
Relevance and Applicability of Semantic Web Services in Electronic Business: A Delphi Study

(Bestimmung des Potentials von Semantic Web Services
für die elektronische Geschäftsabwicklung)

Diplomarbeit

Zur Erlangung des akademischen Grades
eines Magisters der Sozial- und Wirtschaftswissenschaften

Eingereicht bei
Univ.-Prof. Dr. Kerstin Fink

Institut für Wirtschaftsinformatik, Produktionswirtschaft und Logistik
Fakultät für Betriebswirtschaft der Universität Innsbruck

Von
Daniel Bachlechner, Bakk.

Innsbruck, August 2007

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Diplomarbeit selbständig angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher weder in gleicher noch in ähnlicher Form einer andern Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

.....

Innsbruck, August 2007

Kurzbeschreibung

Web Services lautet das neue Schlagwort, wenn über die Weiterentwicklung des World Wide Web als Infrastruktur für unternehmensübergreifende Anwendungen, aber auch als Lösung zur Integration interner Anwendungen diskutiert wird. Web Services zielen nicht nur auf isolierte Lösungen ab, sondern ermöglichen die Implementierung von weltweit verfügbaren Diensten, die entweder direkt vom Endanwender genutzt oder in andere Anwendungen integriert werden können. Sie bilden eine plattform- und sprachunabhängige, technische Grundlage für die Realisierung von serviceorientierten Architekturen, in denen Anwendungen aus weltweit verfügbaren Diensten zusammengebaut werden. Unter Semantic Web Services versteht man Web Services, die neben einer rein syntaktischen Schnittstellenbeschreibung auch mit semantischen Inhalten angereichert werden, um ihr Auffinden, Auswählen, Kombinieren und Ausführen, sowie diverse andere Aspekte zu erleichtern. Im Rahmen dieser Arbeit wird, basierend auf den Ergebnissen einer im ersten Quartal 2007 durchgeführten Delphi-Studie, das Potential von Semantic Web Services für die elektronische Geschäftsabwicklung untersucht.

Da die Zukunft von Semantic Web Services im Hinblick auf die Integration von Anwendungen im betrieblichen Umfeld nicht nur für den deutschsprachigen Raum von Interesse ist und die Studie, aufgrund der Mitarbeit von Experten aus allen Teilen der Welt ohnehin in englischer Sprache durchgeführt werden musste, lag der Entschluss nahe die gesamte Arbeit in Englisch zu verfassen.

Abstract

Web services have revolutionized the enterprise integration field by taking a remarkable step towards the seamless integration of distributed software components. The importance of Web services in this regard is recognized and widely accepted by experts from industry and academia. Nevertheless, current Web service technologies operate at a syntactic level and therefore still require human intervention to a large extent. Semantic Web services pledge the automation of core Web service tasks such as discovery, selection, composition and execution, thus enabling seamless interoperation between systems, keeping human intervention to a minimum. The main question discussed within the scope of this work is whether Semantic Web services will play a significant role in the future of e-business. Of particular interest is the potential of Semantic Web services with respect to their application as basis of enterprise integration architectures. The discourse is based on the results of a Delphi study conducted in early 2007, attaching particular importance on differences in the viewpoints of experts from industry and academia, respectively.

Acknowledgment

My thanks go to my advisor, Univ.-Prof. Dr. Kerstin Fink, head of the Department of Information Systems, Operations Management and Logistics at the University of Innsbruck, for offering me a thesis on the relevance and applicability of Semantic Web services (SWSs) in e-business under her supervision, and for guiding and supporting me during my work. Despite all the work she has to do to keep the department running, she was available to answer my questions and give me valuable suggestions. I am also extremely grateful to MMag. Alexander Hörbst, former assistant at the Department of Information Systems, Operations Management and Logistics and now affiliated with the University for Health Sciences, Medical Informatics and Technology, for his instruction and patient explanations regarding methodological questions at the beginning of my research.

I also thank Dr. Sigurd Harand at the Digital Enterprise Research Institute (DERI) at the University of Innsbruck, a leading research institute focusing on Semantic Web technologies, for his valuable suggestions and profound advice while I worked to confine the scope of the thesis. With his wide experience in the field of research, he was able to give me many valuable inside tips. I am also grateful for his insight on the DIP project. Thanks also go to Univ.-Prof. Dr. Martin Hepp and Univ.-Prof. Dr. Marcus Spies for supporting my work, particularly during the design process of the field study, with helpful comments on the structuring and formalization of the questionnaires that were so important to the success of my research. I am grateful to Mag. Armin Haller from DERI at the

ACKNOWLEDGMENT

National University of Ireland, Galway, for his assistance during the selection of panelists and for spending hours discussing critical questions related to my research.

Special thanks also go to the experts who made valuable contributions to the field study on the relevance and applicability of integration architectures based on SWSs. Without the foundation they built, it would not have been possible to assess the potential of an emerging technology such as SWSs satisfactorily.

Love, sympathy and encouragement from my parents and my girlfriend Claudia enabled me to face the challenging task of researching and writing this thesis. They deserve my special thanks.

Contents

Abbreviations and Acronyms	V
Figures.....	IX
Tables	XI
1 Introduction.....	1
1.1 Problem Statement	1
1.2 Outline	3
2 Electronic Business	5
2.1 Business Relevance.....	7
2.1.1 Key Drivers.....	7
2.1.2 Applications Areas.....	9
2.2 Distribution Mechanisms and Middleware	10
2.2.1 Decision Criteria	11
2.2.2 Integration Technologies	14
2.3 Types of Integration.....	22
2.3.1 Value-Added Networks	23
2.3.2 Hub-and-Spoke versus Bus.....	23
2.3.3 B2B Integration over the Internet	24

2.3.4	Transformation Hubs	26
2.3.5	Hosted Application Integration.....	26
2.3.6	Marketplace Integration	28
2.4	Selected Integration Products.....	28
2.4.1	WebSphere	29
2.4.2	BizTalk Server	29
2.4.3	WebLogic Integration	30
2.4.4	NetWeaver	30
3	Semantic Web Services.....	33
3.1	Constituent Technologies.....	35
3.1.1	Web Services.....	35
3.1.2	Semantic Web	39
3.2	Infrastructure and Usage	43
3.2.1	Usage Activities and Process	43
3.2.2	Architecture.....	48
3.2.3	Service Ontology	48
3.3	Significant Frameworks	49
3.3.1	OWL-S	50
3.3.2	WSMO	52
3.4	Potential Applications	55
3.4.1	Business Process Management	56
3.4.2	Content Syndication.....	57
3.4.3	Enterprise Collaboration	58
3.4.4	Search and Mining	59
4	SWS-enabled E-Business.....	61
4.1	Literature Review.....	61
4.2	Business Scenarios and Case Studies.....	64
4.2.1	ATHENA	65
4.2.2	DIP	68
4.3	Trends.....	70

5	Field Study	71
5.1	Research Approach.....	71
5.2	Survey Design.....	74
5.2.1	Questionnaire Construction	75
5.2.2	Expert Recruitment	78
5.2.3	Implementation Strategy and Correspondence	80
5.2.4	Technical Realization	96
5.3	Data Analysis.....	111
5.4	Results.....	112
5.4.1	SWOT Analysis	113
5.4.2	Requirements	126
5.4.3	Expectations.....	133
5.4.4	Roadmap	146
5.5	Feedback and Comments	157
6	Discussion	159
6.1	General Vision	159
6.2	Business Implications	160
6.3	Research Trends and Industry Needs.....	161
7	Conclusion	165
	Bibliography	169

Abbreviations and Acronyms

A2A	Application-to-Application
ACM	Association for Computing Machinery
API	Application Programming Interface
ASP	Application Service Provides
ATHENA	Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application
B2B	Business-to-Business
BI	Business Intelligence
BPM	Business Process Management
CORBA	Common Object Request Broker Architecture
CPD	Collaborative Product Development
CRM	Customer Relationship Management
CSS	Cascading Style Sheet
DAML	DARPA Markup Language
DCOM	Distributed Component Object Model
DERI	Digital Enterprise Research Institute

DIP	Data, Information, and Process Integration with Semantic Web Services
DL	Description Logics
EAI	Enterprise Application Integration
EDI	Electronic Data Interchange
ERP	Enterprise Resource Planning
FP7	Seventh Research Framework Programme
FTP	File Transfer Protocol
I-ESA	Interoperability for Enterprise Software and Applications Conference
ICWS	International Conference on Web Services
IEEE	Institute of Electrical and Electronics Engineers
ISWC	International Semantic Web Conference
IT	Information Technology
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
MAPPER	Model-based Adaptive Product and Process Engineering
MOM	Message-Oriented Middleware
OAGIS	Open Applications Group Integration Specification
OEM	Original Equipment Manufacturer
OWL	Web Ontology Language
OWL-S	OWL for Web Services
PHP	PHP: Hypertext Preprocessor
PIN	Personal Identification Number
PPM	Product Portfolio Management
RDF	Resource Description Framework

RDF-S	RDF Schema
RPC	Remote Procedure Call
RSS	Real Simple Syndication
SaaS	Software as a Service
SAWSDL	Semantic Annotations for WSDL
SCM	Supply Chain Management
SOA	Service-Oriented Architecture
SMTP	Simple Mail Transfer Protocol
SPSS	Statistical Package for the Social Sciences
SQL	Structured Query Language
SWOT	Strengths, Weaknesses, Opportunities and Threats
SWS	Semantic Web Service
SWWS	Semantic Web-enabled Web Services
TDM	Tailored Design Method
TPM	Trading Partner Management
UDDI	Universal Description, Discovery and Integration
URI	Universal Resource Identifier
VAN	Value-Added Network
VISP	Virtual Internet Service Provider
W3C	World Wide Web Consortium
WS-BPEL	Web Services Business Process Execution Language
WS-I	Web Services Interoperability
WSCPC	Web Service-based Coordinated Process Collaboration
WSDL	Web Service Definition Language
WSMF	Web Service Modeling Framework
WSML	Web Service Markup Language

WSMO	Web Service Modeling Ontology
WSMX	Web Service Modeling Execution Environment
WWW	International World Wide Web Conference
XHTML	Extensible HTML
XML	Extensible Markup Language
XML-S	XML Schema
ZIFA	Zachman Institute for Framework Advancement

Figures

Key drivers for e-business initiatives.....	8
Key application areas of e-business solutions.	10
File transfer.	15
Shared database.....	17
Remote procedure invocation.	19
Messaging.	21
B2B integration using EDI over VAN.....	23
A2A integration.	24
B2B integration over the Internet.	25
Hosted application integration.	27
Publications on SWSs since 2001.....	34
SWS technologies.	35
Web service standards stack.	37
Web service usage process.	38
Semantic Web standards stack.....	41
SWS infrastructure usage activities.	44
SWS infrastructure components.	48
OWL-S upper ontology.	51
Gantt chart of the survey.....	81
Invitation.	83
Pre-registration process.....	84

Registration confirmation.	84
Notification about the beginning of the first survey round.	85
First general reminder.	86
First round of the survey.	87
Reminder for participants who had not answered any questions.	88
Reminder for participants who had partially answered the questionnaire.	89
Second general reminder.	90
Second reminder for study participants.	91
Notification about the beginning of the second survey round.	92
Second round of the survey.	92
First reminder of the second survey round.	93
Second reminder of the second survey round.	94
Notification about the beginning of the feedback phase.	95
Feedback process.	95
Login page.	99
Files involved in the pre-registration phase.	100
Registration page.	101
Personal details page.	102
Files involved in the first round of the survey.	103
Upper part of the qualitative questionnaire page.	103
Lower part of the qualitative questionnaire page with some questions answered.	104
Question page.	105
Definitions page.	106
Files involved in the second round of the survey.	106
Quantitative questionnaire page.	107
Topic page.	108
Text field for comments on a topic page.	108
Schedule page.	109
Files involved in the feedback phase.	110
Feedback page.	110
Logout page during the first round.	111
Expertise distribution subdivided in backgrounds.	113
SWOT matrix.	166

Tables

Most important strengths of SWS-based integration architectures.	114
Most agreed strengths of SWS-based integration architectures.	114
Most controversial strengths of SWS-based integration architectures.	114
Most important strengths of SWS-based integration architectures from an academic perspective.	115
Most important strengths of SWS-based integration architectures from an industrial perspective.	116
Most controversial strengths of SWS-based integration architectures comparing the two groups of respondents.	116
Most important weaknesses of SWS-based integration architectures.	117
Most agreed weaknesses of SWS-based integration architectures.	117
Most controversial weaknesses of SWS-based integration architectures.	117
Most important weaknesses of SWS-based integration architectures from an academic perspective.	118
Most important weaknesses of SWS-based integration architectures from an industrial perspective.	119
Most controversial weaknesses of SWS-based integration architectures comparing the two groups of respondents.	119
Most important opportunities of SWS-based integration architectures.	120
Most agreed opportunities of SWS-based integration architectures.	120
Most controversial opportunities of SWS-based integration architectures.	120

Most important opportunities of SWS-based integration architectures from an academic perspective.	121
Most important opportunities of SWS-based integration architectures from an industrial perspective.	122
Most controversial opportunities of SWS-based integration architectures comparing the two groups of respondents.	122
Most important threats to SWS-based integration architectures.....	123
Most agreed threats to SWS-based integration architectures.....	123
Most controversial threats to SWS-based integration architectures.	123
Most important threats to SWS-based integration architectures from an academic perspective.	124
Most important threats to SWS-based integration architectures from an industrial perspective.	125
Most controversial threats to SWS-based integration architectures comparing the two groups of respondents.	125
Most important functional requirements that integration architectures must fulfill.	126
Most agreed functional requirements that integration architectures must fulfill.	126
Most controversial functional requirements that integration architectures must fulfill....	127
Most important functional requirements that integration architectures must fulfill from an academic perspective.....	128
Most important functional requirements that integration architectures must fulfill from an industrial perspective.....	128
Most controversial functional requirements that integration architectures must fulfill comparing the two groups of respondents.	129
Most important qualitative requirements that integration architectures must fulfill.	129
Most agreed qualitative requirements that integration architectures must fulfill.	130
Most controversial qualitative requirements that integration architectures must fulfill. ...	130
Most important qualitative requirements that integration architectures must fulfill from an academic perspective.....	131
Most important qualitative requirements that integration architectures must fulfill from an industrial perspective.....	131
Most controversial qualitative requirements that integration architectures must fulfill comparing the two groups of respondents.	132

Most important differences between internal and external integration architectures.....	132
Most agreed differences between internal and external integration architectures.....	133
Most controversial differences between internal and external integration architectures...	133
Most important positive effects of using SWS-based integration architectures at the macro level.....	134
Most agreed positive effects of using SWS-based integration architectures at the macro level.....	134
Most controversial positive effects of using SWS-based integration architectures at the macro level.....	135
Most important positive effects of using SWS-based integration architectures at the macro level from an academic perspective.....	135
Most important positive effects of using SWS-based integration architectures at the macro level from an industrial perspective.....	136
Most controversial positive effects of using SWS-based integration architectures at the macro level comparing the two groups of respondents.	137
Most important negative effects of using SWS-based integration architectures at the macro level.....	137
Most agreed negative effects of using SWS-based integration architectures at the macro level.....	138
Most controversial negative effects of using SWS-based integration architectures at the macro level.....	138
Most important negative effects of using SWS-based integration architectures at the macro level from an academic perspective.....	139
Most important negative effects of using SWS-based integration architectures at the macro level from an industrial perspective.....	139
Most controversial negative effects of using SWS-based integration architectures at the macro level comparing the two groups of respondents.	140
Most important positive effects of using SWS-based integration architectures at the micro level.	140
Most agreed positive effects of using SWS-based integration architectures at the micro level.	141
Most controversial positive effects of using SWS-based integration architectures at the micro level.	141

Most important positive effects of using SWS-based integration architectures at the micro level from an academic perspective.....	142
Most important positive effects of using SWS-based integration architectures at the micro level from an industrial perspective.....	142
Most controversial positive effects of using SWS-based integration architectures at the micro level comparing the two groups of respondents.	143
Most important negative effects of using SWS-based integration architectures at the micro level.....	143
Most agreed negative effects of using SWS-based integration architectures at the micro level.....	144
Most controversial negative effects of using SWS-based integration architectures at the micro level.....	144
Most important negative effects of using SWS-based integration architectures at the micro level from an academic perspective.....	145
Most important negative effects of using SWS-based integration architectures at the micro level from an industrial perspective.....	145
Most controversial negative effects of using SWS-based integration architectures at the micro level comparing the two groups of respondents.	146
Most important challenges SWS research is facing today.....	146
Most agreed challenges SWS research is facing today.....	147
Most controversial challenges SWS research is facing today.....	147
Most important challenges SWS research is facing today from an academic perspective.	148
Most important challenges SWS research is facing today from an industrial perspective.	148
Most controversial challenges SWS research is facing today comparing the two groups of respondents.	149
Achievements of SWS research to be made most likely within the next five years.....	150
Most agreed achievements of SWS research to be made within the next five years.....	150
Most controversial achievements of SWS research to be made within the next five years.	150
Achievements of SWS research to be made most likely within the next five years from an academic perspective.....	151

Achievements of SWS research to be made most likely within the next five years from an industrial perspective.....	152
Most controversial statements related to achievements in SWS research within the next five years comparing the two groups of respondents.....	152
Most important problems of current integration architectures that can be solved with SWSs.....	153
Most agreed problems of current integration architectures that can be solved with SWSs.....	153
Most controversial problems of current integration architectures that can be solved with SWSs.....	153
Most important problems of current integration architectures that can be solved with SWSs from an academic perspective.....	154
Most important problems of current integration architectures that can be solved with SWSs from an industrial perspective.....	155
Most controversial problems of current integration architectures that can be solved with SWSs comparing the two groups of respondents.	155
Functionality and usability of the survey system.....	158
Comparison of the most important functional requirements of the two groups of respondents.	162
Comparison of the most important qualitative requirements of the two groups of respondents.	162

Chapter 1

Introduction

The problem statement of this work is presented in section 1.1. Section 1.2 sketches the structure roughly and summarizes the basics of the individual chapters.

1.1 Problem Statement

In today's competitive and dynamic business environments, e-business components such as supply chain management (SCM), customer relationship management (CRM), e-commerce and business intelligence (BI) have become imperative for most enterprises. Many of these enterprises, however, employ multiple mission-critical, best-of-the-breed back-end application systems from different vendors with different technologies and platforms. It meant choosing the best vendor for every operational area and connecting the products via the interfaces they provided, typically point-to-point. However, this approach often led to highly complex systems. Nevertheless, until recently this strategy was considered a silver bullet when assembling business software.

Together with mergers and acquisitions, reorganizations, and leadership changes, which also constitute a great deal of impact on information technology (IT) infrastructures, the resulting best-of-the-breed solutions have led to extreme heterogeneity. Obviously, the operation of such patchworks is extremely complex and costly. The maintenance of numerous vendor relations and the necessity of specific know-how are usually not justifiable. However, the integration of enterprise application systems is essential to realize competitive advantages. Even if just a few critical systems cannot share their data

effectively, they create information bottlenecks that often require human intervention to be solved. Only with properly deployed integration architectures, enterprises can focus their efforts on their value creating core competencies.

The magnitude of the problem enterprises face with respect to e-business and in particular enterprise integration can be illustrated with the aid of data provided by the Zachman Institute for Framework Advancement¹ (ZIFA). It indicates that between 20% and 40% of all labor costs in the United States of America are dedicated to the gathering, storage and reconciliation of data, and that 70% of the lines of software code in enterprises are dedicated to moving data between systems. These values are likely to be valid for quite every developed economy worldwide.

Mission fulfillment and corporate success depend on the ability to effectively integrate systems. Today, e-business and enterprise integration are primarily accomplished with Web-based technologies. Their methods attempt to enable enterprises to make their applications, databases, enterprise information systems and business processes as interoperable as possible. An efficient and flexible integration architecture is necessary to work closely with trading partners and to better satisfy the needs and expectations of customers.

Recently, service-oriented architectures (SOAs) began to become increasingly popular with regard to integration. The term SOA defines an architecture in which applications call services from other applications. SOAs allow independent services, made available via internal or external networks, to be accessed without knowledge of their underlying implementation. The architecture is not tied to specific technologies and for years was implemented using a wide range of interoperability standards including CORBA (Common Object Request Broker Architecture) and DCOM (Distributed Component Object Model) [MaBe06]. Both concepts are based on interface definition languages and tightly coupled mechanisms. As opposed to implementations based on CORBA and DCOM, SOAs based on Web services represent a loosely coupled model that is independent from application platforms and programming languages as well as transport and message formats [ThSH04].

Web services have revolutionized the enterprise integration field by taking a remarkable step towards seamless integration of distributed software components using Web

¹ ZIFA is a network of professionals who exchange knowledge and experience concerning the use, implementation and advancement of enterprise architectures. For more information on ZIFA, see <http://www.zifa.com/>.

standards. The importance of Web services, in particular with respect to integration tasks, is recognized and widely accepted by experts from industry and academia [Myer02, Buss03]. Nevertheless, current Web service technologies operate at a syntactic level and therefore still require human intervention to a large extent.

More powerful approaches based on the application of semantic technologies have already been discussed in recent publications [Ober06, HKMV06]. Semantic technologies can be used to automate core Web service tasks such as discovery, selection, composition and execution, thus enabling seamless interoperation between systems, keeping human intervention to a minimum [McSZ01]. This boost in interoperability is expected to make increased profits, new capabilities and reduced costs possible [PoHo04]. Several initiatives addressing the problem of semantics in Web services have emerged in the past few years, but Semantic Web services (SWSs) have not yet been adopted by the industry. The term SWS describes the conjunction of Web service technologies and Semantic Web technologies. Although many challenges still must be addressed and solved, semantically enabled Web services seem to be the key to a next-generation integration architecture.

Today, the use of Semantic Web technologies to automate Web service tasks is a highly relevant research topic. This is particularly true because of its anticipated potential to achieve a more dynamic, scalable and cost-effective form of integration. The main question discussed within the scope of this work is whether SWSs will play a significant role in the future of e-business. Of particular interest is the potential of SWSs with respect to their application as basis of enterprise integration architectures. In the context of this work, we investigate factors affecting the relevance and applicability of SWSs, attaching particular importance to the differences in the opinions of experts with academic and industrial points of view, respectively.

1.2 Outline

This work consists of seven chapters. Their contents are sketched briefly in the following paragraphs.

Chapter 2 introduces the field of e-business from an integration point of view. The relevance of e-business is discussed as a starting point from two different angles: key drivers and application areas. The relevance discussion is followed by an introduction to distribution mechanisms and middleware. Afterwards, the spectrum of types of integration

that can be found in most enterprises is shown. We describe various scenarios including application-to-application (A2A), business-to-business (B2B), and hosted application integration. Finally, we briefly introduce some commercial off-the-shelf integration products from major vendors.

In Chapter 3, we focus on SWSs. We first introduce the two constituent technologies: Web services and the Semantic Web. After the introduction of the technologies the infrastructure required for the use of SWSs is discussed. In addition to usage activities, the architecture and the principles of services ontologies are described. An overview of the most significant SWS frameworks, OWL-S (Web Ontology Language for Web Services) and WSMO (Web Service Modeling Language), follows. We conclude this chapter with a concise assembly of application areas with particularly high potential for SWSs.

Insight into the first application attempts of SWSs in e-business is discussed in Chapter 4. First, selected publications related to the topic are sketched. This literature review is followed by summaries of business scenarios and case studies analyzed through the DIP and ATHENA projects. Finally, the trends identified are summed up.

Chapter 5 covers a field study conducted to determine the relevance and applicability of integration architectures based on SWSs. As starting points, we explain the research approach and detail the survey design. The description of the survey design contains information about the questionnaire, the sampling technique, the implementation strategy and the technical realization of the survey system. Afterwards, we outline the analysis methods briefly. Within the scope of the description of the survey results, we elaborate on every single question of the questionnaire extensively and consider different points of view. We conclude this chapter with a summary of feedback received from study participants.

In Chapter 6, the key results of the study are discussed under three different aspects. First, the findings are aggregated to formulate a general vision. Afterwards, we outline essential business implications and finally, we discuss the gap between academic research trends and industrial needs.

A short summary of the central ideas expressed within the scope of this work is provided in Chapter 7. Last but not least, we point out some interesting issues for future research in the field.

Chapter 2

Electronic Business

In [HeHR04], e-business is defined as any business process along the value chain supporting internal and external enterprise communication that relies on Internet technologies. Heinrich et al. emphasize that because most enterprises use diverging data processing systems, e-business applications require coordination among the participating parties including a unification of business rules, terms and concepts. Hence, within the scope of e-business integration is a fundamental need.

During the past couple of years the pressure on enterprises in all industries has constantly increased. To attain competitiveness, reduced costs had to be reconciled with investments in IT infrastructure and improvements in the services offered. At the same time, it was required to quickly respond to strategic business objectives. The heterogeneity of the system and application environment as well as the constantly changing market requirements are two of the major issues to get a grip on in this regard. According to [Whyt01], enterprises that can adopt integrated and platform-independent solutions in a secure environment will be successful in the long run.

At present, enterprises must find ways to deal with numerous emerging challenges more than ever. In [Wieh04] some challenges are listed:

- insufficient cost-effectiveness making it difficult to adapt to changing requirements;
- costly and inflexible integration technologies causing intolerable risks;

- monolithic applications implicating high adaptation and maintenance costs;
- dependence on software vendors;
- lack of security in complex and automated business processes;
- missing transparency and insufficient control of automated processes; and
- limited benefit from value networks owing to high complexity and deficient security infrastructure.

Today, in architectural approaches, the ability to get most of these challenges under control lies in SOAs, which are commonly based on Web services. SWSs are expected to help further automate SOAs. The applicability and relevance of integration architectures based on the SOA approach and SWSs is the main subject of this work.

Usually most departments in enterprises have their own computer systems optimized for the particular ways they do their work. Enterprise resource planning (ERP) combines them into an integrated software program that runs on a single database that allows the various departments to share information and communicate with each other more easily. In general, ERP software attempts to integrate all departments and functions across a company into a single computer system that can serve all departments' particular needs.

