006. Universal Business Language 2.0, OASIS.

[155] United Nations Centre for Trade Facilitation and Electronic Business. 2003. Core Component Technical Specification – Part 8 of the ebXML Framework.

[156] Jakobs, K. 2000. Book: Information technology standards and standardization: a global perspective, preface p.i, Idea Group Inc (IGI).

[157] McDermott, D. 1993. Book Review of Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project (D.B. Lenat and R.V. Guha). Artificial Intelligence 61, pp.53-63.

[158] Rada, R. 2000. Chapter 2 Consensus Versus Speed. In Book: Information technology standards and standardization. Idea Group Inc (IGI).

[159] Cox, K. J. 2007. Gloabal B2B Forum, B2B Standards Primer.

[160] Gibb, B. K. and Damodaran, S. 2002. Ebxml: Concepts and Application. John Wiley & Sons, Inc.

[161] Air Transport Association. 2000. e-Business Specifications. Spec 2000, http://www.ataebiz.org/specifications/

[162] Open Application Group, Inc. (OAGi). 2009. Chemistry Industry Data eXchange (CIDX), www.cidx.org

[163] Commerce eXtensible Markup Language (cXML) 1..2. 2008., www.cxml.org

[164] Commerce One. 2000. XML Common Business Library (xCBL), www.xcbl.org

[165] Weisgerber, S. 2008. Interoperabilität und Normung aus Sicht des Deutschen Instituts für Normung. Workshop: E-Government-Standards für Wirtschaft und Verwaltung. D21-Kongress.

[166] Blanchard, H.E. 1996. Standards: A Look Ahead and an Overview, SIGCHI Bulletin, vol.28 no.4.

[167] Translated from: Weisgerber, S. 2008. Interoperabilität und Normung aus Sicht des Deutschen Instituts für Normung. Workshop: E-Government-Standards für Wirtschaft und Verwaltung. D21-Kongress

[168] Turowski, K. 2000. Establishing Standards for Business Components. In Jakobs, K. (Hrsg.): Information Technology Standards and Standardisation: A Global Perspective. Idea Group Publishing, Hershey, pp.131-151

[169] Mahr, B. 2006. Ein Modell der Auffassung, KIT-Report 151, Technical University Berlin.

[170] Mahr, B. 2009. Information Science and the Logic of Models. Software and Systems Modeling. Springer Berlin, vol. 8, no. 3, pp. 365-383.

[171] Mahr, B. 1997. Gegenstand und Kontext: Eine Theorie der Auffassung. In Prinzipien der Kontextualisierung. Technical University Berlin.

[172] Orton, J. D., Weick, K. E. 1990. Loosely Coupled Systems: A Reconceptualization, Academy of Management Review, vol. 15, no. 2, pp. 203-223.

[173] Wikipedia article: Loose Coupling, http://en.wikipedia.org/wiki/Loose_coupling, accessed 30/05/2010

[174] Hess A, Humm B, Voß M. 2006. Regeln für serviceorientierte Architekturen hoher Qualität. Informatik Spektrum, vol. 29, no. 6, pp. 395–411.

[175] Humm, B. 2008. Lecture Script: Service-Oriented Architecture. Darmstadt University of Applied Sciences, www.fbi.h-da.de/fileadmin/personal/b.humm/SOA_SS08/Lecture/07_Loose_Coupling.pdf

[176] Translated and adapted from: Hess, A., Humm, B. and Voß, M. 2006. Figure 7. Regeln für serviceorientierte Architekturen hoher Qualität. Informatik Spektrum, vol. 29, no. 6, pp. 395–411.

[177] Stevens, W., Myers, G. and Constantine, L. 1997. Structured Design. IBM Systems Journal, vol. 13, no. 2, pp. 115-139.

[178] Schorlemmer, M. and Kalfoglou, Y. 2008. Institutionalising Ontology-Based Semantic Integration. Journal of Applied Ontology, vol. 3, no. 3.

[179] Bicer, V. et. al. 2005. Providing Semantic Interoperability in the Healthcare Domain through Ontology Mapping. Sigmod Record, vol. 34, no. 3.

[180] Ehrig, M. and Staab, S. 2004. QOM - Quick Ontology Mapping. In F. van Harmelen, S. McIlraith, and D. Plexousakis, editors, Proceedings of the Third International Semantic Web Conference.