An integrated approach can have a tremendous payback if enterprises install the software correctly. ERP vanquishes the old standalone computer systems in an enterprise's departments and replaces them with a single unified software program divided into software modules that roughly approximate the old standalone systems. Finance, manufacturing and the warehouse, all still have their own views on the data, except that now the application systems are linked. In the 1990s, ERP was developed as a tightly integrated monolith, but most vendors' software has since become flexible enough that some modules can also be used without buying the whole package. ERP vendors also began opening their software to other systems recently.

Although ERP software offers advantages to enterprises, it has not achieved many of its anticipated benefits. Autonomous and heterogeneous applications coexist in enterprises with ERP systems and the integration need is bigger than ever. Nevertheless, integration remains to be the key to harvest the benefits of e-business.

The relevance of e-business for enterprises is discussed in section 2.1. We discuss the key drivers as well as the major application areas based on a recent study. In section 2.2, we focus on the distribution mechanisms and middleware. Section 2.3 illustrates a variety of integration types. Finally, in section 2.4, a selection of commercial off-the-shelf products is introduced briefly.

2.1 Business Relevance

Currently, customer satisfaction, operational efficiency and competitiveness are the key drivers for organizations looking to apply e-business solutions. The key drivers are the subject of section 2.1.1. In section 2.1.2, specific applications are discussed to show that e-business is being used for a range of functions, primarily order management.

2.1.1 Key Drivers

According to Comergent Technologies' most recent annual e-business survey [Come06], three main drivers are behind the application of e-business solutions:

- optimizing the customer experience;
- enhancing operational efficiency; and
- staying competitive.

For 68% of the survey respondents, customer experience is a key driver behind the deployment of e-business. As depicted in Figure 1, the same percentage of respondents is of the opinion that making business easier for customers plays a major role for the application of e-business solutions. Interestingly, cost reduction, which in the past topped the list, was one of the less popular drivers in the 2006 survey. The results suggest that today's customers expect quick and easy access to products and services with personalized interaction and direct links to trading partners involved in the e-business process. To avoid customer frustration, the underlying complexities of the business operations must be seamless and transparent, no matter what interaction or combination of interactions the customer chooses. The complexities may include product lines, service offerings, business units and built-to-order offerings as well as multiple back-end systems and processes. According to Comergent Technologies, e-business further optimizes the customer experience by offering customers and partners unique and personalized access to products

and services. This includes the support of quoting, pricing and selling processes to help customers buy the right products and services to best fit their needs.



Figure 1 Key drivers for e-business initiatives.

At nearly 60%, the second most important driver of e-business is the enhancement of operational efficiency. This means that enterprises turn to e-business to solve complex problems and align internal and external systems. E-business solutions accommodate these needs in three ways:

- e-business solutions manage, collect and syndicate product and service, process, and configuration information across multiple systems and enterprises, both internal and external;
- e-business solutions automate sales and services from the initial inquiry to invoice payment; and

- e-business solutions yield operational efficiency by capturing, distributing and fulfilling orders from multiple sources and sales channels and facilitating the integration by using multiple ERP systems.

Acquiring new markets and customer segments allows enterprises to grow their business models and stay competitive. Staying competitive is ranked as the third key driver for e-business, according to 57% of the survey respondents, who see e-business as a way to drive greater selling efficiency and effectiveness. E-business applications automate what enterprises sell and how they sell it while integrating sales and service touch points inside and outside the organization. Enterprises can benefit customers by creating and managing individual or multiple storefronts and customer self-service portals. Transaction-based storefronts or portals can seamlessly integrate multiple lines of business across the enterprise to present a common unified face to the customer. Furthermore, they can weave together multiple distribution and channel partners to enable and support multitiered channel e-business and alleviate channel conflicts.

As per Comergent Technologies, sales efficiency is another benefit of e-business. Integrating online interactions with call center sales and services representatives provides a unified customer experience. Automating direct, indirect and distributed sales models and transactions, benefits the sales team by streamlining quote preparation, opportunity tracking, order fulfillment, and customer information management and analysis. Furthermore, distribution partners sell more efficiently and collaborate more seamlessly via e-business, which in turn better serves the demands of customers.

2.1.2 Applications Areas

Respondents reported that improving their order management cycle is the primary reason for initial deployment of e-business applications, followed by promotions and product catalogues. As shown in Figure 2, online promotions and product catalogues are key application areas for 50% of the survey respondents, and for 46%, portals are a key application. The order management cycle includes ERP, CRM and SCM. Further popular applications are customer service support, lead management, and quoting and proposals.

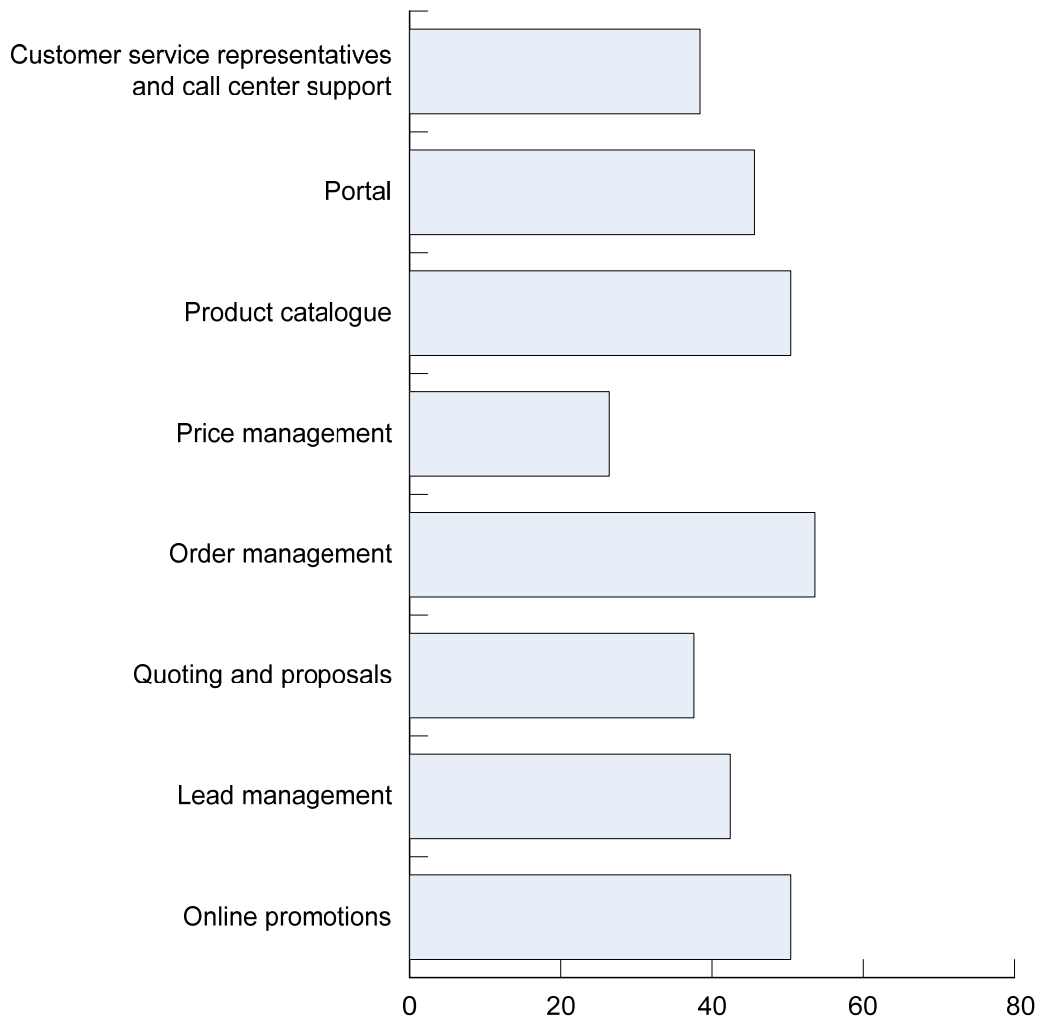


Figure 2 Key application areas of e-business solutions.

Customer services representatives and call center support, lead management, and quoting were also among the top e-business application areas. Guided selling and configuration and account management were implemented by some of the enterprises surveyed by Comergent Technologies. Many reported that their first implementations may include contract management, partner storefronts, and returns and replenishment capabilities.

2.2 Distribution Mechanisms and Middleware

As mentioned earlier, typically enterprises have multiple best-of-the-breed application systems from different vendors with different technologies and platforms. They are required to communicate within the enterprise and across enterprise borders. Enterprise application integration (EAI) facilitates this integration and encompasses approaches, methodologies, frameworks and architectures that are used to integrate a variety of enterprise applications to achieve A2A and B2B integration.

In [HoWo05], enterprise integration is defined as the task of making disparate applications work together to produce a unified set of functionalities. These applications can be custom developed in house or purchased from third-party vendors. They likely run on multiple computers that may represent multiple platforms and may be geographically dispersed. Some applications may also be run outside an enterprise by business partners or customers. Other applications might not have been designed with integration in mind and are difficult to change. These and similar issues make application integration complicated. Nevertheless, efficient integration of application systems is critical for most forms of e-business. In this section we explore multiple current integration approaches that help enterprises overcome most integration challenges.

According to [Vino02], middleware is required for integration because, as technology continues to evolve at an accelerating rate, nontrivial computing systems will remain diverse and heterogeneous. Hardware and applications purchased years ago must work together with newly acquired components. Together with factors such as mergers and acquisitions, reorganizations, leadership changes, and e-commerce, the heterogeneity in the overall systems rises sharply. According to Linthicum and Bussler, the need for integration technologies lies in the increasing IT complexity enterprises face today [Lint03, Buss03].

As with any complex technological effort, application integration involves a range of considerations and consequences that must be taken into account. Section 2.2.1 describes the decision criteria and section 2.2.2 introduces the most important integration technologies.

2.2.1 Decision Criteria

Realistically, even a simple enterprise has multiple applications that must work together to provide a unified experience for the enterprise's employees, partners and customers. The main decision criteria with respect to integration architectures listed in [HoWo05] are described in the following sections.

In section 2.2.1.1, we describe application coupling and in section 2.2.1.2 intrusiveness. Sections 2.2.1.3 and 2.2.1.4 outline technology selection and data formats, respectively. Data timeliness is discussed in section 2.2.1.5. In section 2.2.1.6, we elaborate on the question whether only data should be shared or also functionality. Finally, aspects of

remote communication and reliability are outlined in sections 2.2.1.7 and 2.2.1.8, respectively.

2.2.1.1 Application Coupling

Dependencies between integrated applications should be minimized so that each can evolve without affecting others. Tightly coupled applications make assumptions about how the applications they collaborate with work. As soon as the applications change and the assumptions break, the integration between the applications also breaks. Hence, the interfaces for integrating applications should be specific enough to implement useful functionality but general enough to allow the implementation to change as needed.

2.2.1.2 Intrusiveness

When integrating an application into an enterprise IT infrastructure, it is intended to minimize both changes to the application and the amount of integration code needed. However, changes and new code are often necessary to provide good integration functionality. The approaches with the least impact on the application may not always provide the best integration into the enterprise.

2.2.1.3 Technology Selection

Different integration techniques require varying amounts of specialized software and hardware. On the one hand, such tools are often expensive, lead to vendor lock-in and increase the learning curve for developers. On the other hand, creating an integration solution from scratch usually results in more effort than originally intended and can mean reinventing the wheel.

2.2.1.4 Data Format

Integrated applications must agree on the format of the data they exchange. Changing existing applications to use a unified data format may be difficult or even impossible. Alternatively, an intermediate translator can unify applications that insist on different data formats. A related issue is data format evolution and extensibility because formats often change over time and applications are usually affected.

2.2.1.5 Data Timeliness

Integration should minimize the time between the creation of data and its availability throughout the application systems. This can be accomplished by exchanging data frequently and in small chunks. However, chunking a large set of data into small pieces may introduce inefficiencies. Latency in data sharing must be factored into the integration design. Ideally, receiver applications should be informed as soon as shared data is ready for consumption. The longer sharing takes, the greater the opportunity for applications to become out of sync and the more complex integration can become.

2.2.1.6 Data or Functionality

Many integration solutions allow applications to share not only data but also functionality. Sharing functionality provides better abstraction between the applications. Even though invoking functionality in a remote application may seem the same as invoking local functionality, it works quite differently, with significant consequences for how well the integration works.

2.2.1.7 Remote Communication

Computer processing is typically synchronous. That means that a procedure waits while its subprocedure executes. However, calling a remote subprocedure is much slower than calling a local one, so that a procedure may not want to wait for the subprocedure to complete. Instead, it may want to invoke the subprocedure asynchronously. An asynchronous solution can be much more efficient but also more complex to design, develop and debug.

2.2.1.8 Reliability

Remote connections are not only slow, but they are also much less reliable than a local function call. When a procedure calls a subprocedure inside a single application, there is no question whether the subprocedure is available. This is not necessarily true when communicating remotely; the remote application may not be running or the network may be temporarily unavailable. Reliable asynchronous communication enables the source application to go on to other work, confident that the remote application will act some time later.

2.2.2 Integration Technologies

The criteria discussed in section 2.2.1 must be considered when choosing or designing integration approaches. No one integration approach can address all criteria equally well. Therefore, multiple approaches for integrating applications have evolved over time. Hohpe and Woolf sum up these approaches in four main integration styles: file transfer, shared database, remote procedure invocation (often referred to as remote procedure call, or RPC) and messaging [HoWo05].

As per Hohpe and Woolf, the trick is not to choose one style to use every time but to choose the best style for a particular integration opportunity. Integration approaches can best be viewed as a hybrid of multiple integration styles. To support such integration, many integration and middleware products use a combination of styles, all of which are effectively hidden in a product's implementation.

In an ideal world, it is possible to imagine an enterprise operating from a single, cohesive piece of software, designed from the beginning to work in a unified and coherent way. Of course, even the smallest enterprises do not work like that. Multiple pieces of software handle different aspects of a business for a host of reasons [HoWo05]:

- people buy software developed outside the organization;
- different systems are built at different times, leading to different technology choices;
- different systems are built by different people whose experience and preferences lead them to different approaches to building applications; and
- getting an application out and delivering value is more important than ensuring that integration is addressed, especially when that integration does not add any value to the application under development.

As a result, enterprises have to worry about sharing information between very divergent applications. These can be written in different languages and be based on different platforms and different assumptions about how a business operates.

Tying together such applications requires a thorough understanding of how to link applications on both business and technical levels. Integration is a lot easier if it bases on a common data transfer mechanism that can be used by a variety of languages and platforms

but feels natural to each. Middleware, a distributed software layer that sits above the network operating system and below the application layer, meets these needs as it abstracts from the heterogeneity of the underlying environment. It provides an integrated, distributed environment whose objective is to simplify the task of programming and managing distributed applications and also to provide value-added services such as naming and transactions to enable distributed application development [Mahm04].

The general integration patterns described in the following can be found in most enterprises. Application integration is the common goal of the four patterns described. Sections 2.2.2.1 describes file transfer. In section 2.2.2.2, we introduce an approach which bases on a shared database. Sections 2.2.2.3 and 2.2.2.4 outline the principles of RPC and messaging, respectively.

2.2.2.1 File Transfer

Files are a universal storage mechanism built into any operating system and accessible from any programming language. Hence, the simplest approach is to integrate applications using files. The process of two applications sharing data via file transfer is illustrated in Figure 3.

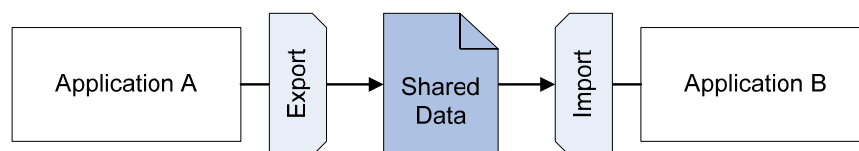


Figure 3 File transfer.

An important decision with files is what format to use. Very rarely the output of one application is exactly what is needed for another, so usually some processing of files along the way is required. This means not only that all applications that use files must be able to read them, but also that it must be possible to use processing tools on them. As a result, standard file formats have grown up over time and a wide variety of readers, writers and transformation tools have built up around each of these formats. Today, the most popular approach is to use XML (Extensible Markup Language).

Another issue with files is the timing of production and consumption, respectively. Because a certain amount of effort is required to produce and process files, it is usually not desirable to work with them frequently. Typically, a regular business cycle drives this decision.

The great advantage of files is that no knowledge of the internals of the applications is needed. As a result, the different applications are nicely decoupled from each other. Each application can be changed without affecting other applications, providing they still produce the same data in the files in the same format. The files effectively become the public interface of each application.

Part of what makes file transfer simple is that no extra tools or integration packages are needed. However, that also means that developers must do a lot of the work themselves. The applications must agree on file-naming conventions and the directories in which they appear. The writer of a file must implement a strategy to keep the file names unique. The applications must agree on which one will delete old files, and the application with that responsibility will have to know when a file is no longer needed. The applications need to implement a locking mechanism or follow a timing convention to ensure that one application does not try to read a file while another is still writing it. If not all applications have access to the same disk, some application must take responsibility for transferring the files from one disk to another.

One of the most obvious issues with file transfer is that updates tend to occur infrequently. As a result, systems can become out of sync easily. This can lead to inconsistencies that are difficult to resolve. The longer the period between file transfers, the more likely and more painful this problem typically becomes.

Of course, there is no reason that files cannot be produced more frequently. Indeed, messaging, which is described in detail in section 2.2.2.4, can be thought of as a file transfer in which a file is produced with every change in an application. In this case, the key problem is managing all the files produced, ensuring that they are all read and that none is lost. This goes beyond what approaches based on file systems can do, particularly because expensive resource costs are associated with processing files. Those costs can become prohibitive, if a lot of files must be produced quickly. As a result, once files become very fine-grained, it is easier to think of them as messaging.

2.2.2.2 Shared Database

File transfer enables applications to share data, but the approach usually lacks timeliness – yet timeliness of integration is often critical. If changes do not work their way rapidly through a group of applications, mistakes are likely to occur because of the staleness of data. For enterprises, it is imperative that every system permanently has the latest data.

This not only reduces errors, but also increases people's trust in the data itself. Rapid updates also allow better handling of inconsistencies. The more frequently applications are synchronized, the less likely inconsistencies are to arise and the less effort they are to deal with.

File transfer may not enforce a unique data format sufficiently. Many of the problems in integration come from incompatible ways of looking at data. Often, these represent subtle business issues that can have huge effects. These cases of semantic dissonance are much harder to deal with than inconsistent data formats. Semantic dissonance describes situations where data that appears to be the same may not necessarily mean the same thing. What is needed is a central, agreed-upon data storage that all applications share, so that each application has access to any of the shared data when required. The process of multiple applications using a shared database is illustrated in Figure 4.

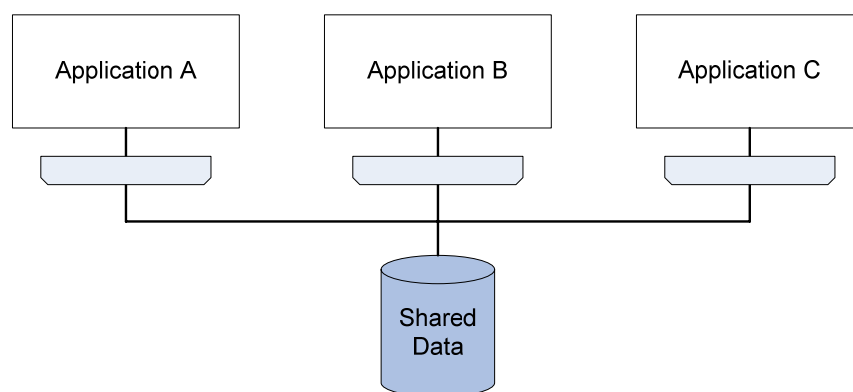


Figure 4 Shared database.

If in a set of integrated applications all rely on the same database, consistency is ensured at all times. Transaction management systems handle simultaneous updates of single pieces of data from different sources.

The use of shared databases was made much easier by the widespread adoption of relational databases based on SQL (Structured Query Language). Almost all application development platforms can work with SQL, often with quite sophisticated tools. Because any application can use SQL, there is no need to worry about multiple file formats.

Because every application is using the same database, problems with semantic dissonance are forced out. Rather than leaving these problems to fester until they are difficult to solve, using a shared database forces developers to deal with them before the software goes live and large amounts of data are collected.

One of the biggest difficulties is arriving at a suitable design for the actual database. Coming up with a unified schema that meets the needs of multiple applications is tricky, often resulting in a schema that is difficult to work with.

Furthermore, most packaged applications do not work with a schema other than their own. Even with some room for adaptation, it is likely to be much more limited than required. Adding to the problem, software vendors usually reserve the right to change the schema with every new release of an application.

This problem also extends to integrations after the development of a shared database. Even if all applications can be organized, there is still an integration problem should a merger or an acquisition occur.

Multiple applications using a shared database to frequently read and modify the same data can turn the database into a performance bottleneck and can cause deadlocks as each application blocks others temporarily. When applications are distributed across multiple locations, accessing a single, shared database across a wide-area network is often too slow to be practical.

2.2.2.3 Remote Procedure Invocation

File transfer and shared databases enable applications to share their data, which is an important part of application integration, but just sharing data is often not enough. Changes in data often require actions across different applications.

This problem mirrors a classic dilemma in application design. One of the most powerful structuring mechanisms in application design is encapsulation, in which modules hide their data through function call interfaces. In this way, they can intercept changes in data to carry out the various actions they must perform when data is changed.

Shared databases provide a large, unencapsulated data structure, which makes doing so hard. File transfer allows an application to react to changes as it processes a file, but the process is delayed. That shared databases have unencapsulated data also makes it more difficult to maintain a set of integrated applications. Changes in any application can trigger changes in the database, and database changes typically have a considerable ripple effect through all applications connected. As a result, enterprises that use a shared database are often reluctant to change the database, which implies the application development work to be much less responsive to changing business needs.

A mechanism is needed that allows one application to invoke a function in another application, passing the data that must be shared and invoking the function that tells the receiver application how to process the data. The RPC mechanism is illustrated in Figure 5.

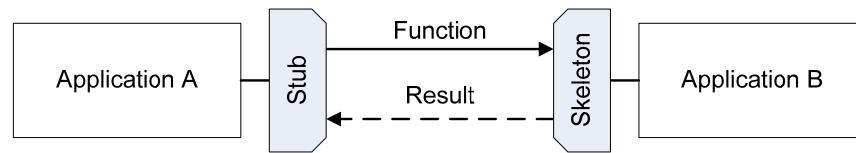


Figure 5 Remote procedure invocation.

RPC applies the principle of encapsulation to application integration. If an application needs some information that is owned by another application, it contacts that application directly. If one application must modify the data of another, it does so by making a call to the other application. This allows each application to maintain the integrity of its data. Furthermore, each application can alter the format of its internal data without affecting every other application.

Many middleware technologies, such as CORBA, DCOM, .NET or Java RMI, implement RPC. Often these environments add additional capabilities such as naming and transactions. Currently, the most popular approach is to use Web services which rely on widespread standards such as SOAP and XML [Mahm04]. They are described in more detail in section 3.1.1. A particularly valuable feature of Web services is that they work easily with the common HTTP¹ (Hypertext Transfer Protocol), which is easy to get through firewalls.

There are methods for wrapping data that make it easier to deal with semantic dissonance. Applications can provide multiple interfaces to the same data, allowing some clients to see one style and others a different style. Even updates can use multiple interfaces. This provides greater ability to support multiple points of view than can be achieved by relational views. However, each application must negotiate its interface with its neighbors.

Although encapsulation helps to reduce the coupling of the applications by eliminating a large shared data structure, the applications are still fairly tightly coupled. The remote calls that each system supports tend to tie the different systems into a growing knot and it can become difficult to change systems independently. These problems often arise because

¹ HTTP is a communication protocol used to transfer or convey information on the Web. Its development is coordinated by the W3C and the Internet Engineering Task Force.

issues that are not significant within a single application become so when integrating a larger number of applications.

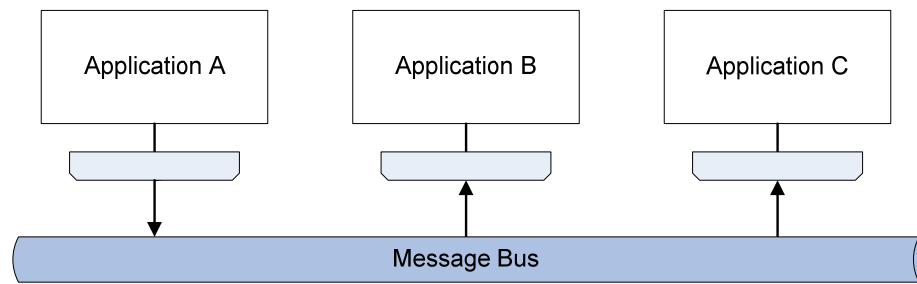
2.2.2.4 Messaging

File transfer and shared databases enable applications to share their data but not their functionality. RPC enables applications to share functionality, but it also tightly couples them. Often the challenge of integration is to make collaboration between separate systems as timely as possible without coupling systems together in a way that they become unreliable either in terms of application execution or application development.

File transfer allows systems to keep applications well decoupled but at the cost of timeliness. Shared databases keep data together in a responsive way but at the cost of coupling everything to the database. File transfer as well as shared databases fail to handle collaborative behavior.

Faced with these problems, RPC seems to be an appealing choice. However, extending a single application model to application integration dredges up many new weaknesses such as the essential problems of distributed development. Although, for instance, RPCs look like local calls, they do not behave the same way. Remote calls are slower and much more likely to fail. With multiple applications communicating across an enterprise, one application's failure should not bring down all other applications. Furthermore, it is not desired to design a system assuming that calls are fast, and each application should not have to know the details about other applications, even if it is only details about their interfaces.

What is needed is something like file transfer in which many little data packets can be produced quickly and transferred easily, and in which the receiving application is automatically notified when a new packet is available for consumption. The transfer also needs a retry mechanism to make sure it succeeds. The details of any disk structure or database for storing the data must be hidden from the applications so that, unlike with shared databases, the storage schema and details can be easily changed to reflect changing needs. One application should be able to send a packet of data to another application to invoke behavior in the other application such as an RPC, but without being prone to failure. Furthermore, the data transfer should be asynchronous so that the sender does not need to wait for the receiver, especially when a retry is necessary. The messaging process is illustrated in Figure 6.