[181] Klein M. et. al. 2000. The Relation between Ontologies and Schema-languages: Translating OIL-specifications in XML-Schema. In Proceedings of the 14th European Conference on Artificial Intelligence (ECAI'00).

[182] Taylor, A. G. 1992. Book: Introduction to Cataloging and Classification. 8th ed. Englewood, Colorado. Libraries Unlimited.

[183] Ranganathan, S.R. 1957. Book: Prolegomena to Library Classification. The Garden City Press Ltd.

[184] World Wide Web Consortium. 2004. OWL Web Ontology Language Reference. 3.1.2 Property restrictions. W3C Recommendation.

[185] Barnickel, N., Weinand R. and Fluegge M. 2008. Semantic System Integration – Incorporating Rule-based Semantic Bridges into BPEL processes. In Proceedings of EON-SWSC 2008 Evaluation of Ontology-based Tools and the Semantic Web Service Challenge (ESWC2008).

[186] Bosch, J. and Kruege, C. 2004. Proceedings Software reuse: methods, techniques, and tools. 8th International conference (ICSR 2004), pp.28.

[187] Fensel, D. 2003. Book: Ontologies: A Silver Bullet for Knowledge Management and E-Commerce. Springer, Berlin, Heidelberg, 2 Edition.

[188] Barnickel, N. 2008. Organisationsübergreifende Entwicklung von Prozessorientierten Datenaustauschstandards. Fachtagung Verwaltungsinformatik der Gesellschaft für Informatik.

[189] Noy, N. F. and McGuinness, D. L. 2001. Ontology development 101: A guide to creating your first ontology, Stanford Knowledge Systems Laboratory Technical Report KSL-01-05

[190] Sure, Y. et al. 2002. Collaborative Ontology Development for the Semantic Web. Lecture notes in computer science. Springer.

[191] Noy, N. F., Crubezy, M. and Fergerson, H. 2003. Protégé Ontology Editor. Stanford University's National Center for Biomedical Ontology, AMIA Annu Symp Proc.

[192] Altova 2010. SemanticWorks Semantic Web tool - Visual RDF and OWL editor, www.altova.com/semanticworks.html

[193] Böttcher, J. 2009. Diploma Thesis: Semantische Datenflussannotation in der Geschäftsprozessmodellierung. Freie Universität Berlin.

[194] Object Management Group 2008. Specification: The Business Process Modelling Notation (BPMN 1.1).

[195] van der Aalst, W. 1999. Formalization and Verification of Event-driven Process Chains. Information & Software Technology, vol. 41, pp. 10, pp. 639-650.

[196] Hepp, M and Roman, D. 2007. An Ontology Framework for Semantic Business Process Management. Proceedings of Wirtschaftsinformatik 2007.

[197] Semtation GmbH. 2008. SemTalk., www.semtalk.com

[198] Microsoft 2007. Visio 2007, http://office.microsoft.com/en-us/visio/

[199] European Commission 2007. Integrated Project: Semantic Utilised for Process Management within and between Enterprises (SUPER). 6th Framework Programme, www.ip-super.org

[200] Ma, Z. et al. 2007. Semantic business process management: A lifecycle based requirements analysis. In Proc. of Workshops on Semantic Business Process and Product Lifecycle Management (SBPM 2007) at the 4th European Semantic Web Conference (ESWC 2007).

[201] 1 a) matching architecture adapted from: Engmann, D. and Massmann, S. 2007. Instance Matching with COMA++. BTW 2007 Workshop: Model Management und Metadaten-Verwaltung.

[202] van Harmelen, F. 2006. Semantic Web Research anno 2006: main streams, popular fallacies, current status and future challenges. Proceedings CIA-2006. Number 4149 in Lecture Notes in Artificial Intelligence.

[203] Euzenat, J. and Shavaiko, P. 2007. Chapter 11 Conclusion in Book: Ontology Matching, Springer, pp. 274.

[204] Euzenat, J. and Shavaiko, P. 2007. Book: Ontology Matching, Springer.

[205] Falconer, S. M., Noy, N. F. and Storey, M. 2007. Ontology Mapping - A User Survey. 2nd International Workshop on Ontology Matching.

[206] O'Connor, M. et al. 2005. Writing Rules for the Semantic Web Using SWRL and Jess. Protégé With Rules workshop collocated with 8th International Protégé Conference.

[207] Horrocks, I. et al. 2004. SWRL: A Semantic Web Rule Language. Combining OWL and RuleML. W3C Member Submission.

[208] Ressler, J. 2007. Application of Ontology Translation. Proceedings of the Sixth International Semantic Web Conference (ISWC2007).

[209] BBN Technologies 2007. Snoggle - A Graphical, SWRL-based Ontology Mapper.

[210] Burnstein, I. 2003. Book: Practical Software Testing. Springer.

[211] Spillner, A. and Linz, T. 2005. Basiswissen Softwaretest. 3rd edition. dpunkt.verlag.

[212] Brügge, B., Dutoit, A. 2004. Book: Objektorientierte Softwaretechnik mit UML, Entwurfsmuster und Java. Pearson Studium.

[213] Paschke, A. et al. 2006. On Self-Validating Rule Bases. Proceedings of 2nd International Workshop on Semantic Web Enabled Software Engineering (SWESE 2006).

[214] Vrandecic, D. and Gangemi, A. 2006. Unit tests for ontologies. Proceedings of the 1st International Workshop on Ontology content and evaluation in Enterprise. Springer.

[215] Ontoprise 2007. OntoStudio User Manual. Version 2.0.

[216] Ding, Y. and Fensel, D. 2001. Ontology library systems: The key to successful ontology re-use. In Int. Semantic Web Working Symposium (SWWS 2001).

[217] Carroll, J. and Roo, J. D. 2004. OWL Web Ontology Language Test Cases. W3C Recommendation.

[218] Paschke, A 2005. The Contract Log Approach Towards Test-driven Verification and Validation of Rule Bases – A Homogeneous Integration of Test Cases and Integrity Constraints into Evolving Logic Programs and Rule Markup Languages (RuleML). Technical Report 10/2005. IBIS. Technical University Munich.

[219] Evan der Vlist, E. 2002. XSLTunit, http://xsltunit.org

[220] Altova 2007. XMLSpy XSLT Debugger, www.altova.com/xmlspy/xslt-debugger.html

[221] Kerrigan, M. et al. 2007. The Web Service Modeling Toolkit - An Integrated Development Environment for Semantic Web Services. Proceedings of the 4th European Semantic Web Conference (ESWC). System Description Track.

[222] Heß, A., Johnston, E. and Kushmerick, N. 2004. ASSAM: A Tool for Semi-Automatically Annotating Semantic Web Services. Lecture Notes in Computer Science. 3rd International Semantic Web Conference (ISWC 2004).

[223] Sirin, E. and Parsia, B. 2004. The OWL-S Java API. Proceedings of the 3rd International Semantic Web Conference (ISWC 2004).

[224] Fluegge, M. and Barnickel, N. et al. 2008. Semantic Interoperability – Report about Technological State-of-the-Art. Working Document wd3.2.1c. European Commission Integrated Project: QualiPSo.

[225] Fluegge, M. and Barnickel, N. et al. 2008, Working document 3.2.3 – Work package Semantic Interoperability. European Commission Integrated Project: QualiPSo.

[226] European Commission 2004. IST-1-002104-STP, SATINE (Semantic-based Interoperability Infrastructure for Integrating Web Service Platforms to Peer-to- Peer-Networks) project.

[227] Flügge, M. and Tourtchaninova, D. 2004. Ontology-derived Activity Components for Composing Travel Web Services. International Workshop on Semantic Web Technologies in Electronic Business (SWEB2004).

[228] Dogac, A. et al. 2004. Semantically Enriched Web Services for Travel Industry. Sigmod Record.

[229] Bicer, V. 2004. OWLmt - OWL Mapping Tool Technical Report. SRDC-METU.

[230] Fluegge, M. 2008. PhD Thesis: Adaptive Service Composition in a Semantics-based Interoperability Infrastructure. Technical University Berlin.

[231] Ambler, S. W. 2005. The Agile Unified Process (AUP), www.ambysoft.com/unifiedprocess/agileUP.html

[232] Kruchten, P. et al. 1998. Book: The Rational Unified Process: An Introduction, Addison-Wesley.