**Figure 6** Messaging.

According to Hohpe and Woolf, a message bus is a combination of a common data model, a common command set and a messaging infrastructure to allow different systems to communicate through a shared set of interfaces [HoWo05].

Asynchronous messaging is a pragmatic reaction to the problems of distributed systems. Sending a message does not require both systems to be up and ready at the same time. Furthermore, thinking about the communication in an asynchronous manner forces developers to recognize that working with a remote application is slower, which encourages design of components with high cohesion and low adhesion.

Messaging systems also allow much of the decoupling of file transfer. Messages can be transformed in transit without either the sender or the receiver knowing about the transformation. Decoupling allows integrators to choose among broadcasting messages to multiple receivers, routing a message to one of many receivers, or other topologies. This approach separates integration decisions from applications development.

Transformation means that separate applications can have quite different conceptual models. Of course, this means that semantic dissonance will occur. However, the messaging viewpoint is that the measures used by shared databases to avoid semantic dissonance are too complicated to work in practice. Also, semantic dissonance is going to occur with third-party applications and with applications added as part of a merger or an acquisition, so the messaging approach is to address the issue rather than design applications to avoid it.

Sending small messages frequently also allows applications to collaborate behaviorally as well as share data. Information can be requested and the request fulfilled rapidly. While such collaboration is not going to be as fast as an RPC, the caller need not stop while the message is being processed and the response returned.

The high frequency of messages in messaging reduces many of the inconsistency problems that bedevil file transfer, but it does not remove them entirely. There are still going to be some lag problems with systems not being updated simultaneously. Asynchronous design still has a learning curve, and testing and debugging are also harder in this environment.

The ability to transform messages has the benefit of allowing applications to be much more decoupled from each other than for RPC. This independence, however, means that integrators are often left with writing a lot of messy glue code to fit everything together. Technologies that, according to [Vino02], some see as competitive such as RPC and messaging, often must be applied together to solve real-world problems. After all, there is no one-size-fits-all solution to the problems enterprise integration addresses. As per Mahmoud, message-oriented middleware (MOM) is a cornerstone of distributed enterprise systems. He defines MOM as any middleware infrastructure that provides messaging capabilities [Mahm04]. Web services equally support RPC- and message-oriented systems.

2.3 Types of Integration

There are many different ways to integrate enterprises, with their various autonomous divisions and with their hosted or nonhosted back-end application systems. To understand the requirements of an enhanced integration architecture, it is useful to explore and examine typical integration scenarios in more depth.

More and more trading partners as well as back-end application systems that follow different formats and process standards must be integrated. The phases of the evolution of integration technologies can be observed throughout many industries. These phases overlap, indicating that the increased requirements have been recognized and built into new products. As per Bussler, enterprises must cope with three forms of integration: A2A integration between back-end application systems, B2B integration to connect trading partners and hosted application integration [Buss03].

In section 2.3.1, we introduce the idea underlying value added networks (VAN). Section 2.3.2 describes A2A integration in general and the role of integration brokers in particular. B2B integration over the Internet is outlined in section 2.3.3. In section 2.3.4 and 2.3.5, we focus on integration typologies based on transformation hubs and hosted application integration, respectively. Finally, in section 2.3.6, we sketch the notion of marketplace-based integration.

2.3.1 Value-Added Networks

Already more than 30 year ago, enterprises typically had a back-end application system that stored critical business data that had to be exchanged with their business partners. Ever since then, nearly every industry has its own dominant standard. Many enterprises used electronic data interchange (EDI) for business message exchange as a B2B protocol standard. In the financial industry, SWIFT was the predominant standard. The enterprises were only connected to value-added networks. The general architecture topology for B2B integration using EDI over VAN is illustrated in Figure 7.

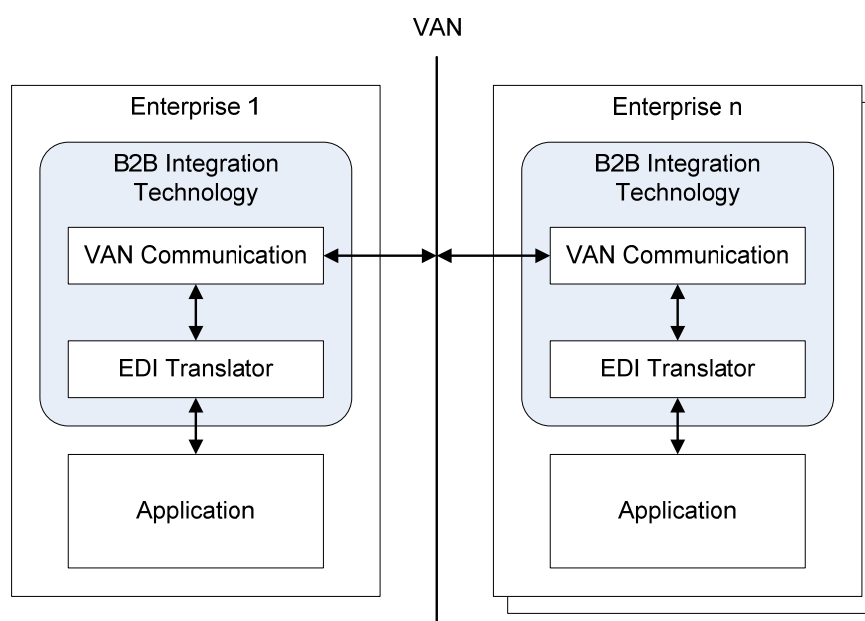


Figure 7 B2B integration using EDI over VAN.

The back-end application systems are usually connected to EDI translators which transform business data from the application systems into EDI syntax and send it over VAN to the trading partners. The topology allows exchanging business data with all business partners following the EDI standard message layout.

2.3.2 Hub-and-Spoke versus Bus

However, following the best-of-the-breed approach, most enterprises have more than one back-end application. Enterprises install the best back-end application systems they can get, often supplied by different vendors. Because the back-end application systems expose different interfaces to access their internal data for message exchange, this results in a heterogeneous environment. The back-end applications do not only have to exchange

business data with trading partners but also between themselves. Hence, A2A integration is implemented as well.

As shown in Figure 8, the information broker connects to the EDI translator. In less advanced variations of this topology, the translator connects to each back-end application system individually. In more advanced architecture topologies, the integration broker is aware of the B2B connectivity and all messages are routed through it.

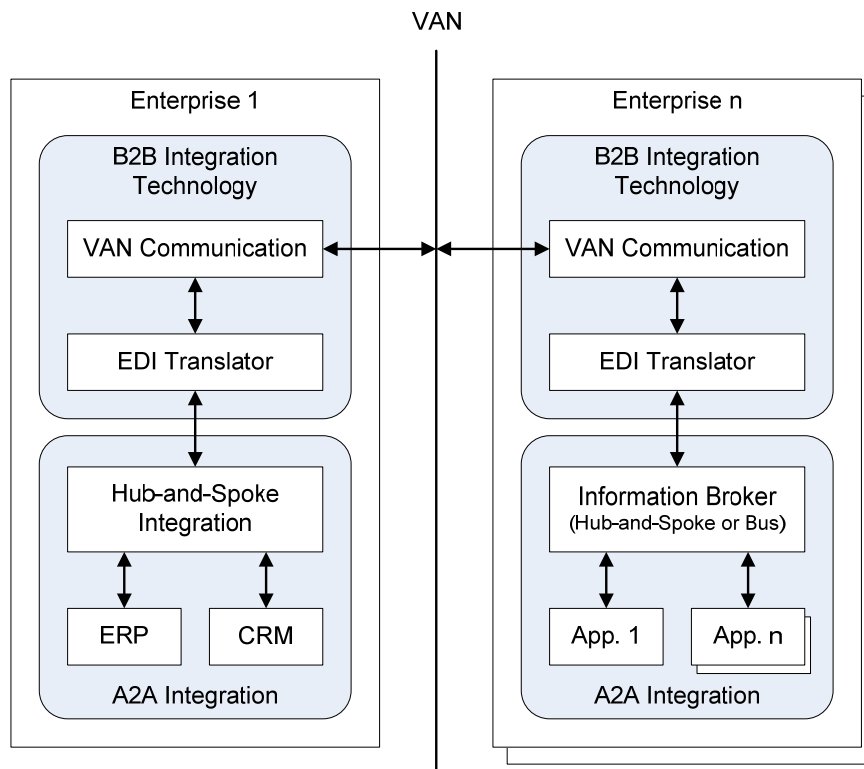


Figure 8 A2A integration.

The information broker can either realize a hub-and-spoke or a bus topology. Each topology has its own advantages and disadvantages. In the hub-and-spoke model, the A2A system is at the center and interacts with the applications via the spokes. In the bus model, the A2A system is the bus [Masa05]. If integration is applied without following one of these models, point-to-point connections are in danger of growing across the enterprise.

2.3.3 B2B Integration over the Internet

More recent developments in B2B protocol standards such as OAGIS¹ (Open Applications Group Integration Specification), RosettaNet¹ or ebXML² posed a new challenge for

¹ For more information on OAGIS, see <http://www.openapplications.org/>.

enterprises. The diversity of B2B protocol standards forces enterprises to support multiple standards simultaneously. Two enterprises can only cooperate, if both agree on a common B2B protocol standard. Software components similar to EDI translators must be maintained and installed for all other standards used. Additional translators are needed to transform formats which are specific to back-end application systems into the format required by the B2B protocol. The use of additional networks such as the Internet requires communication software that supports protocols such as FTP³ (File Transfer Protocol), HTTP or SMTP⁴ (Simple Mail Transfer Protocol). The resulting architecture topology is shown in Figure 9.

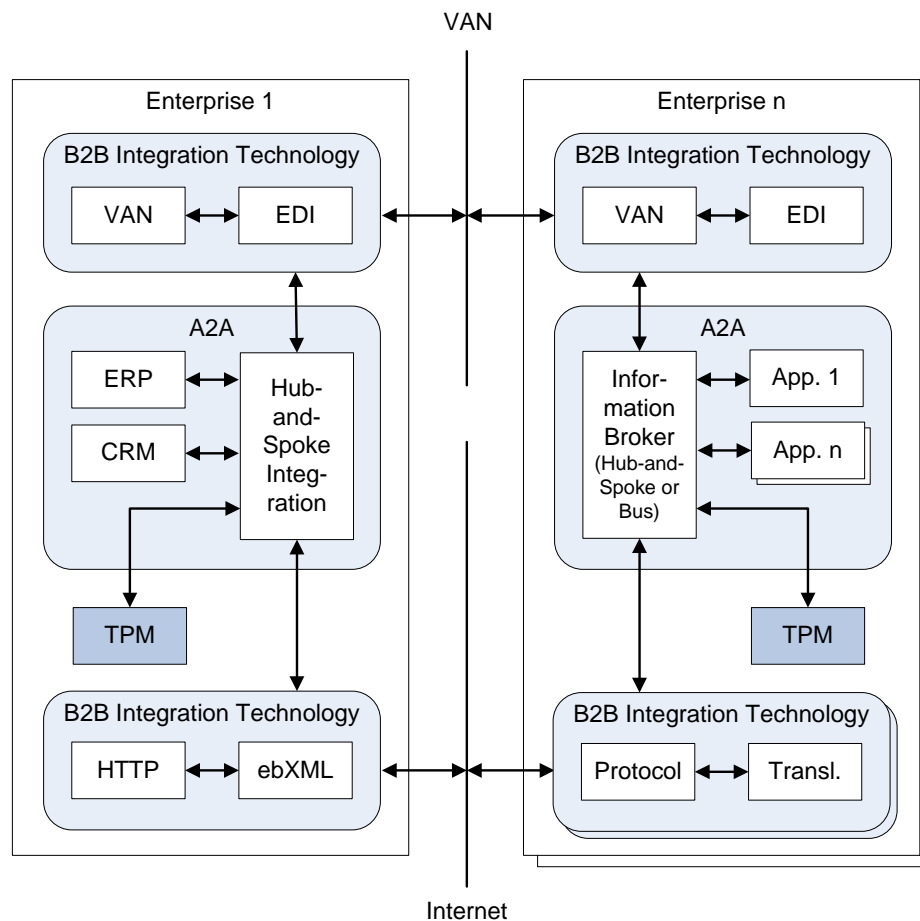


Figure 9 B2B integration over the Internet.

Because all B2B protocols require different data about trading partners, a separate trading partner management (TPM) component is added to manage trading partner information

¹ For more information on RosettaNet, see <http://www.rosettanet.org/>.

² For more information on ebXML, see <http://www.ebxml.org/>.

³ FTP is a commonly used protocol for exchanging files over any network that supports the TCP/IP protocol.

⁴ SMTP is the de facto standard for e-mail transmissions across the Internet.

centrally and consistently. The TPM is accessed by the different translators through an integration broker. This topology is complex and its management requires careful attention. Business rules, for instance, have to be implemented in all B2B protocol translators. As the different components do not share a common data infrastructure, the consistent implementation of rules has to be ensured manually.

2.3.4 Transformation Hubs

The more trading partners an enterprise has, the more B2B protocols it must support. For each business event, usually one transformation per B2B protocol must be implemented. If an event can be both sent and received, two transformations are necessary. For an enterprise, a high number of transformations implies a significant maintenance effort. This issue is addressed by transformation hubs.

Transformation hubs implement the transformations between B2B protocols. Enterprises usually communicate with the hub through a B2B protocol over a specific network. This means that an enterprise has only to transform its internal business data into a B2B protocol. After the transformation, the hub forwards messages to the final destination trading partner.

In addition to transformation, hubs can also provide other services such as message logging and time-stamping as well as basic analysis functionality.

2.3.5 Hosted Application Integration

In the hosted application model, the data of an enterprise and its applications are hosted by an ASP (Application Service Provider)

For that reason, enterprises which want to integrate their hosted data with their back-end application systems must integrate them over a network. To communicate, an ASP protocol is required between the subscriber and an ASP. Figure 10 shows the architecture topology.

Complexity increases through the addition of the ASP protocol component, which implements connectivity and data transfer to and from ASPs. In reality, as many ASP connectors are needed as an enterprise has subscribed-to ASPs because in general different ASPs implement different ASP protocols.

Independent of the integration of the hosted applications of an enterprise with its locally installed back-end application systems, users typically access the hosted applications through browsers over the Internet.

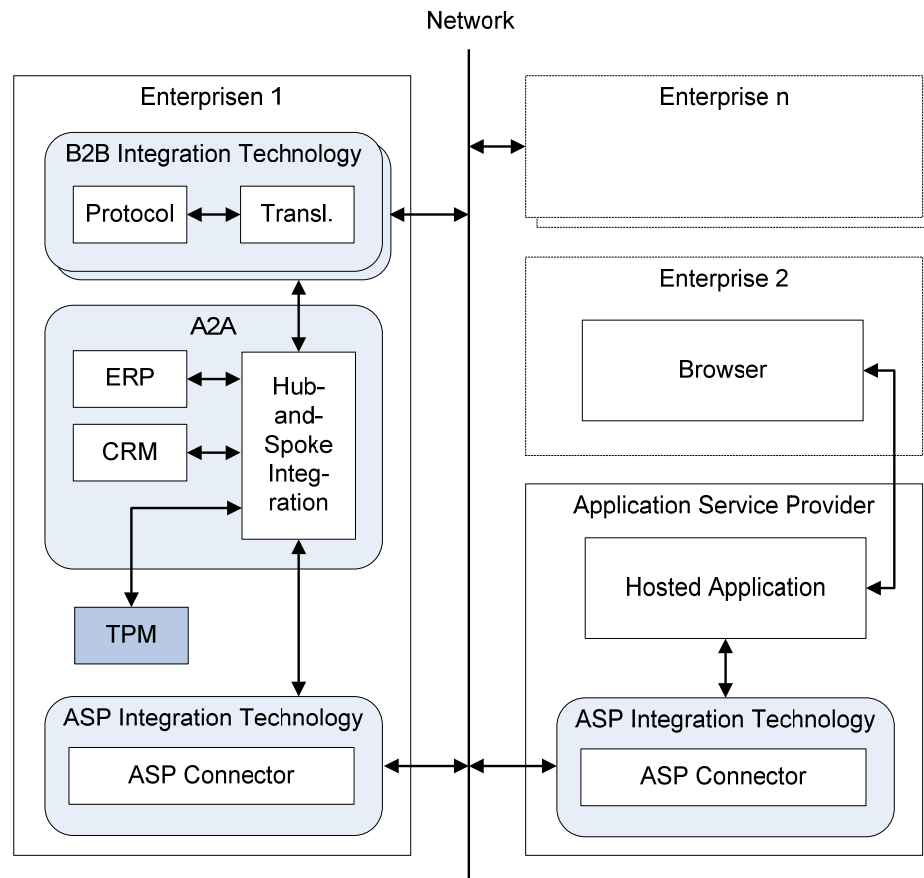


Figure 10 Hosted application integration.

Within an ASP multiple application systems of an enterprise can be integrated with each other using B2B integration technologies. Often enterprises face the situation that the back-end application systems that they want to be hosted cannot all be hosted at the same ASP. In this case ASP aggregators manage the coordination between several ASPs. This architecture allows enterprises to access their hosted applications in a homogeneous way as if all were installed at a single ASP.

Furthermore, it is also possible that the integration functionality is hosted. In this case, each of the back-end application systems is connected to the ASP's B2B integration technology server through dedicated B2B adapters so that the ASP can connect and integrate the back-end applications systems. The application systems are not hosted themselves, but are locally installed with the enterprise.

The delivery model where applications are hosted and operated for the use by customers over a network, is also known as software as a service (SaaS). SaaS implies that customers do not pay for owning the software itself but for using it. The application systems are either hosted and operated independently by the vendor or through third parties.

2.3.6 Marketplace Integration

In its simplest form, a marketplace lists suppliers offering products and allows buyers to access the supplier listings. The marketplace has achieved its goal once a buyer finds a product it wishes to buy. All subsequent business data exchange takes place outside and independent of the marketplace.

According to Bussler, more sophisticated marketplace forms provide automatic matching between buyer and seller. Some also issue the purchase order and manage the purchase order acknowledgment as a response. This means that in some cases the business data exchange is also managed by the marketplace. In extreme cases the marketplace provides all business data exchanges so that the trading partners do not have to connect with each other directly [Buss03].

2.4 Selected Integration Products

Enterprise integration is not yet founded on a commonly agreed-upon set of integration concepts. An indication is that product consolidation has not yet taken place. Journals report continually on an astonishing array of established and new integration products from different vendors. In addition, mergers and acquisitions of integration product vendors constantly change the product landscape. As per Bussler, the observation that the number of products first increases and then decreases to at most a handful, which own the lion's share of the market, is characteristic for a maturing domain [Buss03]. The enterprise integration market is not yet at this level of maturity. A consolidation of integration products, however, is expected in time.

It is impossible to cover the wide range of products within the scope of this work. It is also quite difficult to obtain enough technical information about off-the-shelf integration products to provide a meaningful description because only a few vendors make technical information publicly available. Hence, we decided to characterize only a couple of highly visible products in the marketplace. The descriptions mainly focus on core concepts

because any elaborate discussion would immediately become outdated with the next version of the product.

It is interesting but not surprising that all major vendors such as IBM, Microsoft, BEA and SAP focus on SOAs based on Web services. IBM's platform WebSphere is sketched in section 2.4.1. In section 2.4.2, we present Microsoft's BizTalk Server. Finally, BEA's WebLogic Integration and SAP's NetWeaver are described briefly in sections 2.4.3 and 2.4.4, respectively.

2.4.1 WebSphere

WebSphere¹ is IBM's integration software platform. WebSphere is a modular platform based on industry supported open standards and includes middleware infrastructure such as servers, services and tools needed to write, run and monitor Web applications as well as cross-platform and cross-product solutions.

WebSphere Application Server forms the basis of the infrastructure, which supports SOA as well as non-SOA environments. WebSphere Process Server, which is based on WebSphere Application Server, and WebSphere Enterprise Service Bus provide the foundation for modular applications which are architected in a service-oriented way. Collectively, they support the use of business rules to drive applications that support business processes. Other WebSphere products provide a wide variety of additional services.

2.4.2 BizTalk Server

Like its predecessors, Microsoft's BizTalk Server 2006² allows the connection of diverse applications and graphically creating and modifying process logic. The product also lets information workers monitor running processes, interact with trading partners and perform other business-oriented tasks. From its initial roots in EAI and B2B integration, BizTalk Server has grown into a foundation product supporting a wide range of business processes.

The goal of BizTalk Server 2006 is to help enterprises meet the challenges of creating automated business processes that rely on diverse systems. The product's foundation is the BizTalk Server 2006 engine, which provides core messaging and orchestration capabilities. Developers can also use the Business Rules Engine to address complex business scenarios,

¹ For more information on IBM WebSphere, see <http://www-306.ibm.com/software/websphere/>.

² For more information on Microsoft BizTalk Server, see <http://www.microsoft.com/biztalk/>.

the Health and Activity Tracking tool to debug and examine BizTalk applications, and Enterprise Single Sign-On to create more secure environments. Information workers can use the product's Business Activity Monitoring support to get information about running processes and Business Activity Services to work with trading partners.

2.4.3 WebLogic Integration

BEA WebLogic Integration 9.2¹ is a unified solution for integrating business systems within enterprises. WebLogic Integration provides a development and run-time framework that unifies many components of business integration into a single environment. WebLogic Integration is part of the WebLogic Platform which provides functionality enterprises can use to develop new applications, create both businesses and Web services, integrate them with existing systems, streamline business processes, and extend e-business infrastructure through portal gateways.

The product provides a single environment for building an integrated enterprise application system whether it is for business process integration, custom application development using robust Web services and controls, or developing a portal to provide employees, partners and customers with an integrated view of applications and data.

WebLogic Integration also acts as a strategic building block in enterprise integration solutions for SOAs.

2.4.4 NetWeaver

SAP NetWeaver² is a Web-based, open integration and application platform that serves as the foundation for an enterprise SOA and allows the integration and alignment of people, information and business processes. The product is marketed principally as a service-oriented application and integration platform and represents the technical foundation of SAP xApps and mySAP Business Suite solutions. NetWeaver provides the development and runtime environment for SAP applications and can be used for custom development and integration with other applications and systems. NetWeaver is built using open as well as industrial de-facto standards and can be extended and interoperate with technologies such as .NET and Java EE.

¹ For more information on BEA WebLogic Integration, see <http://www.bea.com/integration/>.

² For more information on SAP NetWeaver, see <http://www.sap.com/platform/netweaver/>.

SAP considers the product release a strategic move for driving enterprises to run their business on a single, integrated platform that includes both applications and technology infrastructure. SAP is fostering relationships with system integrators and independent software vendors. This strong momentum illustrates the power SAP holds over the market for packaged software solutions. NetWeaver is part of SAP's plan to move to a more open SOA and to deliver the technical foundation of its applications on a single, integrated platform and common release cycle.

Chapter 3

Semantic Web Services

The standards used to describe Web services do not provide means to describe semantics. Hence, Web services are not suited for automatic processing. To better support discovery of services and probably automate their composition into complex processes, a formal, standardized and semantic description of services is needed. The idea is to use Semantic Web technologies to describe Web services. The ultimate objective is to combine services on the fly to achieve given goals. Based on the goal descriptions as well as the descriptions of the available services, a complex service yielding the desired result should be composed automatically out of atomic building blocks [AlSm05].

SWSs have received enormous attention in past years. Several efforts are involved in SWS technology research and development, most gathered around OWL-S and WSMO, which are the most significant frameworks.

Figure 11 shows the constantly growing number of publications devoted to Semantic Web services since 2001. McIlraith et al. published one of the first scientific publications on SWSs, in 2001 [McSZ01]. In 2003, 28 works on SWSs were published, up from nine in 2002. In 2006, 147 works were published and it is highly probable that the number for 2007 will be significantly higher. The numbers were gathered by querying the digital libraries of the major publishers in the field. Digital libraries operated by Elsevier, Springer, ACM (Association for Computing Machinery), and the IEEE (Institute of Electrical and Electronics Engineers) Computer Society were investigated.

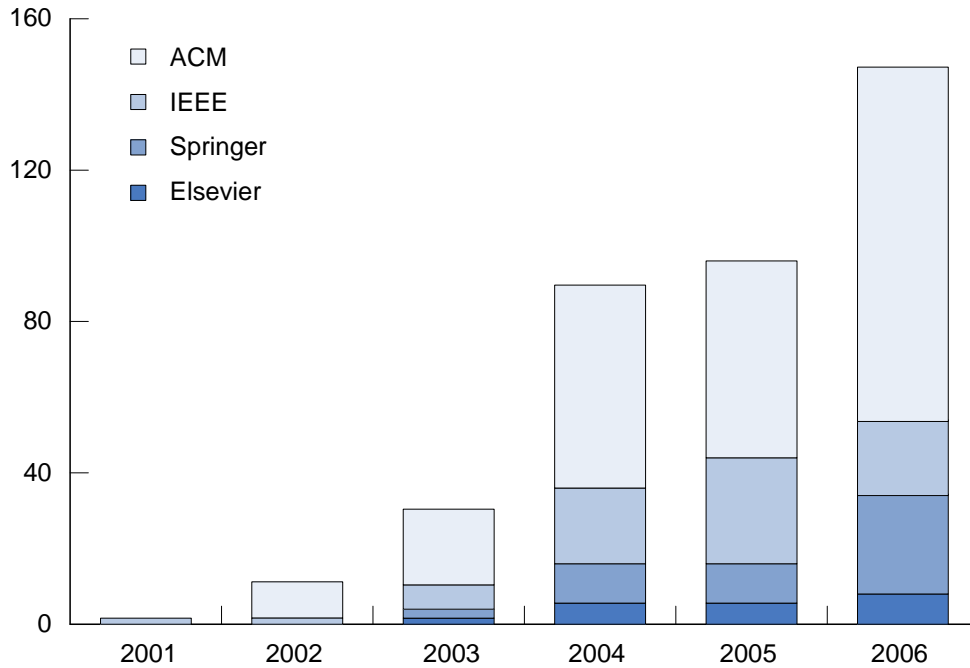


Figure 11 Publications on SWSs since 2001.