[233] Xiaofan, L., Böttcher, J. and Barnickel, N.2008. Enterprise Ontology-oriented Business Process Modeling – Tool Evaluation Study. Seminar Paper. OKS Group of Computer Science Department Technical University Berlin in Cooperation with Fraunhofer Institute FOKUS.

[234] Decker, G., Overdick, H. and Weske, M. 2008. Oryx - An Open Modeling Platform for the BPM Community. BPM. Volume 5240 of LNCS, pp. 382-385, Springer.

[235] World Wide Web Consortium 2003. Scalable Vector Graphics (SVG) 1.1 Specification. W3C Recommendation.

[236] World Wide Web Consortium. 2004. OWL Web Ontology Language Reference. 3.1.2 Property restrictions, W3C Recommendation.

[237] Motik, B. 2008. Reasoning in KAON2, http://kaon2.semanticweb.org/#reasoning

[238] Mindswap 2008. Pellet Reasoner. Release version 1.5.2, http://pellet.owldl.com/

[239] Protégé Community of Practice 2008. Protégé Wiki Article: SWRLExtensionsBuiltIns, http://protege.cim3.net/cgi-bin/wiki.pl?SWRLExtensionsBuiltIns

[240] Clark & Parsia. 2008. FAQ - Does Pellet support rules?, http://pellet.owldl.com/faq/rules

[241] Stanford Center for Biomedical Informatics Research 1997. Protégé-OWL editor. http://protege.stanford.edu/overview/protege-owl.html

[242] Friedman-Hill, E. 2008. The Jess Rule Engine. Sandia National Laboratories, http://herzberg.ca.sandia.gov/jess/

[243] Protégé Community of Practice 2008. Protégé Wiki Article: SWRL-JessBridge, http://protege.cim3.net/cgi-bin/wiki.pl?SWRLJessBridge

[244] Protégé Community of Practice 2008. Protégé Wiki Article: SWRLFactoryFAQ. How do I create a rule using the SWRL Factory?, http://protege.cim3.net/cgi-bin/wiki.pl?SWRLFactoryFAQ#nid6N9

[245] Barnickel, N, Böttcher, J. and Paschke, A. 2010. Semantic Mediation of Information Flow in Cross-Organizational Business Process Modeling. Proceedings Semantic Business Process Management Workshop at 7th Extended Semantic Web Conference (ESWC2010).

[246] Lausen, H., Zaremba, M. and Petrie, C. 2008. The Semantic Web Service Challenge. Scenario: Purchase Order Mediation, http://sws-challenge.org/wiki/index.php/Scenario:_Purchase_Order_Mediation

[247] Fluegge, M. and Barnickel, N. et al. 2008, Working document 3.2.4 – Semantic Interoperability Scenarios and Experiments. European Commission Integrated Project: QualiPSo.

[248] Brun, M. K. and Nielsen, B. 2003. Naming and Design Rules for eGovernment - The Danish Approach. In Proceedings of XML 2003.

[249] Danish Ministry of Science, technology and Innovation 2005. Danish e-Government Project: Infostructurebase, http://en.itst.dk/it-architecture-standards/data-standardisation/infostructurebase/infostructurebase

[250] Deutschland Online Standardisierung 2006. OSCI-XÖV – XML Standards für die öffentliche Verwaltung, www.xoev.de

[251] Active Endpoints 2008. ActiveBPEL. Release: ActiveBPEL-engine 4.1, www.activevos.com/community-open-source.php

[252] Apache Software Foundation 2006. Apache ODE (Orchestration Director Engine), http://ode.apache.org/

[253] Oracle 2005. Oracle BPEL Process Manager, www.oracle.com/technology/products/ias/bpel

[254] Active Endpoints. 2007. Creating and Using Custom XPath Functions, www.activebpel.org/samples/samples-4/ custom_functions/doc/index.html

[255] Kelly Thompson. Apache ODE FAQ - Language-Related BPEL Questions, http://ode.apache.org/faq.html

[256] Antony Reynolds. 2007. Antony Reynolds Blog - Extending XPath, http://blogs.oracle.com/reynolds/2007/07/18/

[257] Codehaus. Jaxen: Universal Java XPath Engine, http://jaxen.codehaus.org

[258] BerliOS 2010. Open Source Software Development Platform. Fraunhofer Institute FOKUS. Berlin, www.berlios.de