Elsevier and Springer, the world's largest publishers in the science, technology and medicine sectors, operate two of the most comprehensive online collections of published scientific research in the world. While Elsevier runs ScienceDirect¹, Springer puts its efforts into SpringerLink². ACM's digital library³ is the world's largest collection of information on computing machinery. The association's primary competitor is the IEEE Computer Society, which also runs a digital library⁴. The IEEE Computer Society focuses on hardware and standardization issues, while the ACM concentrates more on theoretical computer science and end-user applications.

The technologies underlying SWSs are discussed in section 3.1. In section 3.2, we focus on its core components and in section 3.3, on the differences between the most popular frameworks.

¹ For more information on ScienceDirect, see <http://www.sciencedirect.com/>.

² For more information on SpringerLink, see <http://www.springerlink.de/>.

³ For more information on ACM's digital library, see <http://portal.acm.org/dl.cfm>.

⁴ For more information on the IEEE Computer Society's digital library, see <http://www.computer.org/portal/site/csdl/>.

3.1 Constituent Technologies

SWSs, also referred to as intelligent Web services, are supposed to bring the Web to its full potential. Current Web technologies will not be replaced by Semantic Web technologies; the idea of a Semantic Web must rather be seen as some kind of upgrade.

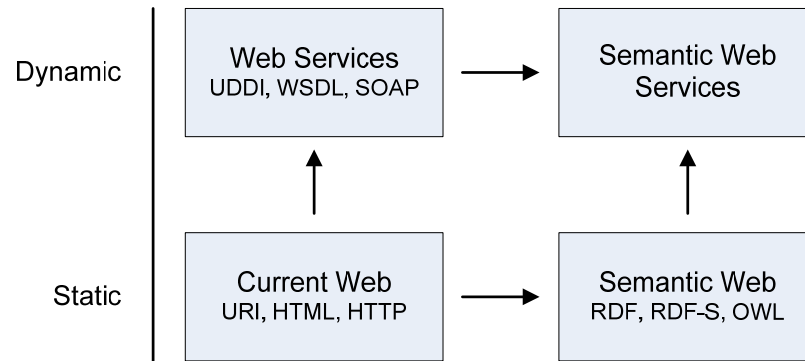


Figure 12 SWS technologies.

Figure 12 shows that SWSs result from the combination of Web service and Semantic Web technologies. Both technologies are composed of sets of standards that are explained in the corresponding subsections. In section 3.1.1, we introduce Web services and briefly sketch the fundamentals of SOAs. In section 3.1.2, the notion of the Semantic Web is summarized.

3.1.1 Web Services

In section 3.1.1.1, we describe the primary objectives of Web services. Section 3.1.1.2 provides a rough overview of the languages used and the progress of the standardization process. Finally, in section 3.1.1.3, the benefits and challenges are outlined.

3.1.1.1 Primary Objectives

The basic idea underlying Web services is to facilitate the use and integration of applications by making them independent from the technologies with which they have been implemented. A Web service should be accessible by clients regardless of the programming languages in which the service or the client have been implemented or the operating systems on which they are running. Moreover, Web services should be highly reusable and easily combinable. Services are seen as building blocks for processes that can be rearranged by business experts without involving developers. Finally, replacing a service with another service offering the same functionality should be straightforward.

Web services are inherently process-oriented. A service, or more precisely a service operation, is an action performed by a program. Processes can then be constructed by combining individual service operations. The W3C¹ (World Wide Web Consortium) has established six working groups and one interest group concerned with the architecture of Web services within the scope of its Web Service Activity².

3.1.1.2 Languages and Standards

Four basic standards formed the initial specification for Web services. They are explained elaborately in [Newc02]. Three of them are generally accepted today: XML-S (XML Schema) for describing data types, SOAP as a message format containing service requests and responses, and WSDL (Web Services Description Language) to describe service interfaces by specifying the protocol bindings and message formats required for interactions. The relevance of the fourth basic standard, UDDI (Universal Description, Discovery and Integration), is less unanimous. UDDI defines four core types of information that provide the data required to discover and use Web services. These information types include business information, service information, binding information and information about specifications for services [Oasi00, Wieh04]. Because most applications using Web services focus on integration within enterprises, there was not yet a real need to discover available services.

In addition to these basic standards, a host of standards that address specific aspects of Web service usage have been developed. Together they form a kind of a Web service standards stack. These standards involve issues such as security, reliable messaging, choreography of services and transactions. The basic Web services stack of standards is depicted in Figure 13.

WS-BPEL (Web Services Business Process Execution Language, formerly BPEL4WS) defines a notation for specifying business process behavior. Based on atomic Web service operations specified in WSDL, it is possible to define complex workflows by specifying sequences of operations, preconditions and post-conditions, data to be exchanged between operations, conditional branches and error cases.

¹ The W3C develops interoperable technologies to lead the Web to its full potential. It is a forum for information, commerce, communication and collective understanding. For more information on the W3C, see <http://www.w3.org/>.

² For more information on the W3C Web Service Activity, see <http://www.w3.org/2002/ws/>.

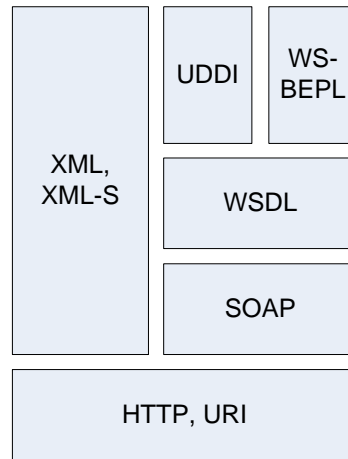


Figure 13 Web service standards stack.

The pace of the standardization progress is impressive. Manes describes in [Mane03] popular Web service technologies and their standardization. Standards are not only available for most aspects of Web services, major software vendors are also rapidly supporting them and integrating them into their product suites. Standard compliance is essential for Web services because it is the key to achieving independence from technologies, vendors and programming languages, which is the primary objective of Web services.

In spite of the advanced state of standardization, interoperability is not always guaranteed. Because existing standards are very extensive, software products often support only subsets of the complete standards. The WS-I Group¹ (Web Services Interoperability) has been founded to overcome these challenges. Its purpose is to promote Web service interoperability across platforms, operating systems and programming languages. Many software vendors have committed to ensuring that their products are WS-I profile compliant, and some products already offer functionality testing for WS-I profile compliance.

3.1.1.3 Benefits and Challenges

The basic functionality provided by Web services addresses issues relevant to all distributed software systems. Web services allow for modularization or decoupling of components, which is essential to keep large systems manageable. In principle, Web services are thus similar to other existing technologies for distributed environments.

¹ For more information on WS-I, see <http://www.ws-i.org/>.

As a consequence, some benefits provided by Web services can also be achieved without actually using Web service technologies in the narrow sense. The term SOA is often used to describe the design principles that allow for loosely-coupled systems which provide a high degree of flexibility and reusability [MaBe06]. While SOAs can be realized with a wide range of technologies, Web services offer the best technological foundation for SOAs at present.

Although WSDL and SOAP can be bound to various transport mechanisms they are often associated with synchronous communication via HTTP. However, in most situations asynchronous communication via messages and queues is more advantageous as message senders and receivers do not need to know anything about each other [Zura05].

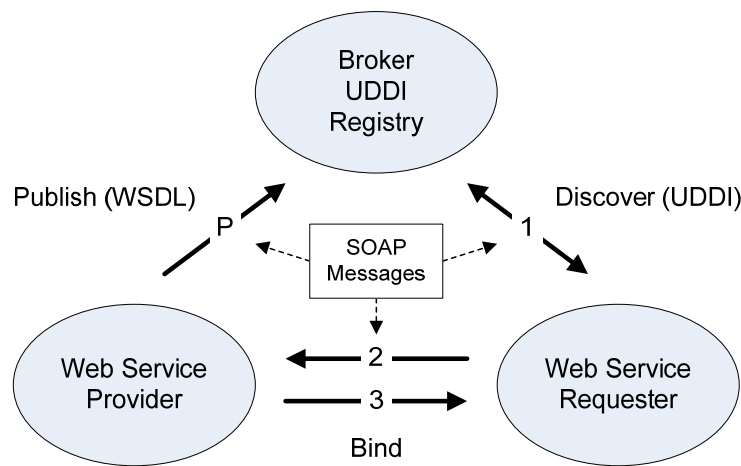


Figure 14 Web service usage process.

The theoretical Web service usage process, shown in Figure 14, involves three parties: Web service provider, broker and requester. Any Web service registered with the UDDI registry can be invoked by arbitrary applications via the network. In general, three steps are necessary: first, the Web service requester queries the UDDI registry for the required service. If successful, a SOAP document is sent to the Web service provider to invoke the Web service. After the execution, the Web service delivers the result as a SOAP document back to the Web service requester.

According to Vinoski, Web services currently present the most promising way to facilitate application integration based on Internet technologies [Vino02]. Web services simply use the ubiquitous Internet infrastructure to apply proven approaches from mature middleware as described in section 2.2.2. As per Vinoski, Web services are based on the convergence of four technology streams:

- Web services operate over the ubiquitous infrastructure of the Internet and communicate with other applications via Internet protocols such as HTTP.
- Web services incorporate fundamental aspects of proven approaches to middleware such as the creation of SOAs and the equal support of RPC-oriented and message-oriented systems.
- Web service contracts are defined in XML and Web services communicate via XML-based messages. This allows developers to create and manipulate structured information with domain-independent tools.
- Web services can incorporate business standards to allow cooperating trading parties to fully understand the Web service semantics and subsequently enable correct interactions.

Web services are used to integrate and expand the capabilities of existing middleware solutions rather than replace them [Vino02].

A major benefit of Web service technologies are the wide-ranging vendor support and their inhering ambition to provide independence from technologies, vendors and programming languages. Despite minor problems with interoperability, Web services are successfully used in real-world applications and many of the promises associated with Web services have been kept.

The completion of the Web service stack of standards and the use of the technologies in large-scale inter-enterprise applications remain two big challenges. So far, Web services are still predominantly used in integration projects within an enterprise or as APIs (Application Programming Interfaces) that offer existing functionality for access over the Internet.

3.1.2 Semantic Web

In section 3.1.2.1, we describe the visions of the Semantic Web. Section 3.1.2.2 provides a rough overview of the languages used and the progress of the standardization process. Finally, in section 3.1.2.3, the benefits and challenges are outlined.

3.1.2.1 Visions

Berners-Lee et al. first described with audience appeal the evolution of a Web that consists largely of documents for humans to read to one that includes data and information for computers to manipulate. In [BeHL01] they state that the Semantic Web is a web of actionable information derived from data through a semantic theory for interpreting symbols. The semantic theory provides an account of meaning in which the logical connection of terms establishes interoperability between systems.

As per Antoniou and van Harmelen, the development of the Semantic Web has a lot of industry momentum and governments are investing heavily. The Semantic Web is among the key action lines of Europe's Seventh Research Framework Programme¹ (FP7) and the government of the United States of America established the DAML² (DARPA Agent Markup Language) program in 2000 [AnHa04].

The basic idea underlying the vision of the Semantic Web is to make websites machine-processable by structuring and enhancing the information contained in them. This allows for the creation of various intelligent applications roaming the Internet and using the Web's information to offer all sorts of services [FHLW05].

The rather unstructured HTML (Hypertext Markup Language) pages of the Web are sufficient for human readers, who use their background knowledge to identify relevant pieces of information when reading pages. However, it is not straightforward to extract such information automatically. Programs that do this at present are usually hand-coded scripts, which are particularly tailored towards the websites from which they extract information. Reusability is low and maintenance is costly as the scripts must be updated as soon as the structure or the layout of the underlying website changes. It would be much easier if information on the Web contained information about itself. The Semantic Web aims to standardize information in a way that metadata can be added to the data which is currently put on websites.

To resolve semantic differences and to intelligently process information semantics are two of the main motivations driving the Semantic Web. It is an extension of the current Web in

¹ FP7 is the European Union's most powerful instrument for funding research over the period 2007 to 2013. For more information on FP7, see <http://cordis.europa.eu/fp7/>.

² The goal of the DAML effort was to develop languages and tools to facilitate the concept of the Semantic Web. The effort ran over the period 2000 to 2006. For more information on DAML, see <http://www.daml.org/>.

which information is given well-defined meaning, better enabling computers and people to work in cooperation [BeHL01].

The Semantic Web is primarily static as it aims at enhancing information representation in order to make automatic processing of this information feasible. Combining it with the more dynamic concept of Web services is thus facilitating its usage in process-oriented application contexts.

3.1.2.2 Languages and Standards

With RDF (Resource Description Framework) and OWL (Web Ontology Language), the Semantic Web Activity¹ at the W3C has released two recommendations so far. The Semantic Web stack of standards is illustrated in Figure 15.

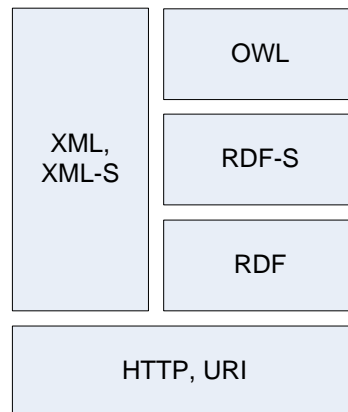


Figure 15 Semantic Web standards stack.

Although RDF is often regarded as the basic Semantic Web standard, some argue that it does not add anything to plain XML that would affect its expressivity. Basically, RDF allows the description of resources via simple statements, each consisting of subject, predicate and object identified by URIs (Universal Resource Identifiers). The purpose of a URI is to uniquely identify a concept by linking to its definition. Because RDF is based on XML, the statement in XML notation looks a bit more complex. RDF offers several different notational variants but as these variants are purely notational, they can be mapped into each other.

Just as XML as a standard only stipulates the general syntax of XML documents and data without restricting the set of permissible tag or attribute names, RDF itself does not fix the

¹ For more information on the W3C Semantic Web Activity, see <http://www.w3.org/2001/sw/>.

potential range of properties and values. And just as XML-S can be used to further specify the structure of XML documents in an application area, RDF-S (RDF Schema) provides information about RDF statements for a particular domain. RDF-S can be used to define a set of properties and values permitted in RDF statements.

OWL goes one step further and allows the specification of logical relationships between properties, classes and values. It is based on description logics¹ (DL) and provides the basis for reasoning on given information. Thus, given a list of OWL statements, additional statements can be automatically derived. OWL uses class hierarchies as a central means for structuring. It thus has mechanisms of inheritance similar to object-oriented languages or XML-S. In contrast to these formalisms it also supports defined classes. Defined classes are classes with necessary and sufficient conditions to determine class membership. This makes it possible to infer that a resource belongs to a certain class, given a set of properties and values.

Ontologies are data models that represent selected concepts within a domain and the relationships between them. They are typically used to reason about objects within a domain. Fensel provides a comprehensive description of ontology languages, tools and applications in [Fens03]. It is often pointed out that the real challenge related to Semantic Web standards is not the standardization of representation formalisms for ontologies, such as OWL, but the standardization of ontologies itself. Roughly put, this position argues that the specification of a set of terms for describing a specific domain is critical. Whether this set of terms is specified in XML-S, RDF-S or OWL is secondary. This is analogous to standardization in EDI. Fixing the syntactic format is comparatively easy compared with the task of agreeing what exactly should be included in an exchanged message.

3.1.2.3 Benefits and Challenges

So far, real-world applications of Semantic Web technologies are difficult to find. Most existing applications have been developed in academic contexts or in research projects funded by public institutions. Descriptions of functionality and benefits thus tend to refer to the vision of the Semantic Web and not to experiences gained in actual applications.

The functionality expected from the Semantic Web centers around the enhancement of information with metadata. As explained in section 3.1.2.2, work on RDF and OWL can

¹ DL is a class of logic-based knowledge representation languages. For more information on DL, see <http://dl.kr.org/>.

only be seen as a first step towards this goal. These standards provide formats in which metadata can be represented. They do not provide a set of metadata to be used, nor do they say anything about how this metadata can be obtained.

The Semantic Web is supposed to give machines the ability to solve problems by performing operations on well-defined data. Information semantics and their conceptual associations are explicitly defined within the Semantic Web framework. Through explicit semantic definitions, meanings embedded in data, applications and systems become readily accessible to computer programs that can process them at high speed. It is agreed that the Semantic Web will allow for a much better position to cope with the challenges of disparate data sources, information overload and complexity of systems. The benefits manifest in better information sharing, more effective information management, more intelligent search and smarter decision-making through machine reasoning and inferences.

3.2 Infrastructure and Usage

A general SWS infrastructure is discussed comprehensively by Cabral et al. in [CDMP04]. They characterize the infrastructure along three orthogonal dimensions:

- *usage activities*, which define functional requirements;
- the *architecture*, which defines the components needed for accomplishing these activities; and
- the *service ontology*, which aggregates all concept models related to the description of SWSs and constitutes the knowledge-level model of the information describing and supporting the usage of the services.

Arroyo and López-Cobo describe usage activities of SWS infrastructures in [ArLó06] extensively. The SWS usage activities and the usage process are described in section 3.2.1. In section 3.2.2, we outline the architecture and in section 3.2.3, the service ontology.

3.2.1 Usage Activities and Process

The activities required to run an application using SWSs are depicted in Figure 16 and include not only publishing, discovery and execution but also composition, mediation, selection, replacement, monitoring, compensation and auditing [ArLó06].

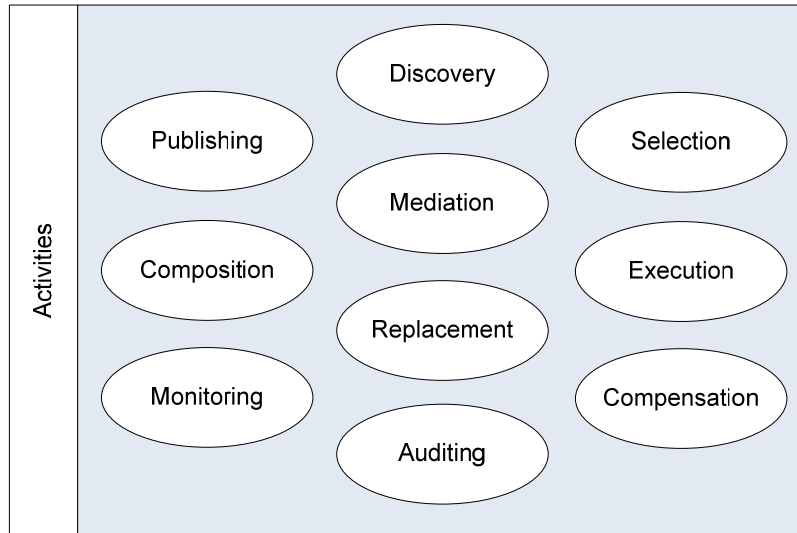


Figure 16 SWS infrastructure usage activities.

Along with the core steps of publication, selection, discovery, composition, execution and mediation, the execution of SWSs requires various other important activities to be performed. Under the term *execution support* we summarize activities such as monitoring, compensation, replacement and auditing.

In [CDMP04], the deployment of services and the management of service ontologies are adduced as usage activities. The deployment of a service by a provider is independent of the publishing of its semantic description because the same Web service can serve multiple purposes. Nevertheless, SWS infrastructures can provide a facility for the instant deployment of code for a given semantic description. The management of service ontologies is a cornerstone activity for SWSs due to the fact that it ensures that semantic service descriptions are created, accessed and reused within the Semantic Web.

In [ArLó06], Arroyo and López-Cobo outline the Web service usage process as follows. Essential for a successful discovery is that the publisher of a service registers the service at the registry and describes its capabilities. To discover a service, a requester must specify a goal and translate it into a machine-readable discovery query. Such goals are then decomposed into constituent subgoals and matched against the capabilities of the registered services, choosing the ones that alone or in combination with others allow achieving the objective.

Mediation is required during the discovery phase to allow services described, using different domain knowledge, to be matched against the goal. A selection process is then carried out to pick the most suitable service or services from the set of alternatives based

on functional and nonfunctional properties of the services. The list of preferred services is then made available to the composition stage. In case composition is required because one service is not sufficient to acquire the goal, the list of services is assembled in a way that enables their interoperation. Again, mediation is required to overcome mismatches and to allow interoperation with regard to data formats used in the message exchange, protocols used to interact and specific business models. Finally, the service is executed at the expenses of the service provider. In case of composed services, execution requires additional support for monitoring, compensation, replacement and auditing since during service execution various issues could lead to failure.

Monitoring controls the execution process. It is concerned with determining the current state of a service execution or finding out what state a service is in, if a problem is reported. In case of a problem, compensation undoes or mitigates the unwanted effects of the execution and provides transactional support. To continue with normal execution, a new service or a combination of new services could be required. Replacement facilitates the substitution of services by suitable equivalents. To that, the usage process is started from the beginning, trying to discover services having capabilities that match those of the malfunctioning service. Auditing is the last step in the execution of a service. It verifies that a service execution occurred in the expected way.

In section 3.2.1.1, the basic ideas of service publication are outlined. Section 3.2.1.2 and 3.2.1.3 describe service discovery and execution, respectively. Service composition is detailed in section 3.2.1.4 and the key aspects of mediation are delineated in section 3.2.1.5. Finally, section 3.2.1.6 explains the principles of service selection.

3.2.1.1 Publishing

The publishing of SWSs allows agents and applications to discover services based on their goals, capabilities, interfaces and nonfunctional properties. Service publishers are in charge of describing the main aspects of services and making that information available for discovery. The semantically marked-up descriptions of services are registered in service repositories, where they can be located easily. Service publication allows for matching the capabilities of Web services against the goals of requesters.

3.2.1.2 Discovery

The discovery of services can be understood as matchmaking between service capabilities and the goals of service requesters. The idea is to locate a selection of services which, by themselves or in combination with others, allow the accomplishment of a goal. For this, the goal is decomposed into subgoals that are submitted as discovery queries. The matching can be done at the level of tasks or goals to be achieved, followed by the selection of appropriate services. Matching requires reasoning support for both goal capability resolution and domain knowledge mediation that is typically not provided as part of the registry functionality.

3.2.1.3 Execution

When the available services have been composed and the service requester has chosen the execution path that best suits its needs according to the functional and nonfunctional requirements, the chosen set of Web services is executed.

Service execution deals with the dynamic invocation of services. Middleware frameworks such as CORBA have already addressed the problem in distributed, object-oriented programming. Essentially, the service requester must determine how to invoke a service without having a static proxy. Introspection and brokers are well-known techniques used in object-oriented programming to do this. With regard to Web services, the grounding mechanism allows the service requester to go from the conceptual to the concrete descriptions of Web services.

Each Web service specifies signatures for its operations that allow executing the operation, defining the inputs required and the outputs returned, describing the format of exchanged messages, figuring out how to physically send a message to the service, and following a specified protocol for interacting with the service. The semantic markup of services makes all this information available, thereby realizing service execution in a more dynamic way.

Before their execution, services may impose some limitations on the service requester. Such restrictions are expressed in terms of assumptions. In addition, service execution requires facilities offered by composition engines to coordinate and control the whole process, particularly with regard to the complex process logic of composite services.

3.2.1.4 Composition

Composition allows services to be defined in terms of other, simpler services. A workflow expressing the composition of atomic services can be defined in the service ontology by using appropriate control constructs. The definition would be grounded on a syntactic description such as WS-BPEL. Dynamic composition is also being considered as an approach during the service request, in which the atomic services required are located and composed on the fly. This requires an invoker to match the outputs of atomic services against the input of the requested service.

3.2.1.5 Mediation

Mediation is, according to Arroyo and López-Cobo, the pivotal element of the usage process. It is interleaved with all other activities of the usage process and relies on the use of ontologies. Depending on how the interfaces of the interacting parties are defined, problems arise when communicating with a number of providers because the messages must be translated specifically for each provider. Ideally, a mediator resolves the differences in representation between the requester and the provider. Once all the services required to solve a task are brought together following some programmatic conventions, interacting services must be mediated in terms of types and meaning of the exchanged information and protocols, and business models used [ArLó06]:

- depending on the application domain for which the services are deployed, different data types and domain knowledge might be used to encapsulate data and its meaning, which requires mediation to allow interoperation;
- different parties might use different message exchange patterns and protocols that must be mediated to enable communication; and
- services belonging to different business models require appropriate process mediation to permit their cooperation.

3.2.1.6 Selection

A selection of services is required if more than one service matches a request. Nonfunctional attributes such as cost or quality are common criteria for choosing a service in this case. In a more specialized or agent-based interaction, a negotiation process can be started between a requester and a provider, but that requires the services themselves to be knowledge-based. In general, a broker checks that the preconditions of tasks and services

are satisfied and proves that the services post-conditions and effects imply goal accomplishment.

3.2.2 Architecture

The SWS architecture is defined by a set of components that realize the activities described in section 3.2.1. Between a requester and a provider, components such as a register, a reasoner, a matchmaker, a decomposer and an invoker are needed. They illustrate the required roles in SWS architectures and have different names and a complexity of their own in different approaches. The components described in [CDMP04] are depicted in Figure 17. Underlying security and trust mechanisms are typically also discussed under the architecture perspective.

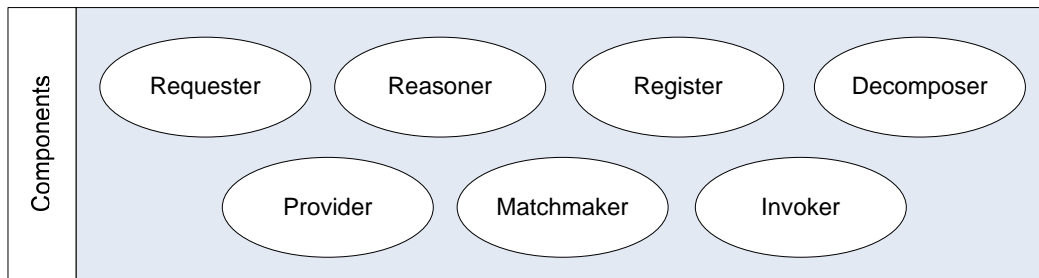


Figure 17 SWS infrastructure components.

The *reasoner* is used during all activities. It provides reasoning support for interpreting semantic descriptions and queries. The *register* provides the mechanisms for publishing and locating services in a semantic registry as well as functionalities for creating and editing service descriptions. The *matchmaker* mediates between the requester and the register during discovery and selection of services. The *decomposer* component is required to execute the composition model of composed services. The *invoker* mediates between requester and provider or decomposer and provider when invoking services.