[259] Barnickel, N and Böttcher, J. 2009. Project: Semantic Dataflow for BPM, http://developer.berlios.de/projects/semext4oryx/

[260] Barnickel, N and Antonenko, L. 2008. Project: Semantic Mapping Testing, http://developer.berlios.de/projects/mappingtesting/

[261] Barnickel, N and Weinand, R. 2008. Project: SWSComposer4BPEL, http://developer.berlios.de/projects/swsc4bpel/

[262] Barnickel, N and Weinand, R. 2008. Project: Semantic BPEL Engine Extension, http://developer.berlios.de/projects/sembpelext/

[263] Association of German Chambers of Industry and Commerce (DIHK). 2008. About Us, www.dihk.de/english/aboutus/overview.pdf

[264] Chappell, D. 2004. Book: Enterprise Service Bus. Publisher: O'Reilly Media Inc.

[265] Bremen Online Services 2001. OSCI – The secure communication standard for E-Government, www.osci.de/materialien/englisch.pdf

[266] Bundesministerium des Inneren 2009. XÖV-Handbuch. Deutschland Online Standardisierung, www.xoev.de/sixcms/media.php/13/2010-03-02-Handbuch-final.pdf

[267] XÖV AG Datenkonferenz. 2007, www.standardisierung.deutschland-online.de

[268] Deutschland Online Standardisierung 2009. XÖV-Kernkomponenten-Bibliothek. Deutschland Online XRepository, www.xrepository.deutschland-online.de

[269] Deutschland Online Standardisierung 2007. Bericht der XÖV-AG „Datenkonferenz" zur Erprobung der XÖV-Kernkomponenten-Bibliothek, p.3

[270] OSCI-Leitstelle 2008. Interoperabilität von XÖV-Standards der Innenverwaltung – Bericht an den AK I der IMK, Version 1.1.

[271] Deutschland Online Standardisierung KoopA-ADV 2006. XÖV-Framework V1.0. Leitlinien für die XÖV-Standardisierung.

[272] Barnickel, N. and Fluegge, M. 2010. Towards a Conceptual Framework for Semantic Interoperability in Service-Oriented Architectures. Proceedings of ACM International Conference on Intelligent Semantic Web – Services and Applications (ISWSA2010).

[273] Barnickel, N. and Fluegge, M. 2010. Transferring the Principle of Loose Coupling to the Semantic Level. Proceedings of ACM International Conference on Intelligent Semantic Web – Services and Applications (ISWSA2010).

[274] European Commission 2008. The Semantic Interoperability Centre Europe - SEMIC.EU, www.semic.eu

[275] Klaus Reichling et al. 2008. Orientation Paper: IDABC XML Clearinghouse: SEMIC.EU.

[276] Hosoya, H., Frisch, A. and Castagna, G. 2009. Parametric polymorphism for XML. ACM Trans. Program. Lang. Syst., vol. 32, no. 1, pp 1-56.

[277] World Wide Web Consortium 2007. Semantic Annotations for WSDL and XML Schema. W3C Recommendation.

[278] Akhtar, W., Kopecký, J., Krennwallner, T. and Polleres, A. 2008. XSPARQL: Traveling between the XML and RDF worlds – and avoiding the XSLT pilgrimage. Proceedings of 5th European Semantic Web Conference, vol. 5021 of LNCS, Springer.

[279] The World Wide Web Consortium 2004. OWL Web Ontology Language Reference. 1.3 OWL syntax. W3C Recommendation, www.w3.org/TR/owl-ref/#Syntax

[280] The World Wide Web Consortium 2004. SWRL: A Semantic Web Rule LanguageCombining OWL and RuleML. 1 2.2. Human Readable Syntax. W3C Member Submission, www.w3.org/Submission/SWRL/#2.2

[281] European Commission 2010. SEMIC.EU Core Person: Final Draft.The Semantic Interoperability Centre Europe, www.epractice.eu/en/news/330117

[282] Barnickel, N., Böttcher, J. and Paschke, A. 2010. Incorporating Semantic Bridges into Information Flow of Cross-Organizational Business Process Models. Proceedings of the 6th International Conference on Semantic Systems (I-Semantics2010).