3.2.3 Service Ontology

The service ontology represents the capabilities of Web services as well as the restrictions that have to be considered when using them. It integrates the information defined by Web service standards such as UDDI and WSDL with related domain knowledge. This includes functional capabilities such as inputs, output, preconditions and post-conditions as well as nonfunctional capabilities such as category, cost and quality of service. Provider related information such as company name and address, task or goal-related information, and

domain knowledge defining, for instance, the type of inputs of the service, are also included. This information can be distributed in several ontologies. However, a service ontology used to describe SWSs relies on the expressivity and inference power of the underlying ontology language.

3.3 Significant Frameworks

OWL-S and WSMO are the most significant SWS frameworks. Both have been submitted to the W3C to set up standardization efforts. Other related effort are SWSF¹ (Semantic Web Services Framework) which has also been submitted to the W3C, IRS (Internet Reasoning Service), and METEOR-S. IRS², described in [MDCG03], is the SWS framework of KMi at The Open University, which allows applications to semantically describe and execute Web services. IRS supports the provision of semantic reasoning services within the context of the Semantic Web. The METEOR-S project³ at the LSDIS Lab at the University of Georgia aims, according to [VGSM05], to extend current Web service standards with Semantic Web technologies to achieve greater dynamism and scalability. The SAWSDL⁴ (Semantic Annotations for WSDL) Working Group was started by the W3C in April 2006 with the objective of developing a mechanism to enable the semantic annotation of Web service descriptions. This mechanism takes advantage of the WSDL 2.0 extension mechanisms to build simple, generic support for adding semantic descriptions to Web services. The SAWSDL specification became a candidate recommendation in January 2007. This means that the specification has been widely reviewed and the W3C recommends its implementation.

Nevertheless, current efforts in SWS technology research and development gather primarily around OWL-S and WSMO. Today, the main lines in research relate to the composition of SWSs, the establishment of a semantic environment for execution and the reasoning needed for the automated discovery of services. Lara et al. provide a comprehensive conceptual comparison of OWL-S and WSMO in [LRPF04]. Cabral et al. discuss and compare OWL-S, WSMO and IRS in [CDMP04].

We introduce OWL-S in section 3.3.1 and WSMO in section 3.3.2. Besides the two publications mentioned previously, we mainly used, if not otherwise stated, resources in

¹ For more information on SWSF, see <http://www.w3.org/Submission/SWSF/>.

² For more information on IRS, see <http://kmi.open.ac.uk/projects/irs/>.

³ For more information on METEOR-S, see <http://swp.semanticweb.org/>.

⁴ For more information on SAWSDL, see <http://www.w3.org/2002/ws/sawSDL/>.

form of working drafts and presentations provided on the websites of the approaches or the W3C, for our discussion.

3.3.1 OWL-S

OWL-S (formerly DAML-S) is an ontology of services that enables users and software agents to discover, invoke, compose and monitor Web resources that offer particular services and have particular properties. These functionalities are carried out with a high degree of automation, if desired. OWL-S was submitted to the W3C in November 2004¹ and is often regarded as a quasi-standard. One of the most important elementary publications on OWL-S is [MPMB04].

OWL-S provides a core set of markup language constructs for describing the properties and capabilities of Web services in an unambiguous, computer-interpretable form. OWL-S markups allow the automation of Web service tasks including discovery, execution, composition and execution monitoring. Following the layered approach of markup language development, the current version of OWL-S builds on top of OWL.

OWL-S consists of a set of ontologies designed for describing and reasoning over service descriptions. It combines the expressivity of description logics and the pragmatism found in the emerging Web service standards. OWL-S shares the vision of WSMO as ontologies are considered to be essential to support automatic discovery, composition and interoperation of Web services. But despite sharing a unifying vision, OWL-S and WSMO differ greatly in the details and the approach to achieving these results. Whereas OWL-S explicitly defines a set of ontologies that support reasoning about Web services, WSMO defines a conceptual framework within which these ontologies must be created. Another difference between OWL-S and WSMO is that while OWL-S makes no distinction between types of Web services, WSMO does.

WSMO places much stress on the specification of mediators that map programs to solve interoperability problems between Web services. As opposed to WSMO, OWL-S does not explicitly address the issue of mediation. With OWL-S this issue is considered to be handled by the underlying infrastructure. OWL-S provides to Web services and their clients with the information needed to find existing mediators that can reconcile their mismatches, or to create mediators through the process of Web service composition.

¹ For more information on the submission of OWL-S to the W3C, see <http://www.w3.org/Submission/OWL-S/>.

OWL-S seems to be more mature than WSMO in some aspects, including choreography and grounding specifications [LRPF04].

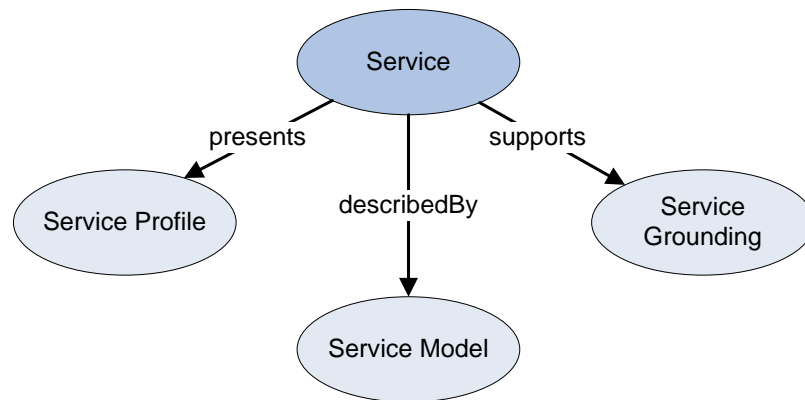


Figure 18 OWL-S upper ontology.

As mentioned earlier, OWL-S facilitates the description of services in terms of four different ontologies. They are illustrated in Figure 18. The key ontologies of OWL-S, the upper service ontology links to, are:

- the *service profile* for advertising and discovering services;
- the *process model* which gives a detailed description of a service's operation; and
- the *grounding* which provides details on how to interoperate with a service via messages.

The service profile ontology is described in section 3.3.1.1. In section 3.3.1.2, we sketch the service model ontology and, in section 3.3.1.3, the service grounding ontology.

3.3.1.1 Service Profile

The profile is used to describe services for the purpose of discovery. Service descriptions and queries are constructed from a description of functional properties such as inputs, outputs, preconditions and effects, as well as nonfunctional properties. In addition, the profile class can be subclassed and specialized, thus supporting the creation of profile taxonomies that subsequently describe different classes of services.

3.3.1.2 Process Model

The process model describes the composition or orchestration of one or more services in terms of their constituent processes. This is used for both reasoning about possible compositions and controlling the invocation of services. Three process classes have been defined: composite, simple and atomic.

An *atomic process* is a single, black-box process with exposed functional properties. Inputs and outputs relate to data channels, in which data flows between processes. Preconditions specify facts of the world that must be asserted for an agent to execute a service. Effects characterize facts that become asserted given a successful execution of the service, such as the physical side-effects of the execution of the service on the physical world.

Simple processes provide a means of describing service or process abstractions. Such elements have no specific binding to a physical service, and thus must be realized by an atomic process or expanded into a composite process.

Composite processes are hierarchically defined workflows consisting of atomic, simple and other composite processes. These process workflows are constructed using a number of different composition constructs such as decisions, forks and loops. The profile and the process model provide semantic frameworks whereby services can be discovered and invoked, based on conceptual descriptions defined within ontologies.

3.3.1.3 Grounding

The grounding provides a pragmatic binding between the concept space and the physical data, machine and port space, thus facilitating service execution. A process model is mapped to a WSDL description of the service through a thin grounding. Each atomic process is mapped to a WSDL operation, and the OWL-S properties used to represent inputs and outputs are grounded in terms of XML data types. Additional properties pertaining to the binding of the service are also provided.

3.3.2 WSMO

OWL-S is not the only mechanism under development to support SWSs. WSMO is also being developed to solve SWS challenges. It provides a conceptual framework and a formal language for semantically describing relevant aspects of Web services to facilitate the automation of discovering, combining and invoking services over the Web. Unlike OWL-S, the WSMO specification focuses on a workflow ontology for Web services that

can capture rich semantics about interfaces and exchanges [PoHo04]. One of the most important elementary publications on WSMO is [FRPD05]. WSMO was submitted to the W3C in June 2005¹.

WSMO is based on WSMF (Web Service Modeling Framework). WSMF, described at length in [FeBu02], separates the elements needed to describe services into three areas: Web services, goals and mediators. WSML (Web Service Modeling Language) is the language used to describe these elements. WSMX (Web Service Modeling Execution Environment) is a reference implementation of WSMO [HCMO05].

While in OWL-S, the service model makes no clear distinction between choreography and orchestration, in WSMO, choreography and orchestration are specified in the interface of Web service descriptions. Choreography describes the external, visible behavior of the service and orchestration describes how other services are composed to achieve the required functionality of a service. OWL-S allows only one service model per service and hence there is only one way to interact with each service. WSMO, as opposed to OWL-S, allows the definition of multiple interfaces for a single service.

WSMO provides a more complete conceptual model than OWL and addresses additional aspects such as goals and mediators [LRPF04].

The four main elements of WSMO are:

- *ontologies*, which provide the terminology used by other WSMO elements;
- *Web service descriptions*, which describe the functional and behavioral aspects of Web services;
- *goals* that represent user desires; and
- *mediators*, which aim at automatically handling interoperability issues between different WSMO elements.

The use of ontologies in WSMO is described in section 3.3.2.1. Web services are outlined in section 3.3.2.2 and goals in section 3.3.2.3. Finally, in section 3.3.2.4, we sketch the use of mediators.

¹ For more information on the submission of WSMO to the W3C, see <http://www.w3.org/Submission/WSMO/>.

3.3.2.1 Ontologies

Ontologies are a key element in WSMO because they provide terminologies for describing the other elements of the framework. Within WSMO, ontologies serve two purposes: they define the formal semantics of information, and they link machine and human terminologies. WSMO specifies the following constituents as parts of the description of an ontology: concepts, relations, functions, axioms, and instances of concepts and relations as well as nonfunctional properties, imported ontologies and used mediators. The latter allows the interconnection of different ontologies by using mediators that solve terminology mismatches.

3.3.2.2 Web Services

As mentioned earlier, each Web service represents an atomic piece of functionality that can be reused to build more complex services. To allow discovery, invocation, composition, execution, monitoring, mediation and compensation of Web services, Web services are described in WSMO from three points of view: nonfunctional properties, functionality and behavior. Thus, a Web service is defined by its nonfunctional properties, its imported ontologies, its used mediators, its capability and its interfaces. A Web service can be described by multiple interfaces but has only one capability. The capability of a Web service encapsulates its functionality and an interface of a Web service describes the behavior of the Web service from the perspective of communication and collaboration.

3.3.2.3 Goals

Goals describe aspects of user desires with respect to the requested functionality as opposed to the provided functionality described as the Web service's capability. In WSMO, a goal is characterized by a set of nonfunctional properties, imported ontologies, used mediators, the requested capability and the requested interface.

3.3.2.4 Mediators

Mediators describe elements that aim to overcome structural, semantic or conceptual mismatches that appear between the different components that build up a WSMO description. The current specification covers four types of mediators:

- mediators that import the target ontology into the source ontology by resolving all representation mismatches between the source and the target;

- mediators that connect goals that are in a relation of refinement and resolve mismatches between those;
- mediators that link Web services to goals and resolve mismatches; and
- mediators that connect several Web services for collaboration.

3.4 Potential Applications

SWSs combine the Semantic Web and Web services in two ways: they offer Web service interfaces to Semantic Web technologies and allow to semantically describe Web service interfaces. Berlecon Research analyzed SWS application areas during the DIP project and ranked them according to their potential to apply and benefit from SWSs in the near- to medium-term future. The results were published in a project deliverable. Section 4.2.2 describes the DIP project in more detail. Berlecon Research was member of the DIP project consortium.

The areas differ with respect to maturity and the degree to which they would benefit from integrating SWS technologies. As per Berlecon Research, business process management (BPM), content syndication, enterprise collaboration, search and mining as well as social software are application areas with particularly high potential.

The assessment of the potential of SWSs for the application areas is based on two main types of functionality provided by SWSs:

- SWSs enhance Web services with semantic descriptions. These semantic descriptions allow the automatic discovery of services and the composition of service operations into complex processes. They also support the evaluation and monitoring of process execution based on semantic criteria.
- SWSs provide functionality based on Semantic Web technologies with Web service interfaces. These interfaces make Semantic Web functionality available over the Internet, so that it can be used by arbitrary applications.

In this section we briefly sketch the application areas and discuss possible scenarios for using SWSs. The list of application areas is not necessarily complete. However, it provides a good starting point for assessing the potential of SWSs in real-world applications. BPM is presented in slightly more detail in section 3.4.1 because it is most relevant for the kind

of application of SWSs discussed in this work. In section 3.4.2, content syndication is described. In sections 3.4.3 and 3.4.4, enterprise collaboration as well as search and mining are outlined with regard to SWSs. The potential of SWSs in social software is not discussed within the scope of this work.

3.4.1 Business Process Management

BPM is concerned with the modeling, automation, administration, monitoring, measuring, evaluation and optimization of business processes. It combines several existing technologies such as workflow tools, EAI products and BI software. Depending on their background, BPM offers tend to focus on technical integration, workflow modeling or process monitoring.

Technical integration concerns the automation of process execution and the task of connecting the central business process to the various systems involved in the process. This is the area in which traditional EAI vendors possess considerable know-how. Berlecon Research claims that the EAI field will vanish as an application area within the next years and will instead merge completely with BPM.

Workflow modeling, on the other hand, focuses on providing formalisms, methodologies and tools, supporting the graphical design of complex processes. The main challenge with respect to workflow modeling is to facilitate the design process and to hide the complexities of the technical format used to internally represent workflows from the end user.

Finally, *process monitoring* addresses the issue of how to collect and present information on process execution. This concerns real-time monitoring and administration of running services as well as the retrospective analysis of service executions.

Despite these different focal points, the various players in BPM share the same overall vision. They aim at providing solutions that allow business experts to model, monitor and optimize processes on the business level. According to Berlecon Research, the underlying technological details should be hidden from the business experts and adapted as automatically as possible according to the actions performed on the business level. Current offerings still lack this deep integration and provide only partial functionality. Therefore, the manual effort to map high-level business process models to actual software implementations is still intensive.

With the advent of Web services and the idea of SOAs, services have become one of the most obvious building blocks for constructing processes. This made Web services a key technology for BPM.

As per Berlecon Research, there are at least two aspects of BPM directly related to the Semantic Web:

- metadata is considered highly useful in modeling, monitoring and evaluating business processes; and
- business rules are often cited as an important ingredient or extension to BPM.

There are two main application scenarios for SWS technologies within the scope of BPM:

- Semantic descriptions of services could make it easier for business experts to find appropriate services when modeling complex business processes. SWS technologies could also facilitate the combination and arrangement of services into complex processes.
- When processes are executed, input and output data as well as timestamps are usually logged in databases. This logged information could be the basis for online monitoring, auditing and tracking, or for offline evaluation and reporting of process performance. Based on semantic annotations of resources, people and roles involved in the processes, such evaluations could be performed on a semantic level.

It is difficult to imagine alternatives to these applications of SWSs. The probability that BPM will use Web services as underlying technology is very high. That metadata will be needed to improve BPM quality in areas such as monitoring is also almost certain.

3.4.2 Content Syndication

Metadata could be used to provide information on content to be syndicated. This would be especially useful for text-based content without explicit formal structure. RSS¹ (Real Simple Syndication) is an increasingly popular XML-based format for exchanging and integrating content from news sites and weblogs. More and more sites are providing their news and weblogs via RSS feeds, which can then be centrally read by RSS readers.

¹ For more information on RSS, see <http://www.rssboard.org/>.

Different versions of RSS exist, one of which, RSS 1.0, is based on RDF. Thus, RSS 1.0 is a good candidate for the integration of additional Semantic Web technologies. Atom¹, a competitor to RSS, is under development and aims to support a broader functionality than RSS. It will be specified in XML but not in RDF, and Web service interfaces in WSDL are envisaged. It is not clear yet whether it will address the issue of semantic categories for content classification. According to Berlecon Research, Semantic Web technologies could be used to automatically categorize content which in turn would allow filtering and personalization and thus make RSS readers much more user friendly. The most likely scenario would be the implementation of Web service interfaces for categorization and classification tools.

Basically, such services would receive free text content as input and would return semantic categories or other metadata describing the semantic content of the input. To increase flexibility, they might also support more than one classification scheme and thus allow users to select their preferred classification scheme when invoking the service. These services could be called after content is created by the content provider, or when content is syndicated by the content consumer.

3.4.3 Enterprise Collaboration

Enterprise collaboration could make use of full-fledged SWS technologies by providing semantically described Web service interfaces for collaboration components. This would include APIs for both accessing collaboration functionality and for exchanging data between components.

On the one hand, this would allow automatic discovery and composition of adequate collaboration functions in a given context. On the other hand, semantically rich data could be passed to the services as arguments and in succession allow smarter handling of information passed between collaboration components.

Whether SWS technologies are selected will probably depend on how fast standards are available for use in enterprise collaboration. This does not mean technical standards but rather standardized ontologies or metadata for content used in enterprise collaboration.

¹ Atom was developed because of the existence of many incompatible versions of RSS. For more information on Atom, see <http://www.w3.org/2005/Atom>.

According to Berlecon Research, enterprise collaboration has high potential for SWSs. This emerging area is still open for innovation and standardization. Modeling interfaces for collaboration modules as SWSs could significantly improve their usability and help users to combine basic operations into complex workflows.

3.4.4 Search and Mining

Search and mining are distinct though related application areas. While search is more concerned with finding texts and documents, mining supports the exploration of semi-structured and structured data. As per Berlecon Research, search and mining will probably converge in the coming years and the application of Semantic Web technologies might even speed up this convergence, as it would enhance the potential of both techniques, which are currently still rather syntactical. In particular, the market of enterprise search is expected to grow rapidly in the near future, making it one of the key markets for IT innovation.

Because search is closely related to Internet and intranet applications, Web services are a natural candidate for the integration into other applications. It is often claimed that search and, to a lesser extent, mining results would be of higher quality if queries were interpreted semantically instead of purely syntactically. This would require the interpretation of queries as semantic categories instead of just plain keywords. Thus, when entering a search term, semantic search engines would not only retrieve documents containing a specified term but also documents containing other terms belonging to the same semantic category. According to Berlecon Research, special purpose search will become increasingly important. Due to the fact that the results of specialized searches are very specific types of information, it is possible to develop standardized formats for describing them.

SWSs provide ideal technologies for implementing interfaces to both specialized search systems and semantic search engines. A major contribution of SWSs would be the use of ontologies to standardize information returned by specialized search systems and to categorize Web content. As per Berlecon Research, it would also be feasible to initially develop such interfaces on top of existing traditional search engines. SWSs could then take a search term, use an ontology to map it to a list of keywords, send this list to a standard search engine and return the results.

Similarly, SWSs could take the results of a specialized search system and transform it into a list of semantically described results. The benefits of integrating semantic information

might only be fully exploited if semantic information is tightly integrated with indexing and retrieval techniques.

Mining could benefit from using semantically described services or operations as basic building blocks of processes. This would allow the use of semantic information when evaluating service executions.

Chapter 4

SWS-enabled E-Business

The promise of SWS technologies is that they will make dynamic enterprise integration possible for enterprises of all types and sizes, unlike more static and difficult-to-manage, traditional technologies. Enterprise integration is also expected to become more reliable and easier to achieve without the low-level implementation problems common with today's approaches. However, a new technology can only be justified by successful applications.

Section 4.1 contains a comprehensive literature review of recent publications on SWS-enabled e-business. Section 4.2 introduces several real-world case studies and section 4.3 briefly summarizes trends identified from the literature review.

4.1 Literature Review

According to [BuFS05], three domains have recently begun to draw enormous attention throughout academia and industry and are of relevance to computer science and the business world:

- Web service technologies, manifested through SOAP, WSDL and UDDI;
- Semantic Web technologies, manifested through ontology languages; and
- enterprise integration manifested through A2A, B2B and hosted application integration.

Few current efforts have attempted to define a coherent model in the form of an SWS framework. As described in section 3.3, OWL-S and the more recent approach WSMO are the most significant ones. Most scientific publications refer to one of these two SWS frameworks. The following overview reveals the breadth and maturity of the research that has been or is currently being conducted on the application of SWSs in e-business.

Friesen and Namiri, for instance, introduce in [FrNa06] an approach to dynamic service selection based on a service description's semantic interpretation of its capabilities and the parameters specifying a request. The proposed solution takes into account conditions of service usage, which can be contractually agreed upon or specified by the requester.

In [EsPW04], Esplugas-Cuadrado et al. present a case study using SWSs based on OWL-S to integrate a new partner into a freight logistics chain. The case study describes the process of substituting one freight forwarder with another logistic partner, where the substitution has to result in the same functionality as that which existed before the substitution. Esplugas-Cuadrado et al. describe the service description requirements encountered during the development.

In [YZGZ05], Yang et al. discuss an approach for constructing a service-oriented manufacturing environment wherein many functions of an enterprise are virtualized as Web services, and SWS technologies are used to enhance manufacturing Web services. Yang et al. use OWL-S for the semantic markup of Web services.

In [KiCL05], Kim et al. present a framework called Web Service-based Coordinated Process Collaboration (WSCPC) that facilitates collaborative design and manufacturing processes using Web services. In WSCPC, a process is structured as a network of activities, and each activity is represented as a service that can be advertised and identified through Web semantics based on OWL-S. WSCPC also presents service models which represent manufacturing behavior, facilitate the complex communication required for collaborative process management and help enterprises devise an optimal solution.

In [ARBR04], Al-Naeem et al. introduce a design methodology with which an enterprise can define and develop its B2B connectivity with other enterprises. The authors use a running example of inter-enterprise e-payment that deals with service providers, service consumers, and their respective banks and inter-bank clearing organizations. A design process methodology – based on the possible choices of process models, architecture models and implementation technologies – which, working in terms of specific B2B

integration needs, outline the steps that can be taken to derive a successful B2B integration with trading partners is presented.

In [PoTJ06], Podobnik et al. propose an auction-based approach to solving the problem of mediation between buyers and sellers in electronic markets. The approach uses provider agents, which can autonomously advertise semantic descriptions of available services, while requester agents filter the advertised services through a two-level filtration system to identify eligible services.

In [SMSV04], Sivashanmugam et al. describe the concept of semantic process templates, which constitute a workflow-style approach to modeling semantic processes using activities, data flow and control flow as prominent concepts. Web services can also be invoked from semantic process templates, as illustrated by the example of product purchasing. A discovery mechanism based on semantic and quality-of-service matching is provided which allows developers to request lists of all matching Web services from repositories in order to select the appropriate ones to be put into the semantic process template. Once a Semantic Web template has been established, executable processes can be generated. The approach currently supports BPEL environments.

Losada et al. describe in [LKBC06] how the deficiencies of conventional SOAs based on Web services can be overcome by applying SWSs. The authors use the example of a mortgage simulator to show how SWSs enable faster and cheaper B2B integration by automating the processes of service discovery, composition and execution. The illustrated application is based on WSMO.

In [HKMV06], Haselwanter et al. present a B2B integration scenario building on the principles of SWSs that shows the benefits of semantic descriptions used within the integration process. Semantic descriptions are employed to enable conversation between systems with the aid of data and service process mediation. The approach is illustrated on the basis of WSMX.

The notion of SOAs is enlarged to include applying SWS technologies by Haller et al. in [HaGB05]. The authors examine what current SWS technologies offer with respect to requirements imposed by A2A integration scenarios and point out the challenges for SWS frameworks to fully enable dynamic service discovery and execution in A2A integration. Within the scope of the publication, WSMX is presented as adequate basis for a semantic SOA.

Preist et al. present in [PEBG05] a system which applies SWS technologies to B2B integration, focusing specifically on a logistics supply chain. The system described is able to handle all stages of the service lifecycle, including service discovery, selection and execution. An interesting feature of the presented approach is its protocol mediation, wherein service requestors are able to dynamically modify the way they communicate with providers, based on a description of the providers' protocols.

In [KVHR06], Kotinurmi et al. present another approach to accomplish dynamic, heterogeneous B2B integration systems based on WSMX. The authors focus on how WSMX can be enhanced to support the RosettaNet e-business framework and how it can add dynamics to B2B interactions by automating the mediation of heterogeneous messages. According to Kotinurmi et al., the benefits of applying SWS technologies include more flexibility in accepting heterogeneity in B2B integrations and simplifying back-end integration.

The publications sketched discuss either selected SWS usage activities such as service discovery, composition and invocation, application scenarios of integration architectures based on SWSs, or related integration methodologies and concepts. The enhancement of the B2B connectivity of enterprises and the facilitation of business processes are examples for typical application scenarios. Our literature review shows that there is not only a lively discussion but also promising early results with respect to the potential of SWS technologies in e-business in general and advanced enterprise integration in particular.

4.2 Business Scenarios and Case Studies

The collaborative work of multi-skilled teams – including industrial users, methodologists and solution providers – within projects such as ATHENA and DIP aims at closing the gap between academic research and industry needs by facilitating the users' deeper understanding of research trends, as well as the researchers' deeper understanding of industrial requirements. While researchers approached integration issues within the scope of the ATHENA project with a palette of advanced technologies at hand, the researchers involved in the DIP project explicitly intended to perform integration through SWSs.

Sections 4.2.1 and 4.2.2 describe a list of real-world business scenarios and case studies, the summaries of which are based on documents publicly available on the respective project websites. The business scenarios and cases studies show typical situations in which

integration is necessary. An explanation of how SWSs are intended to be applied is also given in the context of the cases studies of the DIP project. More information on the business scenarios and case studies can be obtained from the project websites.

4.2.1 ATHENA

ATHENA¹ (Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application) was an integrated project sponsored by the European Commission and conducted from February 2004 to January 2007 with a total committed project budget of more than € 26 million. ATHENA aimed to enable enterprises to interoperate seamlessly with others and spanned the full spectrum from technology components to applications and services. Its multi-disciplinary research was particularly focused on industrial relevance.

Within the scope of the project, specific interoperability requirements were extracted through the analysis of real-world business scenarios based on the needs and interests of the industrial partners in the project. The pilots covered the aerospace, the automotive, the telecom and the furniture industry, and the pilot activities included the identification and implementation of test cases, test scenarios and test procedures.

The SCM and the e-procurement scenario described in sections 4.2.1.1 and 4.2.1.3, respectively, included trading partners such as suppliers and clients of the main industrial enterprises. The collaborative product development (CPD) and the product portfolio management (PPM) scenario outlined in sections 4.2.1.2 and 4.2.1.4 took the intra-departmental integration of the selected enterprise into consideration, and SWSs were considered for some of the business scenarios.

4.2.1.1 Product Data Management

The ATHENA project dealt among other issues with the sharing and exchange of technical information in the aeronautic sector to facilitate change management and different types of supply-chain relationships. The aeronautic scenario's primary goal was to establish collaboration between organizations that needed to interconnect their diverse change and configuration management processes in a way that allowed interoperability between the product data management (PDM) applications and other software products. Alignment of the processes, applications and software was generally based on using business and

¹ For more information on ATHENA, see <http://www.athena-ip.org/>.

technical standards on which all sides could agree. Product models were integrated by using neutral information models related to the specific disciplines. In addition, the project needed a solution which could be supported easily by all stakeholders' IT departments and that would allow the reuse and alignment of existing manufacturing and IT standards. The PDM project also aimed to allow collaboration among the participating enterprises and faster mapping and transformation processes.

4.2.1.2 Collaborative Product Development

The CPD scenario focused on the product development process that specifies suppliers' involvement in defining objectives and product planning. CPD is a three-phase process that starts with setting the target, moves to choosing the supplier, and ends with the actual product design. Along the way, the interaction between the original equipment manufacturer (OEM) and the suppliers involves the exchange of a lot of information conveyed through the Internet and during meetings. However, in most cases, data is managed by enterprise information systems, and human involvement is necessary to carry it from one system to another.

The core process in this case was the testing, which involved the OEM and the suppliers. The problem to be solved was that there were different applications inside the CPD that managed vehicle testing information in different ways: the data was stored in different databases using different formats, and application integration was accomplished through the definition and management of point-to-point translations that were computationally difficult and expensive. Collaboration among all stakeholders was accomplished by using automatic business process model exchange and data format interoperability. Among other advanced technologies, SWSs were considered for the solution of the problem.

4.2.1.3 E-Procurement

The objective of the e-procurement scenario was to facilitate e-business service interoperability and to implement integration mechanisms by analyzing the e-business implementation level in the furniture sector and promoting multi-sector international agreements. The scenario of the ATHENA project focused on an e-procurement application, through which members of an enterprise would purchase goods from multiple suppliers.

The specific scenario was based on a major furniture manufacturer, which was planning to implement new technologies to assist its interactions with both customers and suppliers. The scenario was divided into two parts in each of which there was either a supplier or a retailer involved in addition to the manufacturer. The supplier part of the e-procurement scenario dealt with the raw material procurement, and the client part dealt with quotations, orders and product delivery. The issues to be solved were:

- a repetitive manual process for regular bulk orders;
- confusion resulting from poor product descriptions;
- missing information;
- unacceptable lag time from product order to delivery; and
- time spent rating suppliers.

The project aimed for a major reduction in incorrect orders, a significant shortening of time from order to delivery and a better integration between internal systems.

4.2.1.4 Product Portfolio Management

The focus of the PPM scenario was the management of product development projects. PPM is particularly important to large enterprises with many business units and complex products. The efficient performance of the product portfolio process requires information from marketing, project execution and product life-cycle management to be aligned. It is also a knowledge-intensive process, as it presupposes an accurate and holistic view of the enterprise, so it requires interoperability in many areas. The PPM scenario studied in ATHENA focused on the intra-enterprise level.

The case investigated the use of model-generated workplaces to support simultaneous project, resource, results management and performance measurement through work management views, and the provision of shared project and work monitoring views. The overall objective was to support collaborative work by providing the actors in the enterprise with the tools, information and communication support they needed to perform their work efficiently.

The system was expected to:

- support faster collaboration and assessment;
- support integration of the various associated enterprise processes, information and roles through a single point of entry;
- provide the different actors with an integrated view of the project, products or development initiatives, depending on their roles;
- provide the actors with up-to-date information related to sales, products, project plans, project status, and available and consumed resources;
- improve communication and coordination between the different participants; and
- facilitate more justified decisions.

The goal in this scenario was to support users in various positions and with different roles, who use the same information but from different perspectives, to perform tasks from strategic and tactical levels to operational levels. Further, an integrated environment for application execution via a model-generated environment was created and existing systems were integrated.

4.2.2 DIP

DIP¹ (Data, Information, and Process Integration with Semantic Web Services) was an integrated project, supported by the European Commission, which ran from January 2004 to December 2006 with a total funding of more than €16 million. The final project review was held in Innsbruck, Austria, in October 2006.

DIP took the vision of SWSs and worked on transforming it into a mature and scalable technology by defining and implementing additional layers of functionality on top of the existing Web service stack, and by mechanizing e-work and e-commerce relationships based on Semantic Web technologies.

Sections 4.2.2.1, 4.2.2.2 and 4.2.2.3 outline the three DIP case studies with regard to dynamic and smart e-business as well as intelligent information management and

¹ For more information on DIP, see <http://dip.semanticweb.org/>.

application integration. Solutions based on SWSs were sought for all of the use cases described.

4.2.2.1 Virtual Internet Service Provider

IT has enabled service providers and their partners to reach new customers and to support the growth into new markets while simultaneously reducing the costs of operation. The use of SWSs allows enterprises to combine existing offerings creatively into new bundles of products and services.

The business model of a virtual Internet service provider (VISP) enables third-party companies to sell and access the products of a partner under their own names. The partner offers a VISP infrastructure for the B2B market to create new virtual services from existing services and to support the building of new virtual enterprise portals.

The semantic approach offers advantages which enable typically non-ISP organizations to create enhanced virtual services based on existing service elements. These existing services were made available as semantically described Web services.

Within the scope of this DIP case study, a VISP platform was created for B2B integration and a corporate group clearing station to support service discovery through the use of existing Web services in novel ways and creating added value by packaging particular sets of services together.

4.2.2.2 E-Government

The e-government case focused on local government authorities in Europe, which needed to develop strategies for delivering more usable and comprehensive services to their citizens and constituents in a cost-effective way in order to comply with the goals of eEurope¹ and the related growth in e-government imperatives to which they were required to respond. They needed to understand and have confidence in the application of technologies which could improve electronic service delivery.

This scenario intended to move current applications to a Web service environment supported by a rich, citizen-focused ontology that facilitated description, discovery and matching of services. The development of applications to enable an e-government supply

¹ The eEurope initiative finished at the end of 2005 and was followed by the i2010 initiative. Both are strategic policy frameworks of the European Commission laying out policy guidelines for the future of the information society. For more information on i2010, see http://ec.europa.eu/information_society/eeurope/i2010/.

chain were planned, resulting in providing citizens and business constituencies of local authorities seamless access and easier service discovery and reuse.

4.2.2.3 E-Banking

In recent years, an increasing number of services have been developed and offered through the Internet. Some of these services are delivered by legacy systems that interface with the Internet, while others have been designed with native Internet technologies. In the area of e-banking services were developed to enable clients to check their balances and make financial transactions.

While e-banking is attracting increasingly more customers, its potential strength for both businesses and consumers has not yet been realized. While most financial institutions offer simple, straightforward information portals, just a few provide advanced services such as financial aggregators, where heterogeneous information is aggregated into a single interface. While those applications are advanced and attractive, their challenge lies in the cost of construction and maintenance.

Potential applications to consider in this DIP case study included mortgage contracting, risk analysis, mobile phone banking transactions, user-friendly interfaces for clients and related possibilities. Basically, the DIP project was intended to introduce SWSs to the financial world.

4.3 Trends

The current literature on the application of SWSs for e-business as well as the business scenarios and case studies discussed through recent research projects make evident that SWSs are not a fad. The application areas of SWSs in the context of enterprise integration are well defined and there are no salient alternatives. On the one hand, SWSs seem to be a prime candidate for facilitating dynamic service discovery, selection, composition and execution in semantic SOAs, but on the other hand SWS technologies have not yet been adopted widely for anything but research prototypes and project pilots. For this reason, it seems worthwhile to put more effort into research concerning the relevance and applicability of SWS-based integration architectures, bringing the opinions from academia and industry face to face.

Chapter 5

Field Study

The main goal of the field study was to collect and quantify the opinions of a clearly defined group of experts on the potential of SWSs as basis for an integration architecture that enables enterprises to link their data processing systems efficiently. We expected that an understanding of the relevance and applicability of SWSs would help to align future research efforts with industry needs effectively. Another goal was to make participating experts from academia and industry more sensitive to the progress and focus of SWS research in general.

In section 5.1, we present the research approach. In section 5.2, we outline the survey implementation and in section 5.3, we describe the data analysis techniques used. Section 5.4 describes the results of the study and section 5.5 summarizes feedback received from participants.

5.1 Research Approach

To achieve our goal, a Delphi study with experts from industry and academia seemed to be particularly suitable. In principle, the Delphi method is based on a structured group process through which experts assess issues about which knowledge is usually uncertain and imperfect by nature. The method allows collecting and distilling knowledge from groups of experts by means of a series of questionnaires interspersed with controlled opinion feedback [AdZi96]. The questions are usually formulated as hypotheses and the experts are encouraged to revise their earlier answers in the light of the replies of other panel

members. It is believed that during this process the range of answers decreases and the group converges towards a common answer.

The classical Delphi approach was defined by Linstone and Turoff but, according to Häder, numerous authors agree on the proposed main criteria that separate the Delphi method from other methodologies [LiTu75, Häde02]:

- the *structuring of the information flow*;
- *regular feedback*; and
- the *anonymity* of participants.

However, many variants and modifications have been discussed in literature.

The initial contributions from experts are collected in the form of answers to questionnaires and their comments about these answers. To enable experts to improve their judgments, the opinions of the participants are grouped and returned in the form of aggregated feedback. The interaction among the participants is controlled through information processing and content filtering. These encroachments help to avoid some of the negative effects of face-to-face panel discussions and solve some of the problems typical for group dynamics [KöSc07]. Participants comment not only on their own forecasts, but also on the responses of others and on the progress of the panel as a whole. At any moment they can revise their earlier statements. While in regular group meetings participants tend to stick to previously stated opinions and often conform too much to the group leader, the Delphi method prevents that.

Usually the anonymity of all participants is maintained. Their identity is not revealed, even after completion of the study. This stops participants from dominating others in the process by using their authority or personality and frees them to some extent from their personal biases, minimizes bandwagon and halo effects, allows them to freely express their opinions, and encourages open critique and admitting errors by revising earlier judgments. The *bandwagon effect* describes the observation that people often do or believe things because many other people do or believe the same. The *halo effect* refers to the cognitive bias in which the assessment of an individual quality serves to influence and bias the judgment of other qualities [Wood06].

The overall track record of the Delphi method is mixed. There have been many cases when the method produced poor results. Häder, for instance, attributes this to poor application of the method and not to weaknesses in the method itself [Häde02]. It must also be realized that in areas such as science and technology forecasting, the degree of uncertainty is so great that exact and fully correct predictions are impossible. Often, a high degree of error must be expected. Furthermore, it must be kept in mind that future developments may not always be predicted correctly by iterative consensus of experts, but instead by unconventional thinking of amateur outsiders.

A basic limitation of the Delphi, as with many other forecasting methods, is its inability to make complex forecasts with multiple factors. Potential future outcomes are usually considered as if they had no effect on each other. Most events and developments, however, are in some way connected to each other. Hence, these interdependencies must be taken into consideration for more consistent and accurate forecasts.

Several extensions to the Delphi method have been developed to address this problem. The cross-impact analysis, for instance, is a suitable mechanism for identifying mutually exclusive or conflicting scenarios. It also takes into consideration the possibility that the occurrence of one event may change probabilities of other expected events. The Delphi method can be used most successfully in forecasting single scalar indicators. Despite these shortcomings, today the method is a widely accepted tool for technology foresight and has been used successfully in many studies.

The Delphi method is also not new to Web-related research. In 2000, Beck, Glotz and Vogelsang used the Delphi method to forecast the development of online communication. The results were published in [BeGV00]. Five years later trends in the field of Semantic Web research were identified through the use of the Delphi method through the KnowledgeWeb¹ project. The findings were published in the form of a project deliverable.

The foresight process has another important effect because it connects the activities of participating experts and thereby causes, according to Aichholzer, a significant increase in communication and commitment. Hence, the benefit of the Delphi method is not limited to the actual findings in content, but the method also fulfils the function of a linking and implementation support tool that gives the communities involved the necessary impulses to overcome persistent problems they face [Aich05].

¹ For more information on KnowledgeWeb, see <http://knowledgeweb.semanticweb.org/>.

To ensure the highest achievable response rates, elements of Dillman's Tailored Design Method (TDM) were adopted for the survey described in this work. The theory underlying the TDM is *social exchange*, which suggests that the likelihood of individuals responding to a survey questionnaire is a function of how much effort is required to respond, and what individuals feel they are likely to get in exchange for completing the questionnaire [Dill00].

Within the scope of our Delphi study, the selected experts were provided two questionnaires. The first questionnaire contained open-ended questions designed to capture the experts' views concerning factors potentially affecting the relevance and applicability of SWS-based integration architectures. The responses from the first phase were aggregated into groups and classified by the unique issues that best summarized their contents. The questionnaire of the second round was based on the responses of the first round. The participants were asked to review the aspects identified in the first round and rank them on structured bipolar rating scales.

Each candidate had about two weeks' time to complete and return each of the two questionnaires. Two rounds were expected to be sufficient to attain a first impression with a reasonable amount of effort. However, a third round would have been desirable to prove the stability in the responses. Nevertheless, as per Häder, further rounds usually tend to show only slight changes in experts' opinions [Häde02].

5.2 Survey Design

Web-based surveys provide capabilities far beyond those available for any other type of self-administered survey technique. They can be designed in a way facilitating a dynamic interaction between respondents and the survey system, which is of particular interest for Delphi studies. Furthermore, Web-based surveys allow the immediate coding of most answers. The potential Web surveys offer for conducting innovative research is enormous. However, it must always be kept in mind that a survey corresponds to a level of technical sophistication that makes it possible for most users to respond to them. Within the scope of the development of our survey system, programming and design steps were taken to minimize the differences across respondents caused by different operating systems, Web browsers and screen configurations.

A substantial number of guidelines for the design of Web-based surveys emerged that go into the processes of questionnaire construction, expert recruitment, survey implementation and correspondence planning. Whenever it seemed worthwhile, we tried to take the guidelines into account in the design of our survey.

The construction of the questionnaires is described in section 5.2.1. Section 5.2.2 explains the considerations for the recruitment of the expert candidates and section 5.2.3 provides details on the implementation of the survey. The technical realization is outlined in section 5.2.4.

5.2.1 Questionnaire Construction

While the questionnaire of the first round of the survey consisted of 15 open-ended questions, the questionnaire of the second round consisted of 454 statements to be rated. The most important independent variable was the candidates' professional background. The participants were asked if they considered themselves to have either an academic or industrial background.

Based on current literature and the case studies and business scenarios described in sections 4.2.1 and 4.2.2, respectively, a facet-theoretical approach was used to formulate the hypotheses. The facet theory, a method for integrating content design with data analysis, was particularly helpful in formulating sets of statements for the second round of the survey by reducing the complexity of the responses from the first round [ShEH94, FoBY06].

The initial 15 questions were formulated as the basis of the questionnaires for the Delphi study. The questionnaires consisted of four parts, structured and formalized in a way that allowed various analyses: a SWOT analysis, a requirements analysis, an analysis of expected effects and a technology roadmap.

In section 5.2.1.1, the questions related to the SWOT analysis are explained. The questions related to requirements are introduced in section 5.2.1.2 and the questions related to effects in section 5.2.1.3. Finally, in section 5.2.1.4, we describe the questions related to future developments in SWS research.

5.2.1.1 SWOT Analysis

Questions 1 to 4 were required to set up the basis for a SWOT analysis. A SWOT analysis is a strategic planning tool used to evaluate the strengths, weaknesses, opportunities and threats involved in an approach to solving a specific problem. Strengths and weaknesses are approach attributes while opportunities and threats are environmental attributes that are either helpful or harmful to achieving an objective. The objective of our SWOT analysis was to evaluate whether the adoption of SWS-based integration architectures leads to significant improvements as compared with traditional approaches. We expected to gather information about the premises of an ideal application of integration architectures based on SWSs.

A SWOT analysis helps to identify the best match between environmental trends and the internal capabilities of an approach by analyzing its strengths, weaknesses, opportunities and threats:

- *strengths* are resources or capabilities an approach can draw on to achieve its objectives;
- *weaknesses* are limitations, faults or defects that keep an approach from achieving its objectives;
- *opportunities* are favorable situations in the environment. They are mostly trends that permit an approach to enhance its position by reacting to them; and
- *threats* are unfavorable situations in the environment that are potentially damaging to an approach. Threats may be a barrier, a constraint, or anything external that might cause problems, damage or injury.

The questions related to the SWOT analysis were:

- (1) Where do you see the strengths of integration architectures based on SWSs?
- (2) Where do you see the weaknesses of integration architectures based on SWSs?
- (3) What factors do you think will drive the use of integration architectures based on SWSs in the future?
- (4) What factors do you think will restrict the use of integration architectures based on SWSs in the future?

5.2.1.2 Requirements

Requirements analysis, the notion behind questions 5 to 7, encompasses those tasks that go into determining the requirements for a new or altered system. Within the scope of this work, we aimed to gather the most important functional and qualitative requirements that integration architectures must fulfill. Furthermore, we tried to shed light on an area in which research has not yet found a clear answer. Our intent was to identify commonly agreed differences in the requirements for internal and external integration architectures.

The questions related to the requirements were:

- (5) What are, from your point of view, the functional requirements that integration architectures must fulfill?
- (6) What are, from your point of view, the qualitative requirements that integration architectures must fulfill?
- (7) Where do you see differences in the requirements for internal and external integration architectures?

5.2.1.3 Expectations

Questions 8 to 11 were stated to identify the positive and negative effects of using SWS-based integration architectures. We expected that this information can be turned into suggestions for near-term measures to improve responses to expected effects. Just as with the questions related to requirements that integration architectures must fulfill, the questions related to positive and negative effects were aimed at finding differences between the academic and industrial experts' points of view. To compare these effects on the strategic or macro and on the operative or micro level, the questions were formulated accordingly.

The questions related to the expectations were:

- (8) What positive effects of using an SWS-based integration architecture do you expect at the macro level?
- (9) What negative effects of using an SWS-based integration architecture do you expect at the macro level?
- (10) What positive effects of using an SWS-based integration architecture do you expect at the micro level?

- (11) What negative effects of using an SWS-based integration architecture do you expect at the micro level?

5.2.1.4 Roadmap

Questions 12 to 15 were formulated to develop a technology roadmap for the future development of SWSs with respect to integration architectures. Question 12 and 13 were related to SWS research in general. Question 14 embodied another approach embracing the potential of SWSs, in particular with respect to the problems of current integration architectures. Question 15 was posed just in the first round of the survey because no real-world case studies were named by the experts who responded to this question. All responses described, as expected, academic research projects rather than mature implementations in industry.

The questions related to the roadmap were:

- (12) What challenges do you know of that SWS research is facing today?
- (13) What will be achieved in SWS research within the next five years?
- (14) What are problems of current integration architectures that you believe can be solved with SWSs?
- (15) Do you know of any real-world case studies where SWS-based integration architectures have been used?

5.2.2 Expert Recruitment

The candidates were selected from academia and industry in equal proportion. Obtaining a reasonable balance was critical because the candidates' backgrounds have a substantial impact on the study results. The candidates were assigned to either the academic or the industrial group according to their affiliation type. Because a significant share of candidates work in both areas and also hold positions in established consortia and initiatives, a precise assignment was difficult. Hence, the final assignment was made by the candidates themselves when they filled out the registration form.

Although, there is no clear consensus in literature, it seems plausible to do a Delphi study with a rather small panel. Duffield, for instance, concludes in [Duff93] that there is typically no reason to use many panel members. The author reasons that smaller panels can be organized more easily and that experiments show that the size of the panel has no

significant influence on the findings. However, because the results of Delphi studies cannot be generalized in the sense of a random sample, an adequate number of candidates must be recruited.

The rate of return after the call for participation was expected to be 30%, and for each of the two rounds of the survey, 70% [Häde02]. When it became apparent that a significantly smaller percentage of the invited candidates would respond, we decided to nominate 500 experts. The sample was a nonprobability convenience sample and consisted of experts from industry and academia in equal proportion.

Repeated contributions to major conferences related to SWSs and publications in the field of SWSs were two of the main criteria used to find suitable representatives of the target population. The candidates were exclusively people involved in at least one of the major international conferences related to SWSs and associated technologies, enterprise integration architectures and middleware solutions, and book authors or members of widely recognized initiatives active in at least one of the related research fields. Some of the most popular international conferences relevant for research on SWSs and associated technologies that were taken into account are the International Conference on Web Services¹ (ICWS), the International World Wide Web Conference² (WWW) and the International Semantic Web Conference³ (ISWC). Conferences related to enterprise integration that were considered are the Interoperability for Enterprise Software and Applications Conference⁴ (I-ESA) and the International Middleware Conference.

The Science Citation Index⁵ and the principle of co-nomination are often used to ensure that the most reputable experts are asked to participate [Cuhl00]. In the relatively small research fields focusing on the aspects of SWSs or enterprise integration, it seemed possible to identify the experts directly. Nevertheless, some candidates provided names of experts who could either also contribute or act in place of them.

There was no explicit test of the experts' expertise as proposed by various authors [RiMH85, RoWB91]. Having people with different levels of expertise in the target population reduces, according to Grupp, Blind and Cuhls, the risk of achieving too optimistic forecasts [GrBC00]. However, importance was attached to the influence of the

¹ For more information on the ICWS conference series, see <http://conferences.computer.org/icws/>.

² For more information on the WWW conference series, see <http://www.iw3c2.org/>.

³ For more information on the ISWC conference series, see <http://iswc.semanticweb.org/>.

⁴ For more information on the I-ESA conference series, see <http://www.i-esa.org/>.

⁵ For more information on the Science Citation Index, see <http://scientific.thomson.com/products/sci/>.

experts, as suggested by Duffield, in order to increase the probability of the findings being put into practice later [Duff93].

Gender and age are not considered criteria for the composition of the expert panel in present literature and hence did not play a role in our study. Geographic region is a relevant criterion in the scope of many Delphi studies, but owing to the limited number of experts in the field and the experts' general independence of their country of employment it was omitted.

To make sure the panel members dealt intensively with the feedback after the first survey round, it was important to ensure that the feedback had strong authority. According to Aronson, Turner and Carlsmith, the shift in the opinion of an expert is primarily a function of the credibility of the other experts and the difference between their own estimation and the group's [ArTC63]. For this reason the participants were informed about the composition and structure of the expert panel. However, a full list of the participants was not made public.

The most critical element during the composition of the panel was the experts' motivation to participate. However, no financial incentive was offered for an expert's participation within the scope of this Delphi study.

5.2.3 Implementation Strategy and Correspondence

Besides questionnaire design, the implementation strategy significantly determines the success of self-administered surveys. According to the TDM proposed by Dillman, among others, attributes of the communication process, frequency of contacts, content of letters, appearance of the survey materials, incentives and personalization have a great collective influence on response rates [Dill00]. Multiple contacts are the most effective technique for increasing response rates, and it has been shown by Schaefer and Dillman that this is also true for e-mail contacts [ScDi98]. Within the course of our study 12 different e-mail messages were sent to the sampled candidates, depending on a set of predefined rules. The minimum number of messages a candidate received was three and the maximum eight. Although the TDM suggests five contacts, recent literature explicitly states that further contacts provide additional opportunities to shape the kind of request made and to improve the response rates through refusal conversion [GrCo98].

All e-mail messages sent to the candidates were personalized to some extent. Because programming word processors has become easy, personalization has lost some effectiveness in recent years. Nevertheless, personalization is an integral part of the TDM and has been proven to improve response rates. Under social exchange conditions, stimuli that are different from previous ones are generally more powerful than repetitions of previously used stimuli [Dill00]. Each e-mail message sent within the scope of our study differed from the previous messages and conveyed a sense of appropriate renewal of the content communicated. Besides the standardized and highly automated correspondence, candidates were invited to contact the survey administrator directly by e-mail or phone if any questions arose.

According to the TDM, just as with content, the timing of the multiple contacts is also an important aspect of the survey implementation. Our study, which was conducted in the first quarter of 2007, consisted of four phases. A Gantt chart showing the survey milestones is displayed in Figure 19. All phases of this study began at 00:00 GMT¹ and ended at 23:59 GMT of the indicated day.













Task	Date(s)	January	February	March
Pre-Registration	Jan. 15 – Jan. 21			
Invitation E-Mail	Jan. 15			
First Survey Wave	Jan. 22 – Feb. 7			
Notification E-Mail	Jan. 22			
First Reminder E-Mail	Jan. 31			
Second Reminder E-Mail	Feb. 4			
Second Survey Wave	Feb. 19 – Mar. 12			
Notification E-Mail	Feb. 19			
First Reminder E-Mail	Feb. 26			
Second Reminder E-Mail	Mar. 3			
Feedback	Mar. 14 – Mar. 21			
Notification E-Mail	Mar. 14			

Figure 19 Gantt chart of the survey.

¹ GMT (Greenwich Mean Time) refers to mean solar time at the Royal Observatory in Greenwich, England.

Owing to their importance to the study, factors such as response-friendliness of the questionnaires, number of contacts, personalization and content of the correspondence, and others are elaborated in more detail in the following sections. All sample e-mail messages were personalized to a fictive person named *John Hancock*.

In section 5.2.3.1, the course of the pre-registration phase is described. Sections 5.2.3.2 and 5.2.3.3 outline the two survey rounds in detail. Finally, in section 5.2.3.4, we explain the feedback phase.