[283] Folmer, E. and Bastiaans, J. 2008. Quality of Electronic Messaging Standards' Specifications. Proceedings of 14th International Conference on Concurrent Enterprising (ICE2008)

[284] World Wide Web Consortium 2004. Web Services Glossary. W3C Working Group Note 11, www.w3.org/TR/ws-gloss/#choreography

[285] Decker, G. 2009. PhD Thesis: Design and Analysis of Process Choreographies. Hasso Plattner Institute, University of Potsdam.

Appendix

Sample Documents:

Domain Ontology Sample “RosettaNetOntology”

(excerpt in RDF/XML Syntax [279])

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns="http://localhost:8080/ontologies/RosettaNetOntology.owl#"
                xmlns:rosetta="http://localhost:8080/ontologies/RosettaNetOntology.owl#"
                xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
                xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
                xmlns:owl="http://www.w3.org/2002/07/owl#"
                xml:base="http://localhost:8080/ontologies/RosettaNetOntology.owl">
  <owl:Ontology rdf:about="http://localhost:8080/ontologies/RosettaNetOntology.owl"/>
  <owl:Class rdf:ID="Partner">
    <owl:equivalentClass>
      <owl:Class>
       <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
         <owl:onProperty><owl:FunctionalProperty rdf:ID="hasBusinessDescription"/></owl:onProperty>
         <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
        </owl:Restriction>
        <owl:Restriction>
         <owl:onProperty><owl:FunctionalProperty rdf:ID="hasContactInformation"/></owl:onProperty>
         <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
        </owl:Restriction>
        <owl:Restriction>
        <owl:onProperty><owl:FunctionalProperty rdf:ID="hasPhysicalAddress"/></owl:onProperty>
        <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
        </owl:Restriction>
      </owl:intersectionOf>
     </owl:Class>
    </owl:equivalentClass>
  </owl:Class>
...
  <owl:FunctionalProperty rdf:about="#hasBusinessDescription">
   <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
   <rdfs:range rdf:resource="#BusinessDescription"/>
 </owl:FunctionalProperty>
...
  <owl:Class rdf:ID="BusinessDescription ">
   <owl:equivalentClass>
     <owl:Restriction>
        <owl:onProperty><owl:FunctionalProperty rdf:ID="hasBusinessName"/></owl:onProperty>
        <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
      </owl:Restriction>
    </owl:equivalentClass>
   </owl:Class>
...
 <owl:FunctionalProperty rdf:about="#hasBusinessName">
   <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
   <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
```

```
  </owl:FunctionalProperty>
…
</rdf:RDF>
```

Domain Ontology Sample “MoonOntology”

(excerpt in RDF/XML Syntax [279])

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns="http://localhost:8080/ontologies/MoonOntology.owl#"
               xmlns:moon="http://localhost:8080/ontologies/MoonOntology.owl#"
               xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
               xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
               xmlns:owl="http://www.w3.org/2002/07/owl#"
               xml:base="http://localhost:8080/ontologies/MoonOntology.owl">
   <owl:Ontology rdf:about="http://localhost:8080/ontologies/MoonOntology.owl"/>
   <owl:Class rdf:ID="Customer">
     <owl:equivalentClass>
       <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
         <owl:Restriction>
          <owl:onProperty><owl:FunctionalProperty rdf:ID="hasCustomerInfo"/></owl:onProperty>
          <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
         </owl:Restriction>
         <owl:Restriction>
          <owl:onProperty><owl:FunctionalProperty rdf:ID="hasPostalAddress"/></owl:onProperty>
          <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
         </owl:Restriction>
       </owl:intersectionOf>
      </owl:Class>
     </owl:equivalentClass>
   </owl:Class>
…
   <owl:FunctionalProperty rdf:about="#hasCustomerInfo">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:range rdf:resource="#CustomerInfo"/>
  </owl:FunctionalProperty>
…
   <owl:Class rdf:ID="CustomerInfo ">
    <owl:equivalentClass>
     <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
       <owl:Restriction>
         <owl:onProperty><owl:FunctionalProperty rdf:ID="hasBusinessName"/></owl:onProperty>
         <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
       </owl:Restriction>
       <owl:Restriction>
         <owl:onProperty><owl:FunctionalProperty rdf:ID="hasEmail"/></owl:onProperty>
         <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</owl:cardinality>
       </owl:Restriction>
…
      </owl:intersectionOf>
     </owl:Class>
    </owl:equivalentClass>
   </owl:Class>
…
  <owl:FunctionalProperty rdf:about="#hasBusinessName">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
…
```

```
</rdf:RDF>
```