5.2.3.1 Pre-Registration

The pre-registration phase began on January 15 and ended on January 21. In this phase, the sampled candidates were contacted by e-mail and asked to register online. The pre-registration phase was particularly useful for estimating the candidates' willingness to participate and plan the nomination of further experts.

Considerable research by Dillman, Clark and Sinclair suggests that a notice before the start of a survey is an effective stimulus that significantly reduces nonresponse [DiCS95]. From a social exchange perspective, the invitation e-mail provided an opportunity to send a message to the candidates that was shaped to build interest and anticipation and thereby influence the balance of rewards and costs. The selected candidates were contacted by e-mail on January 15 and 16. The invitation e-mail is shown in Figure 20.

The e-mail message contained, besides some general information about the study, the user ID and e-mail address needed to activate the individual user account. We stated explicitly what was happening, why it was useful, what was expected from participating experts and, finally, what incentives we were providing. We further included trust-inducing elements such as the name of the supporting university, a personalized salutation, the signature of the responsible human administrator, a declaration of confidentiality and our willingness to answer questions. The main goals besides building trust were to convey the idea that the study was something important and that the experts had been carefully selected. To maximize user-friendliness, a hyperlink was included that forwarded directly to the individual registration page. As incentive, an exclusive executive summary of the survey results was promised. However, Dillman shows in [Dill00] that promised incentives do not have nearly so great effect as the precedent provision of incentives, even if they are financial.

<p>Subject: Scientific Study</p> <p>Dear Mr. Hancock,</p> <p>The University of Innsbruck is conducting a large-scale Delphi study in order to assess the potential of Semantic Web services (SWS) as basis for an integration architecture to enable organizations to link their data processing systems more efficiently. By combining the opinions of carefully selected experts from both academia and industry, we hope to come to an understanding of the relevance and applicability of SWS, which will help to align future research efforts more effectively with industry needs.</p> <p>We would like to ask for your help in answering a series of short questionnaires within the next few weeks. The questionnaires can be completed easily in a reasonable time; nevertheless, the Web-based system allows answering the questionnaires in multiple sessions. The first round of the survey begins on January 22, 2007.</p> <p>To participate, please complete the registration form available at http://delphi.bachlechner.info/login.php?uid=123&email=john@hancock.com. Alternatively, you can login manually at http://delphi.bachlechner.info by providing the data below.</p> <p>---</p> <p>User ID: 123 E-Mail: john@hancock.com ---</p> <p>Registration takes less than a minute and does not obligate you to any further cooperation. It should be completed by January 21, 2007 preferably; however, late registrations during the first round of the study are acceptable. Your contribution to this study as a leading expert in one of the related fields would be greatly appreciated.</p> <p>A final report summarizing the findings of this scientific study will be made available to contributing experts prior to publication. Your participation would thus give you privileged access to all results.</p> <p>Should you have any questions, please feel free to contact me at +43-699-11202425, or by e-mail at mailto:daniel.bachlechner@uibk.ac.at.</p> <p>Sincerely, Daniel Bachlechner University of Innsbruck, Austria</p>
--

Figure 20 Invitation.

A flowchart depicting the pre-registration process in detail is shown in Figure 21.

Experts who provided valid login data but had not activated their user accounts before could register for the survey by completing a short form on the survey website concerning their expertise in the research area and their professional background. The candidates could also change their contact data, which consisted of forename, surname, gender and e-mail address.

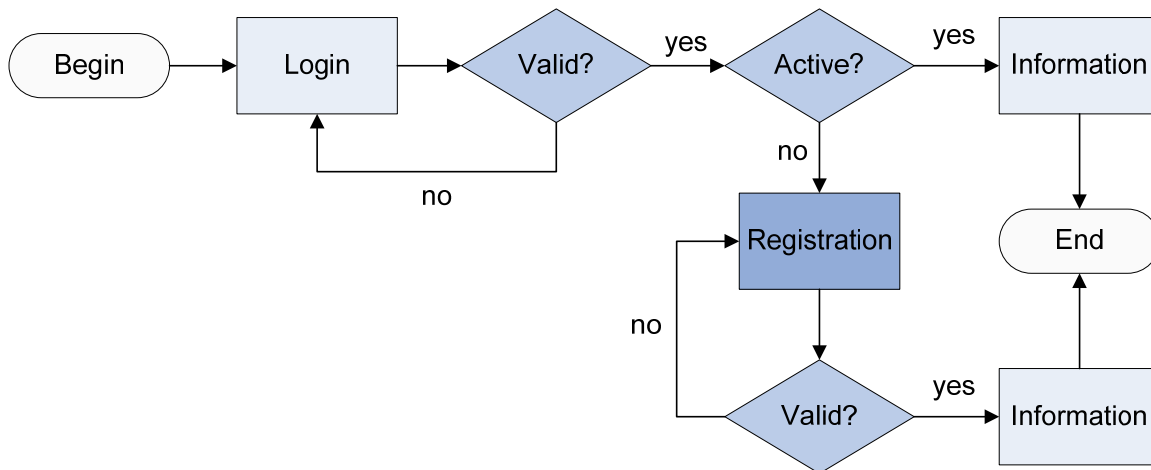


Figure 21 Pre-registration process.

If the registration data was invalid for some reason, an error message was shown and the candidate was asked to complete the fields properly. If the registration data met the requirements, an informative message was shown and a confirmation e-mail sent to the expert and to the survey administrator. The confirmation e-mail is shown in Figure 22.

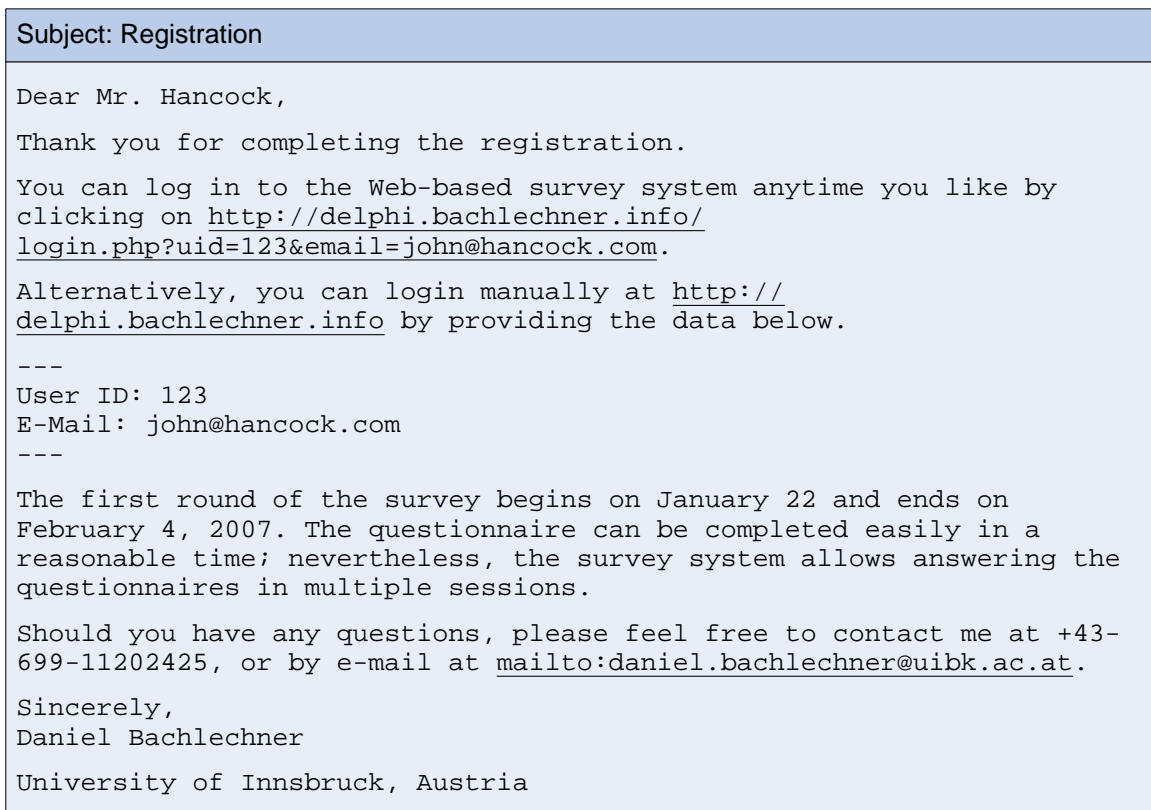


Figure 22 Registration confirmation.

The end date in the confirmation e-mail was changed from February 4 to February 7 after the deadline was extended. The e-mail message was kept as short as possible and covered only highly relevant information.

Candidates who provided valid login data but had already activated their accounts were shown informative messages also indicating the commencement date of the first round of the survey. If the login data was invalid, an error message was shown and a candidate could retry to log in. The number of tries was not limited.

5.2.3.2 Qualitative Survey

The first round of the survey began on January 22 and ended on February 7. During this phase, the experts were asked to complete a qualitative questionnaire containing the 15 questions introduced in section 5.2.1. It was still possible for candidates to complete the registration during this phase. The registered candidates were notified by e-mail about the beginning of the survey on January 22. The notification e-mail is shown in Figure 23.

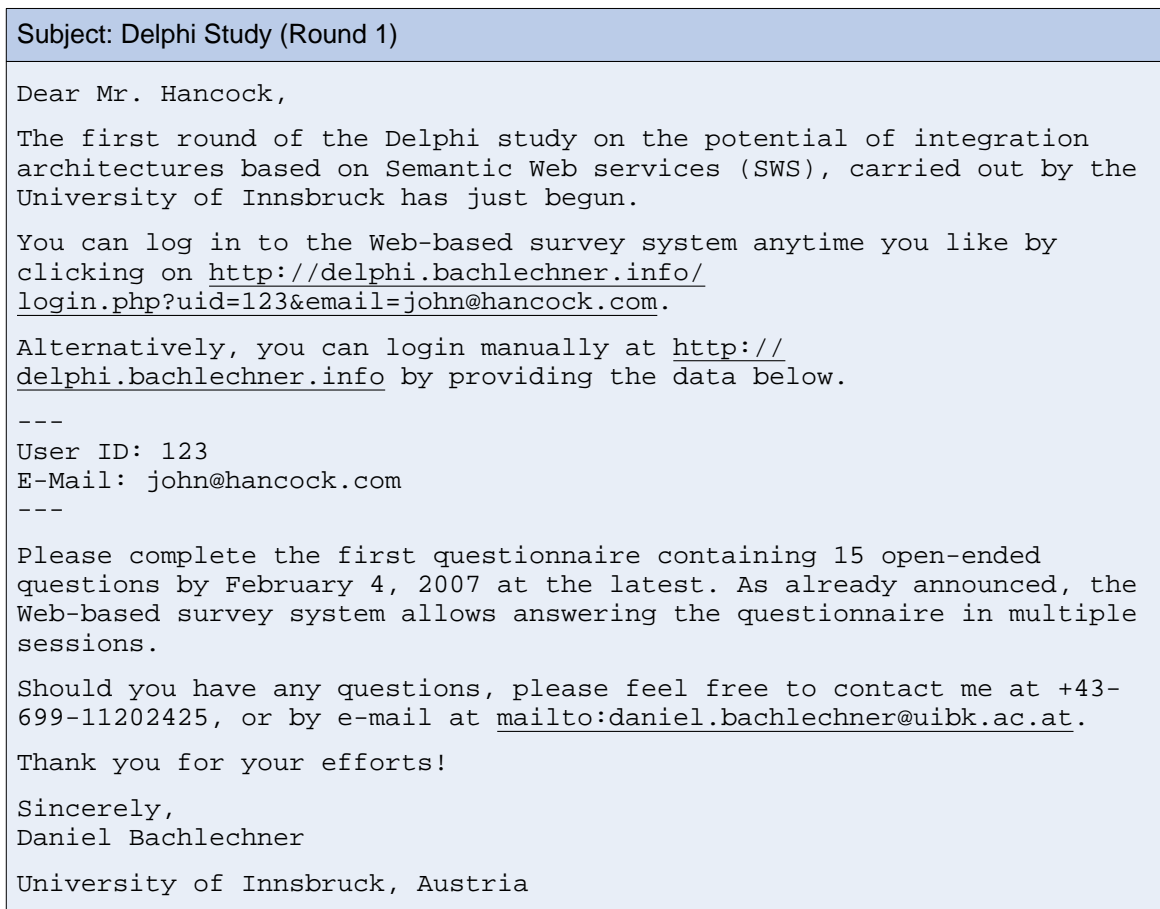


Figure 23 Notification about the beginning of the first survey round.

Also on January 22, a reminder e-mail was sent to all candidates who had not yet registered or explicitly stated that they did not intend to participate. As per Dillman, response rates are usually between 20% and 40% lower than those typically attained if there are no follow-up contacts [Dill00]. This fact makes a carefully designed follow-up sequence imperative, in particular within the scope of surveys such as Delphi studies which have several rounds.

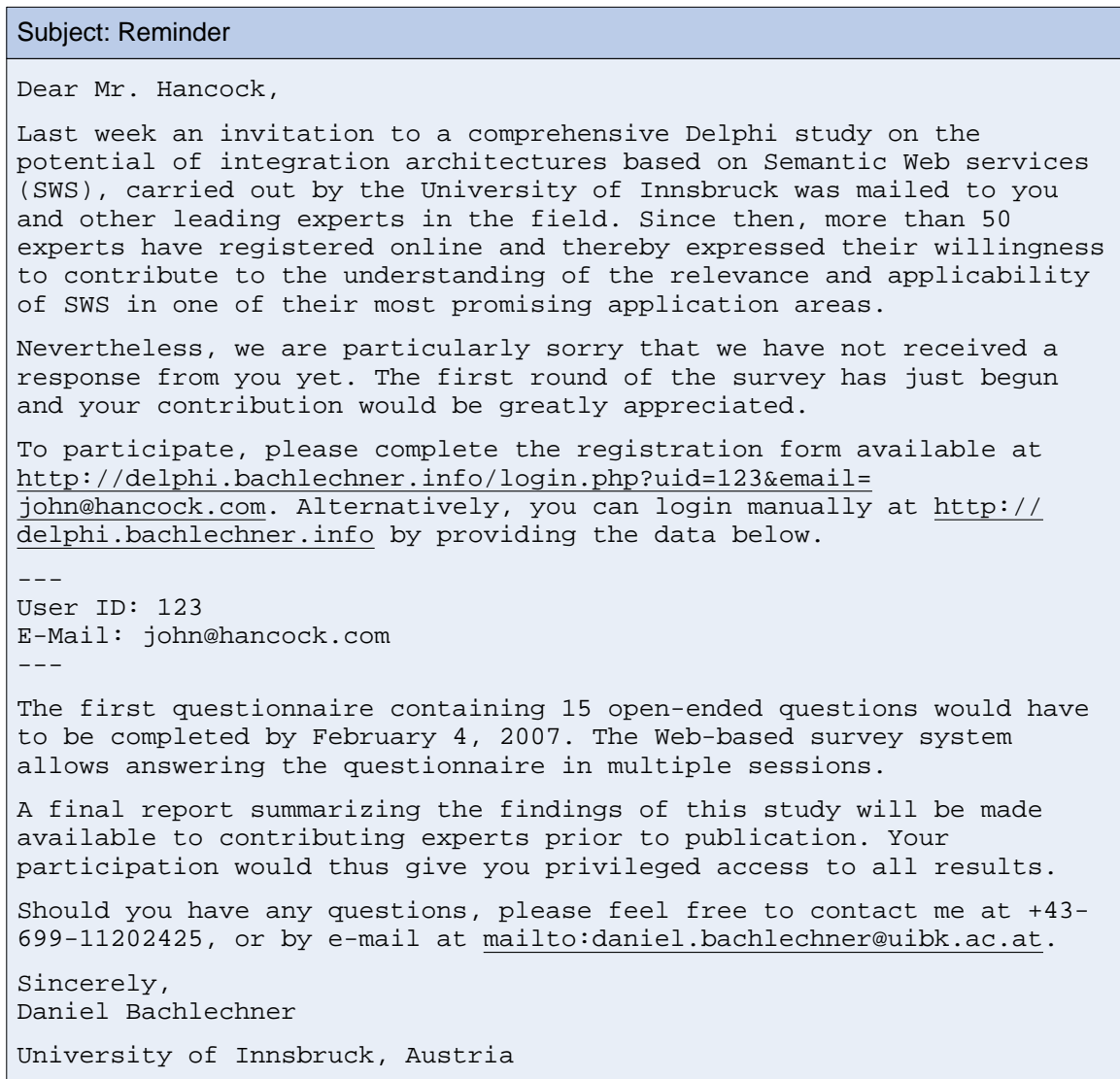


Figure 24 First general reminder.

We attempted to improve response rates in a way that did not in any from offend the candidates. Each e-mail message provided a new opportunity to appeal for participation by using a slightly different approach. For instance, in the first e-mail of the follow-up we stated that more than 50 experts had already registered and expressed our particular

disappointment that the respective candidate had not responded. The first reminder e-mail is shown in Figure 24.

If an expert provided valid login data but the user account had not been activated yet, the registration process was very much the same as in the pre-registration phase. The only difference was that the candidates were forwarded to the questionnaire directly after completing the registration. Candidates who provided valid login data and had already activated their accounts were forwarded to the questionnaire. A flowchart depicting the first round of the survey in detail is shown in Figure 25.

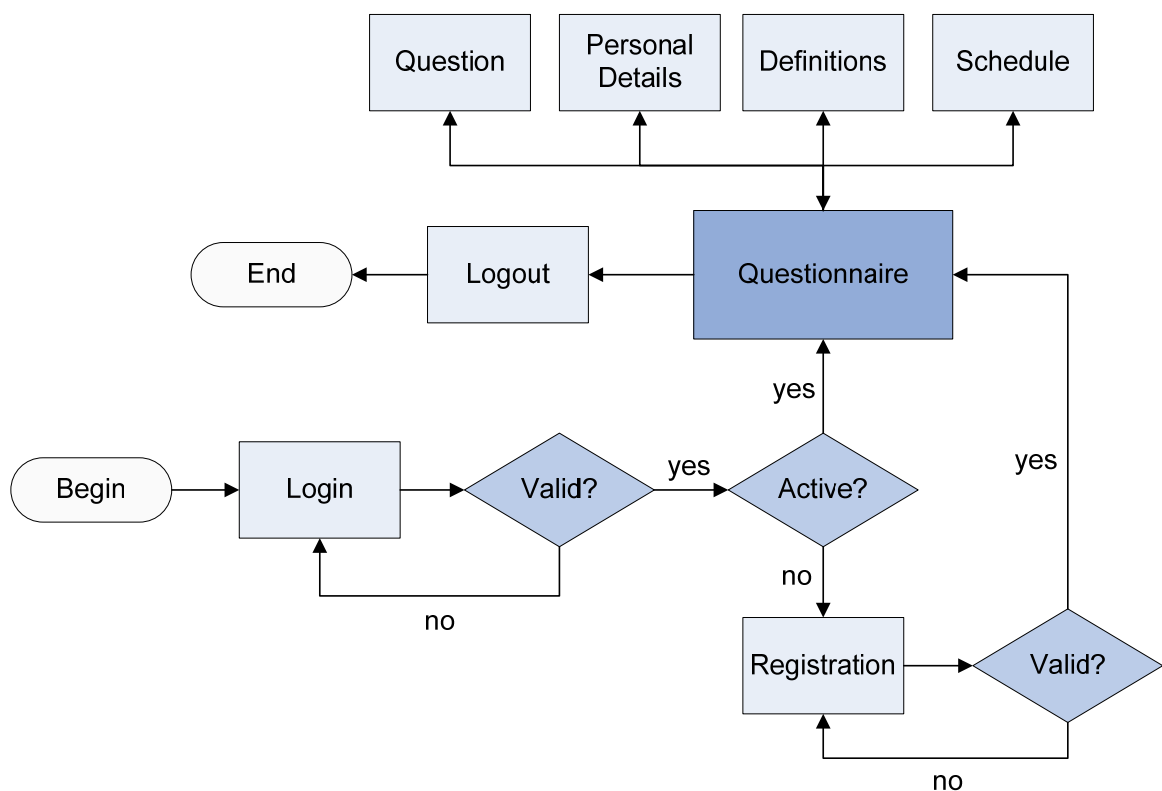


Figure 25 First round of the survey.

The qualitative questionnaire consisted of 15 open-ended questions. The participating experts could structure their responses by using separate text areas. They could enter as much text as they liked in each area. There was also a *No Comment* checkbox provided if a participant did not want to answer a question. Graphical symbols were used to convey a sense of where the respondent was in the completion process. After either answering a question or checking the *No Comment* box, the question was checked off the list.

The structure of the system did not require respondents to provide an answer to every single question before being allowed to answer any subsequent ones. Instructions on how

to take necessary actions when working with the survey system were provided online at the points where they were needed, in the e-mail messages and individually, if requested.

Experts could also change some of their personal details during the first survey round. Only contact data such as forename, surname, gender and e-mail address could not be changed anymore at this point. Some important definitions and the survey schedule were available on separate pages.

After logging out, candidates were shown a message indicating the number of questions that had been answered so far. During this phase, experts could login as many times as they wanted and continue answering questions.

On January 31, a first reminder e-mail was sent to registered experts who had not answered all questions by that time. The reminder e-mail sent to experts who had not answered any question is shown in Figure 26 and the e-mail sent to experts who had answered part of the questionnaire is shown in Figure 27.

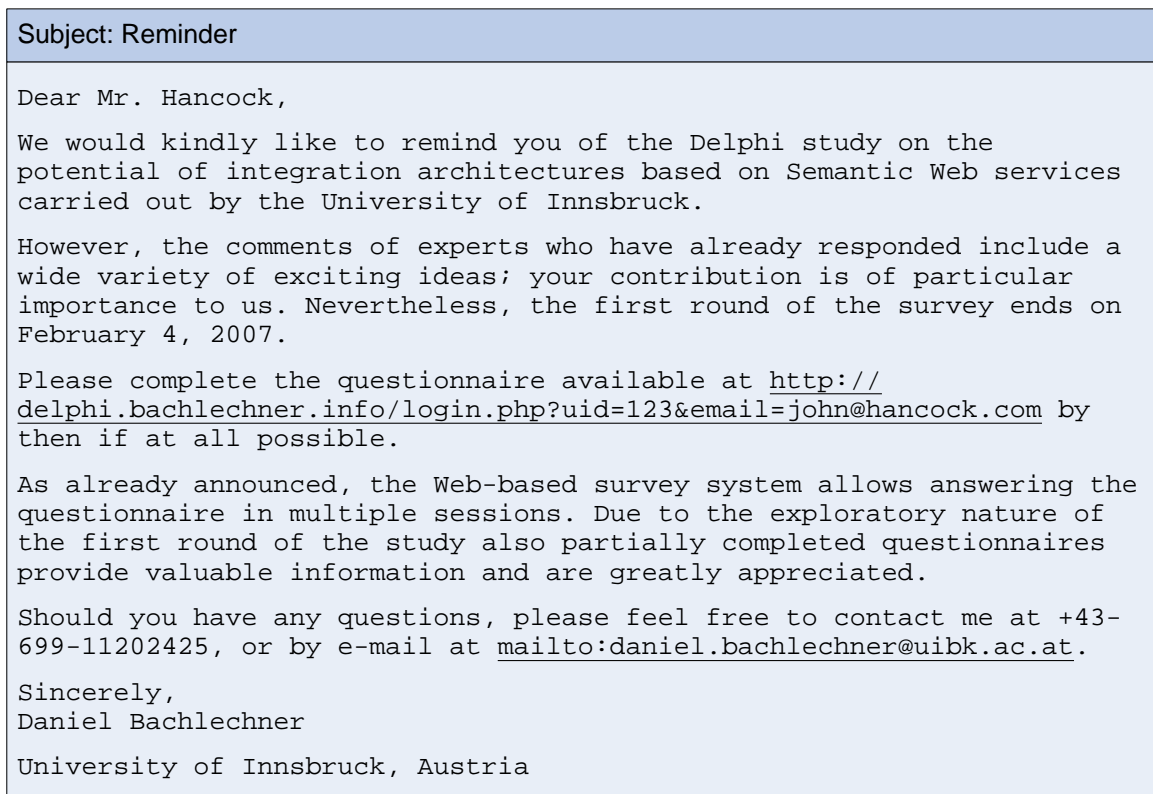


Figure 26 Reminder for participants who had not answered any questions.

Most experts who answer questionnaires do so almost immediately after they receive them. A questionnaire that remains unanswered for more than a few days is unlikely to be

answered at all [Dill00]. The phenomenon described by Dillman was also observed within the scope of our study. After three days, the number of responses declined sharply at first, then gradually. Hence, a period of one week seemed to be an appropriate interval before making another appeal to convey a sense of importance. After this amount of time a reminder does not sound impatient or unreasonable.

The reminder e-mail sent to registered experts who had not answered any question stated explicitly that time was running out and, as was done before, that others had responded. Because the survey looks quite long at first glance, we made it clear that even partly completed questionnaires were appreciated. The reminder sent to registered experts who had answered part of the questions also showed the number of questions answered and the total number of questions to further motivate the respondent to complete the remaining questions.

<p>Subject: Reminder</p> <p>Dear Mr. Hancock,</p> <p>The comments of you and other experts who have already responded to the first questionnaire of our Delphi study include a wide variety of exciting ideas. You have answered 7 of 15 questions so far.</p> <p>As announced, the first round of the survey ends on February 4, 2007. You still have the chance to edit and complete your answers by then.</p> <p>To log in to the survey system, please click on http://delphi.bachlechner.info/login.php?uid=123&email=john@hancock.com.</p> <p>Due to the exploratory nature of the first round of the study also partially completed questionnaires provide valuable information and are greatly appreciated.</p> <p>Should you have any questions, please feel free to contact me at +43-699-11202425, or by e-mail at mailto:daniel.bachlechner@uibk.ac.at.</p> <p>Thank you for your efforts!</p> <p>Sincerely, Daniel Bachlechner University of Innsbruck, Austria</p>
--

Figure 27 Reminder for participants who had partially answered the questionnaire.

Also on January 31, a final reminder e-mail was sent to all candidates who had not yet registered or explicitly stated that they did not intend to participate at that point. The final reminder e-mail is shown in Figure 28.