Semantic Bridge Sample "RosettaNetOntology2MoonOntology"

(excerpt in Human Readable Syntax [280])

SWRL Rule "BusinessDescription&ConatctInformation2CustomerInfo"

rosetta:**Partner** (?partner) ∧

rosetta:**hasBusinessDescription** (?partner, ?businessDesc) ∧

rosetta:**hasBusinessName** (?businessDesc, **?businessName**) ∧

rosetta:**hasConatctInformation** (?partner, ?contactInfo) ∧

rosetta:**hasEmailAddress** (?contactInfo, **?email**) ∧

swrlx:**makeOWLThing** (?newCustomerInfo, ?partner)

⇒

moon:**hasCustomerInfo** (?partner, ?newCustomerInfo) ∧

moon:**hasBusinessName** (?newCustomerInfo, **?businessName**) ∧

moon:**hasEmail** (?newCustomerInfo, **?email**) ∧

rdfs:**label** (?partner, "firedRule_BusinessDescription&ConatctInformation2CustomerInfo ")

SWRL Rule "PhysicalAddress2PostalAddress"

(in Human Readable Syntax):

rosetta:**PhysicalAddress** (?physicalAddress) ∧

rosetta:**hasAddressLine1** (?physicalAddress, **?addressLine1**) ∧

rosetta:**hasCityName** (?physicalAddress, ?city) ∧

rosetta:**hasGlobalCountryCode** (?physicalAddress, ?countryCode) ∧

rosetta:**hasNationalPostalCode** (?physicalAddress, ?postalCode)

⇒

moon:**hasStreet** (?physicalAddress, **?addressLine1**) ∧

moon:**hasCity** (?physicalAddress, ?city) ∧

moon:**hasCountryCode** (?physicalAddress, ?countryCode) ∧

moon:**hasPostalCode** (?physicalAddress, ?postalCode) ∧

rdfs:**label** (?physicalAddress,"firedRule_PhysicalAddress2PostalAddress")

Semantic Web Service Sample “MoonCRMService”

(excerpt in RDF/XML Syntax [279])

```
<?xml version="1.0" encoding="WINDOWS-1252"?>
<rdf:RDF xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
        xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns="http://www.example.org/service.owl"
        xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
        xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
        …
        xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
        xml:base="http://localhost:8080/SemanticWebServices/MoonServices/MoonCRMService.owl#">
<owl:Ontology rdf:about="">
        <owl:imports rdf:resource="http://localhost:8080/ontologies/MoonOntology.owl"/>
</owl:Ontology>
<!-- Service description -->
<service:Service rdf:ID="MoonCRMServiceService">
        <service:presents rdf:resource="#MoonCRMServiceProfile"/>
        <service:describedBy rdf:resource="#MoonCRMServiceProcess"/>
        <service:supports rdf:resource="#MoonCRMServiceGrounding"/>
</service:Service>
<!-- Profile description-->
<profile:Profile rdf:ID="MoonCRMServiceProfile">
        <profile:serviceName xml:lang="en">MoonCRMService</profile:serviceName>
         <profile:textDescription xml:lang="en">
                Looks up a Customer in a CRM and returns an IdentifiedCustmer.
        </profile:textDescription>
        <profile:hasInput rdf:resource="#CustomerLookupRequest"/>
        <profile:hasOutput rdf:resource="#CustomerLookupResponse"/>
</profile:Profile>
<!-- Process description -->
<process:AtomicProcess rdf:ID="MoonCRMServiceProcess">
        <rdfs:label>MoonCRMServiceProcess</rdfs:label>
        <process:hasInput rdf:resource="#CustomerLookupRequest"/>
        <process:hasOutput rdf:resource="#CustomerLookupResponse"/>
</process:AtomicProcess>
<process:Input rdf:ID="CustomerLookupRequest">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
                http://.../MoonCRMService.owl#CustomerLookupRequest
        </process:parameterType>
         <rdfs:label>CustomerLookupRequest</rdfs:label>
</process:Input>
<process:Output rdf:ID="CustomerLookupResponse">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
                http://.../MoonCRMService.owl#CustomerLookupResponse
        </process:parameterType>
        <rdfs:label>CustomerLookupResponse</rdfs:label>
</process:Output>
<!-- Grounding description -->
<grounding:WsdlGrounding rdf:ID="MoonCRMServiceGrounding">
…
<grounding:WsdlAtomicProcessGrounding rdf:ID="MoonCRMServiceAtomicProcessGrounding">
  <grounding:owlsProcess rdf:resource="#MoonCRMServiceProcess"/>
  <grounding:wsdlDocument rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
        http://sws-challenge.org/services/CRMService?wsdl
  </grounding:wsdlDocument>
  <grounding:wsdlOperation>
        <grounding:WsdlOperationRef>
        <grounding:portType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
                http://sws-challenge.org/services/CRMService?wsdl#CRMServicePortType
        </grounding:portType>
        <grounding:operation rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
```

```
                http://sws-challenge.org/services/CRMService?wsdl#search
        </grounding:operation>
        </grounding:WsdlOperationRef>
</grounding:wsdlOperation>
<grounding:wsdlInputMessage rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
        http://sws-challenge.org/services/CRMService?wsdl#SearchCustomerRequestMessage
</grounding:wsdlInputMessage>
<grounding:wsdlInput>
        <grounding:WsdlInputMessageMap>
        <grounding:owlsParameter rdf:resource="#CustomerLookupRequest"/>
        <grounding:wsdlMessagePart rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
                http://sws-challenge.org/services/CRMService?wsdl#SearchCustomerRequest
        </grounding:wsdlMessagePart>
        <grounding:xsltTransformationString>
   <![CDATA[ <xsl:stylesheet version="1.0" xmlns:moon="http://localhost:8080/ontologies/MoonOntology.owl#"
                …
                <xsl:template match="/">
                        <searchString>
                                <xsl:value-of select="/rdf:RDF/rdf:Description/moon:hasBusinessName"/>
                        </searchString>
                </xsl:template>
                </xsl:stylesheet> ]]>
        </grounding:xsltTransformationString>
        </grounding:WsdlInputMessageMap>
</grounding:wsdlInput>
<grounding:wsdlOutputMessage rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
        http://sws-challenge.org/services/CRMService?wsdl#SearchCustomerResponseMessage
</grounding:wsdlOutputMessage>
<grounding:wsdlOutput>
        <grounding:WsdlOutputMessageMap>
        <grounding:owlsParameter rdf:resource="#CustomerLookupResponse"/>
        <grounding:wsdlMessagePart rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
                http://sws-challenge.org/services/CRMService?wsdl#SearchCustomerResponse
        </grounding:wsdlMessagePart>
        <grounding:xsltTransformationString>
        <![CDATA[<xsl:stylesheet version="1.0" … xmlns:xmoon="mooncompany">
                <xsl:template match="/">
                 <rdf:RDF
                        xmlns:moon="http://localhost:8080/ontologies/MoonOntology.owl#"
                        …
                        xmlns="http://localhost:8080/process/MediationProcess/MoonCRMServiceOutput.owl#"
                        xml:base="http://localhost:8080/process/MediationProcess/MoonCRMServiceOutput.owl">
                        <moon:CustomerLookupResponse rdf:ID="CustomerLookupResponse_X">
                        <moon:hasCustomer>
                        <moon:IdentifiedCustomer rdf:ID="Customer_X">
                        <moon:hasCustomerID rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                                <xsl:value-of select="/xmoon:SearchCustomerResponse/xmoon:customerId"/>
                        </moon:hasCustomerID>
                        <moon:hasBusinessName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                                <xsl:value select="/xmoon:SearchCustomerResponse/xmoon:businessName"/>
                        </moon:hasBusinessName>
                        </moon:IdentifiedCustomer>
                        </moon:hasCustomer>
                        </moon:CustomerLookupResponse>
                </rdf:RDF>
                </xsl:template>
        </xsl:stylesheet>]]>
        </grounding:xsltTransformationString>
        </grounding:WsdlOutputMessageMap>
   </grounding:wsdlOutput>
  </grounding:WsdlAtomicProcessGrounding>
</rdf:RDF>
```