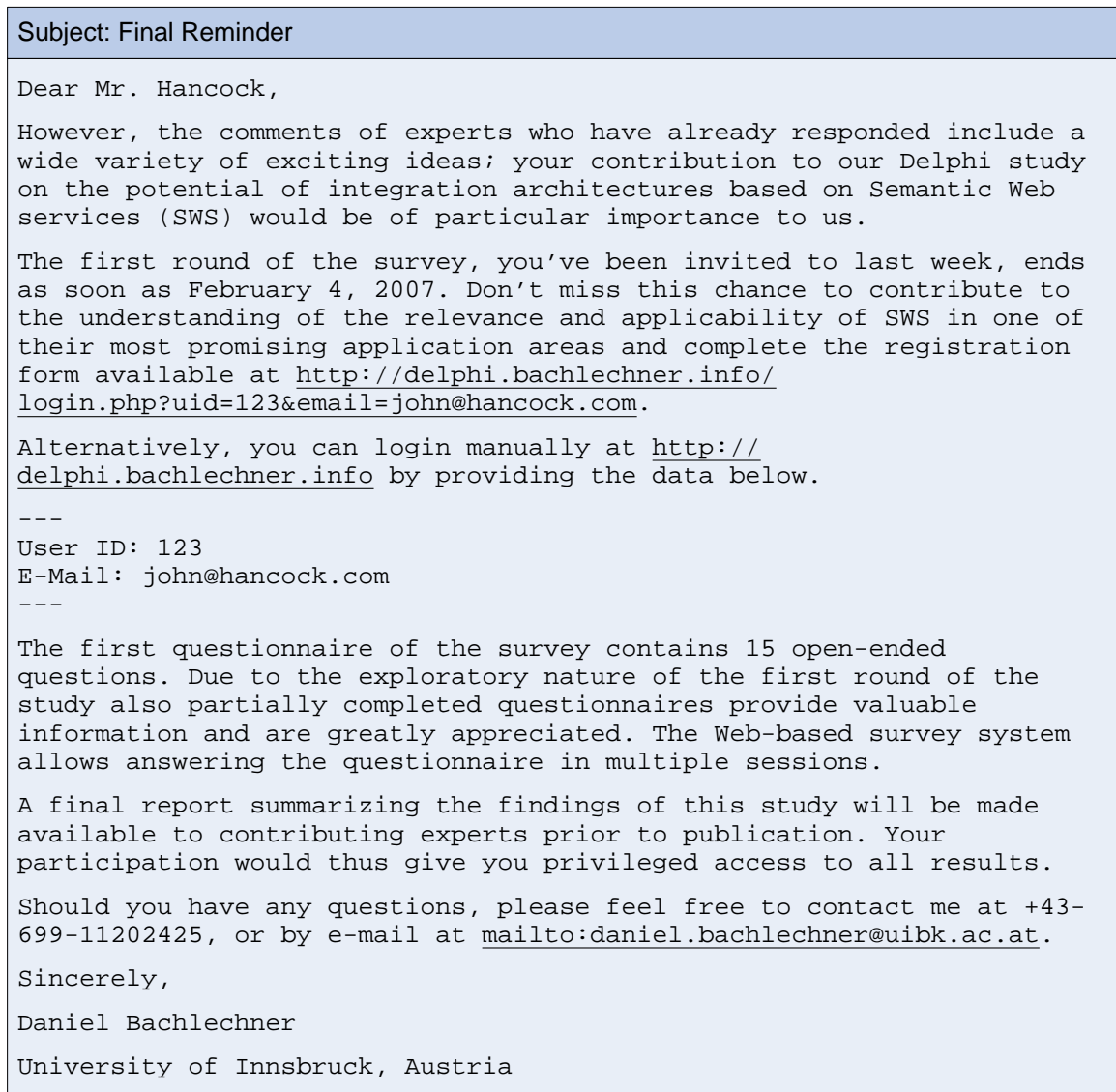


Figure 28 Second general reminder.

On February 4, a final reminder e-mail, shown in Figure 29, was sent to the registered experts. This final reminder e-mail also announced the deadline extension.

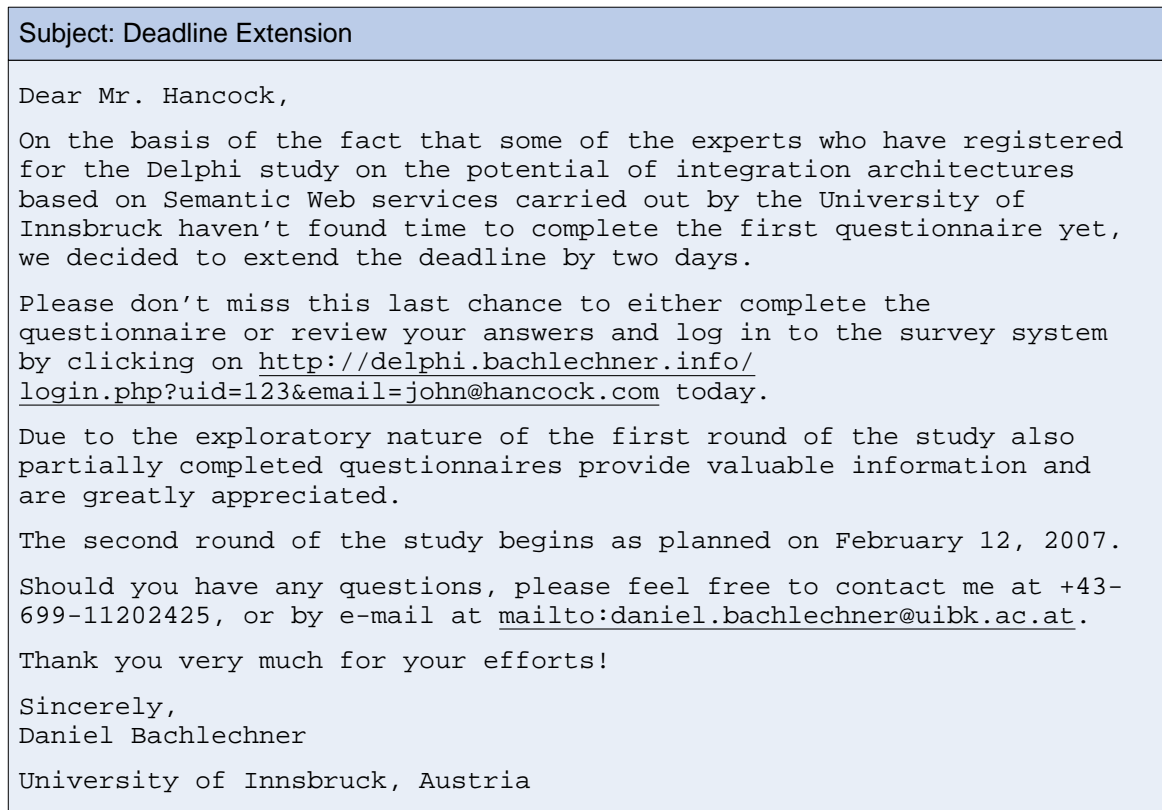


Figure 29 Second reminder for study participants.

5.2.3.3 Quantitative Questionnaire

The second round of the survey began slightly behind schedule on February 19 and ended on March 12. In this phase, the experts were asked to complete a quantitative questionnaire consisting of 14 topics with up to 40 statements each. The registered candidates who participated in the first round were notified by e-mail on February 19 about the beginning of the second round of the survey. The notification e-mail is shown in Figure 30.

This was the first e-mail sent to the participants in a more personal style. Instead of title and surname the candidate's forename was used in the salutation. This was done in response to the vast majority of e-mails received by the survey administrator being written in a similar, rather personal style.

It was not possible anymore to register during this phase. If experts had not activated their user account before the end of the first round of the survey, they could not participate at the Delphi study. Their login attempts were handled as if they were invalid. A flowchart depicting the second round of the survey in detail is shown in Figure 31.

Subject: Delphi Study (Round 2)

Dear John,

We are very grateful for your valuable participation in our Delphi study on the potential of integration architectures based on Semantic Web services (SWS) so far.

Your contribution to the second round of our study would be greatly appreciated now.

Please log in to the Web-based survey system by clicking on <http://delphi.bachlechner.info/login.php?uid=123&email=john@hancock.com>, and complete the second questionnaire by March 4, 2007, which consists of several rating scales formulated from the responses from the first round.

Alternatively, you can login manually at <http://delphi.bachlechner.info> by providing the data below.

User ID: 123
E-Mail: john@hancock.com

As before, the Web-based survey system again allows you to answer the questionnaire in multiple sessions.

Should you have any questions, please feel free to contact me at +43-699-11202425, or by e-mail at <mailto:daniel.bachlechner@uibk.ac.at>.

Thank you for your efforts!

Best regards,
Daniel Bachlechner
University of Innsbruck, Austria

Figure 30 Notification about the beginning of the second survey round.

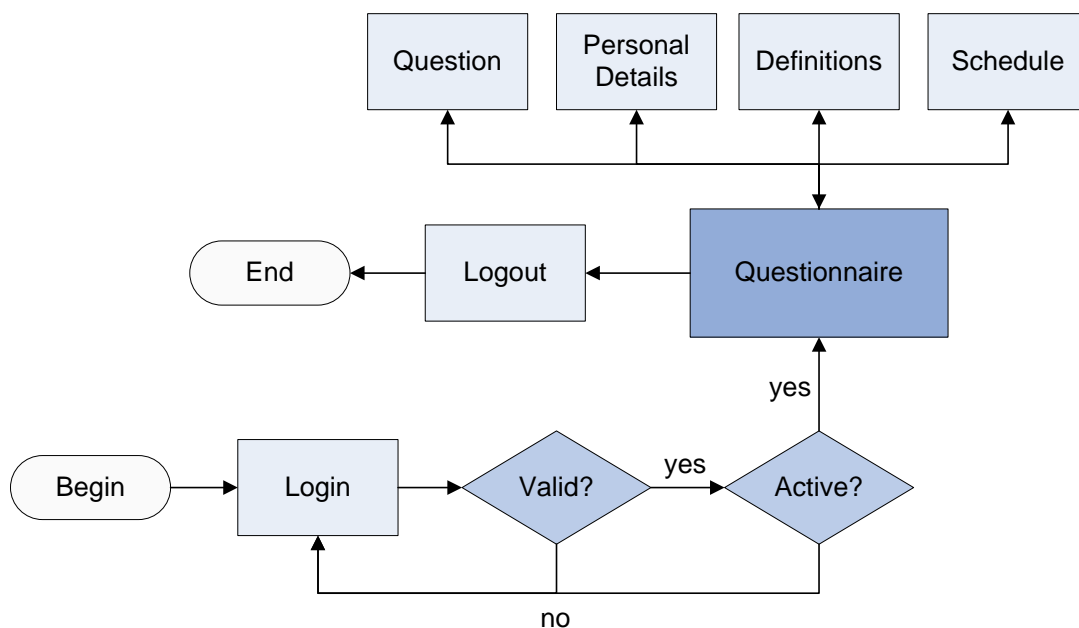


Figure 31 Second round of the survey.

As mentioned previously, the second questionnaire consisted of 14 topics with up to 40 statements each. The participating experts were asked to rate how strongly they agreed or disagreed with each statement. The order of the statements was randomized to avoid order effects. Dillman describes in [Dill00] five distinct situations in which answers to subsequent questions are altered dramatically because of the question immediately prior. To preserve clarity, statements that had already been rated were always placed at the top. There was also a text area provided for every topic to add comments on existing or missing statements or on a topic in general. The experts could enter as much text as they liked in each area and were asked to indicate a statement's numeric code if they commented on an existing statement. For every topic the percentage of statements that had already been rated was indicated graphically.

<p>Subject: Reminder</p> <p>Dear John,</p> <p>We would kindly like to remind you of the Delphi study carried out by the University of Innsbruck.</p> <p>As may be imagined, the comments received in the first round were related to a wide variety of aspects and exhibited different degrees of abstraction. That made it difficult to completely avoid a certain amount of ambiguity and vagueness when formulating a limited set of short statements to be rated. Nevertheless, we are confident that - with your continued support - the results will help us to come to a better understanding of the relevance and applicability of integration architectures based on Semantic Web services.</p> <p>Your contribution is of particular importance for the success of our study. Please complete the questionnaire available at http://delphi.bachlechner.info/login.php?uid=123&email=john@hancock.com by March 4, 2007 if at all possible.</p> <p>As already announced, the Web-based survey system allows answering the questionnaire in multiple sessions. Also partially completed questionnaires provide valuable information and are greatly appreciated.</p> <p>Should you have any questions, please feel free to contact me at +43-699-11202425, or by e-mail at mailto:daniel.bachlechner@uibk.ac.at.</p> <p>Thank you for your efforts!</p> <p>Sincere regards, Daniel Bachlechner University of Innsbruck, Austria</p>
--

Figure 32 First reminder of the second survey round.

As in the previous phase, experts could again change their personal details and view important definitions as well as the up-to-date survey schedule during the second survey

round. Also during this phase, experts could login as many times as they wanted and continue rating the statements.

On February 26, a first reminder e-mail was sent to the registered experts who had not rated all statements at that time. The reminder e-mail is shown in Figure 32.

Because some candidates criticized the ambiguity and vagueness of selected statements, we also used this e-mail to explain that the comments received in the first round were related to a wide variety of aspects and exhibited different degrees of abstraction that made formulating a limited set of short and unambiguous statements very difficult. Comments related to the ambiguity and vagueness of selected statements and other topics are summarized in section 5.5.

On March 3, a final reminder e-mail was sent to the registered experts. The final reminder e-mail is shown in Figure 33.

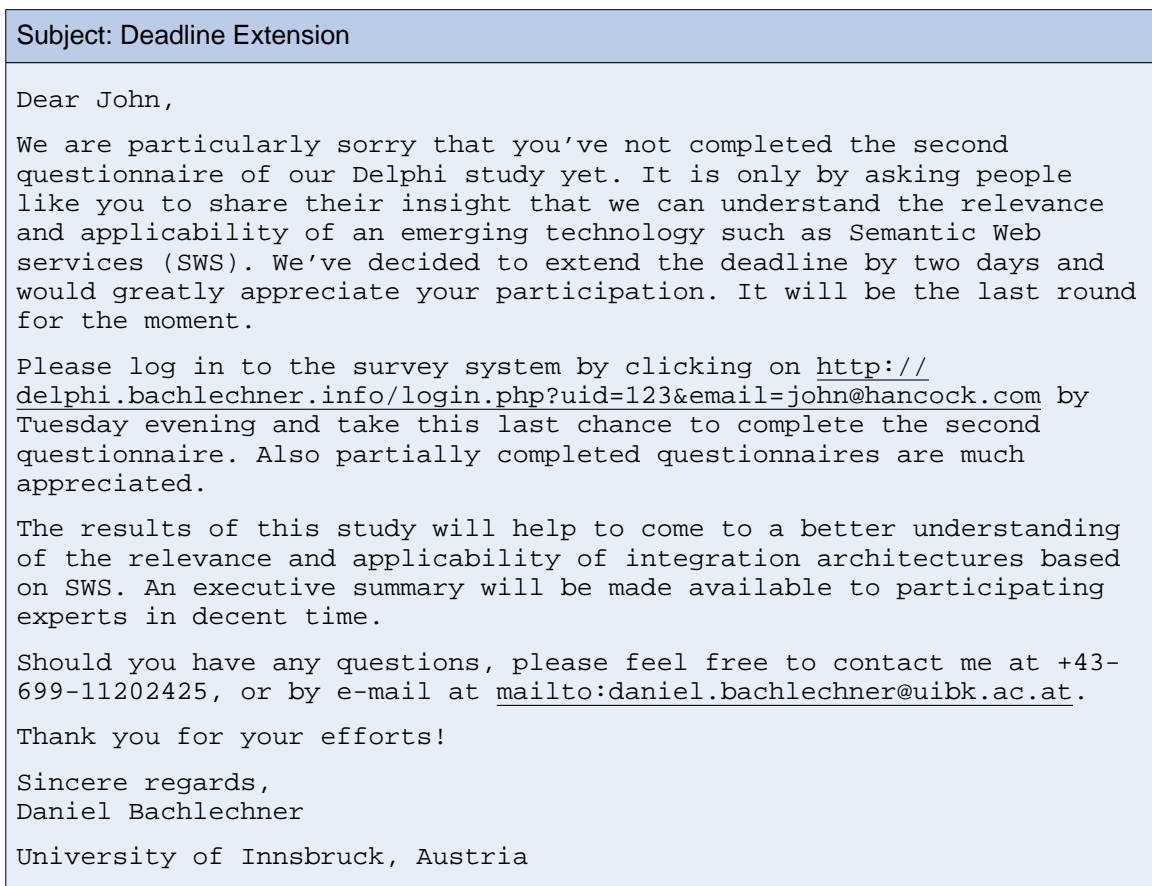


Figure 33 Second reminder of the second survey round.

The final reminder also announced the extension of the deadline. Furthermore, we announced that it was the last round of the survey for the moment.

5.2.3.4 Feedback

The feedback phase began on March 14 and ended on March 21. In this phase, the experts who contributed to the study were asked to provide feedback on the survey process in general and on the functionality and usability of the Web-based survey system in particular. The experts were notified about the beginning of the feedback phase by e-mail on March 14. The notification e-mail is shown in Figure 34.

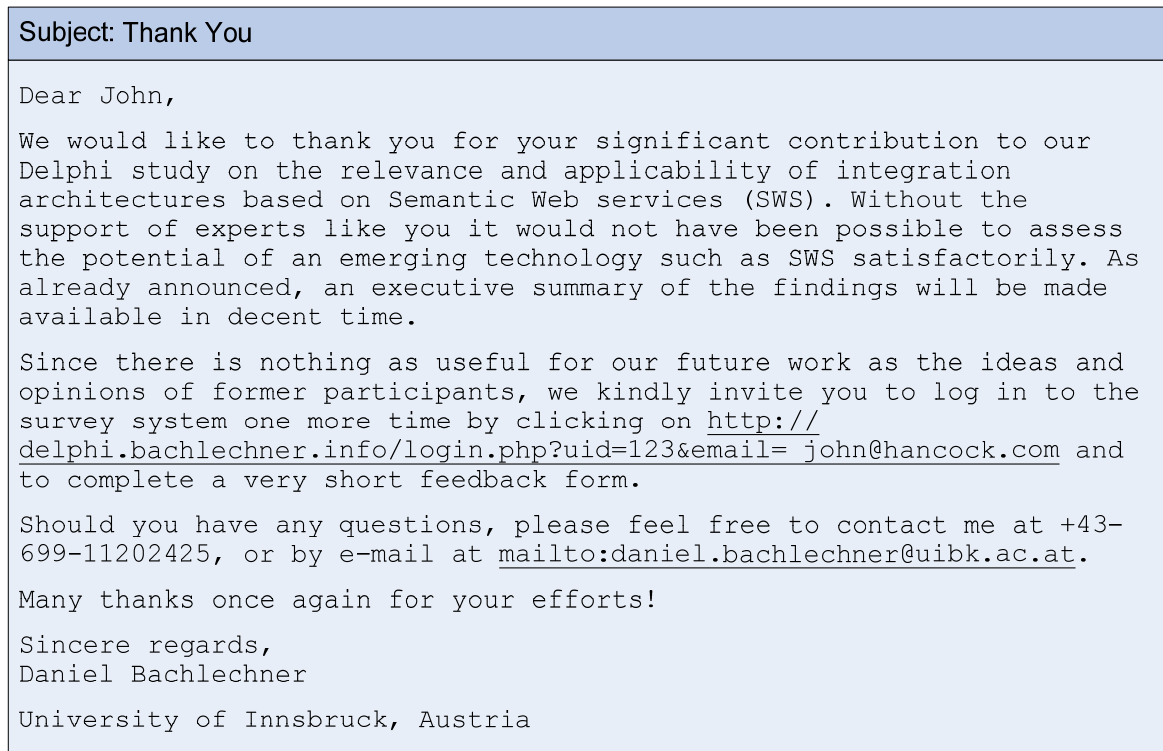


Figure 34 Notification about the beginning of the feedback phase.

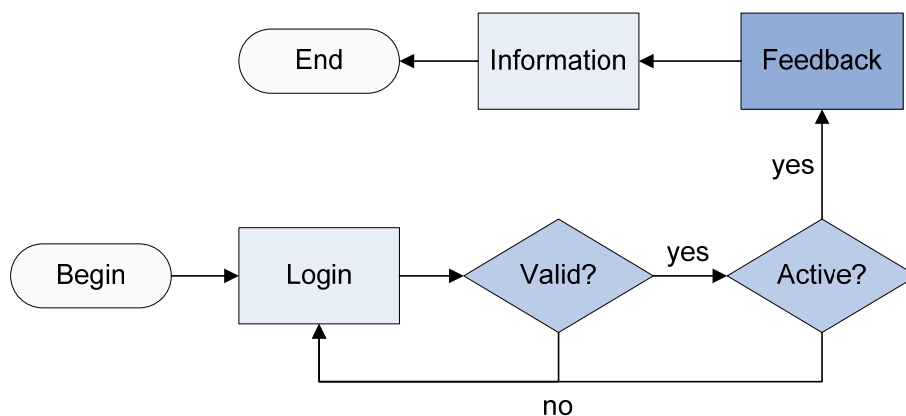


Figure 35 Feedback process.

This e-mail message was also used to thank all participants for their efforts. Only experts who participated at the survey could give feedback. Login attempts from others were handled as if they were invalid. A flowchart depicting the feedback phase is shown in Figure 35.

The feedback consisted of ratings of the perceived functionality and usability of the Web-based survey system as well as a text area used for free-text comments on the survey in general. According to the standard ISO 9126¹, which is primarily concerned with the definition of quality characteristics to be used in the evaluation of software products, functionality and usability are major criteria for the assessment of software quality among reliability, efficiency, maintainability and portability. The standard defines functionality as the existence of a set of functions satisfying stated or implied needs and their specified properties. Usability is defined as the effort needed to use a system and the individual assessment of such use. Participants were only asked to comment on functionality and usability because the others are difficult to measure from a user perspective. After submitting the feedback, a confirmation message was shown. All form fields in the feedback phase were optional.

5.2.4 Technical Realization

In this section, the technical details of the Web-based survey system implemented for the study are outlined. The survey system and in particular the two questionnaires were constructed in a fixed format with the goal of making them appear the same for all respondents. However, for reasons discussed in this section, this goal was not always achieved easily.

The survey was run on a Linux system with an Apache Web server and a MySQL database server. The survey system was available online at <http://delphi.bachlechner.info/> nonstop from January 15 until May 15, 2007.

Many different programming languages and styles can be used to build Web-based survey systems, some of which are quite sophisticated. We used the reflective programming language PHP (PHP: Hypertext Preprocessor) for server-side scripting. With regard to cross-browser compatibility, the use of JavaScript or any other client-side script language was avoided. While the use of client-side technologies such as JavaScript makes it possible

¹ For more information on ISO 9126, see <http://www.iso.org/>.

to significantly improve the functionality and usability of a survey system, doing so also implies difficulties for people with older, less powerful systems and in particular older Web browsers to respond to such surveys.

Differences in the visual appearance of the survey system resulting from different screen configurations, operating systems and browsers were avoided, if possible. The survey system was designed to be displayed identically on all systems with a display resolution of at least 800 x 600 pixels. Displays with fewer pixels, which are extremely rare for desktop and laptop computers, could also be used to display the survey, however, horizontal scrolling was inevitable. The screenshots displayed in this section were taken with a display resolution of 1024 x 786 pixels and a full-screen Web browser window. Furthermore, all pages of the system were made fully XHTML¹ (Extensible Hypertext Markup Language) compliant. The survey system was tested extensively with the most popular Web browsers before going live.

The stylesheet language CSS (Cascading Style Sheet) was used, in addition to XHTML, to describe the presentation of the pages. The combination of XHTML and CSS allows the use of advanced design features such as split screens, embedded programs, animations and sound tracks. However, as Web-based surveys become more complex, the data files exchanged between clients and servers increase in size. The more features used, the greater the likelihood that some people will receive the pages slowly or not at all. Because of considerable heterogeneity in browser capabilities and line transmission speeds available to respondents, the gains in creativity and advanced programming had to be balanced against the costs of making it impossible for some to access and respond. According to Dillman, nonresponse in Web-based surveys is also likely to be a result of ignoring or misjudging compatibility issues [Dill00].

The header at the top of all pages was kept reasonably small to limit the waste of vertical screen space. Contact information was provided at the bottom of all pages. Larger fonts and spacing were used to clearly structure the page contents and to indicate where a user should start reading each screen. The use of color was restrained to maintain ground consistency as well as readability and unimpeded navigational flow. As per Dillman, the inappropriate use of color represents one of the biggest threats to the development of good

¹ XHTML is a markup language that has the same depth of expression as HTML but also conforms to XML syntax.

Web-based surveys [Dill00]. We used black letters on neutral backgrounds exclusively for our survey system.

A lack of equipment or aptitude was not expected to be an impediment to using a Web-based survey to get responses from the defined target group. The experts of interest generally have Internet access. Differences that may exist in the capabilities of candidates' computers and software were considered within the scope of the surveys' design and implementation.

In section 5.2.4.1, we describe the function of the files used to login to the survey system. In section 5.2.4.2, the files required to register and to change personal information, respectively, are outlined. In section 5.2.4.3, we introduce the files relevant for the actual questionnaires, and in section 5.2.4.4, we describe the files of the feedback phase. Finally, in section 5.2.4.5, the function of the logout files is sketched. Sample screens were personalized to the fictive person introduced in the previous section.

5.2.4.1 Login

The *index* and the *login* file played a major role in all four phases of the survey process. In all phases the *index* file sent a specific HTTP header to the clients to redirect them to the *login* file. This approach enabled automatically redirecting users entering the system via <http://delphi.bachlechner.info> to the login page. The login page is shown in Figure 36. The purpose of the *login* file was to collect login data and pass it to the next file.

The data was either sent to the login file through the query component¹ of the URI or entered in a form and confirmed by clicking on the *Login* button. The query component was used in case a user clicked on the hyperlinks provided in some e-mail messages. If the parameters were already known the login page was not shown. Depending on the phase of the survey, the data was forwarded to the *registration* file, one of the two *questionnaire* files or the *feedback* file.

¹ The query component of a URI is specified as the part between the question mark and the end of the URI or the number sign.

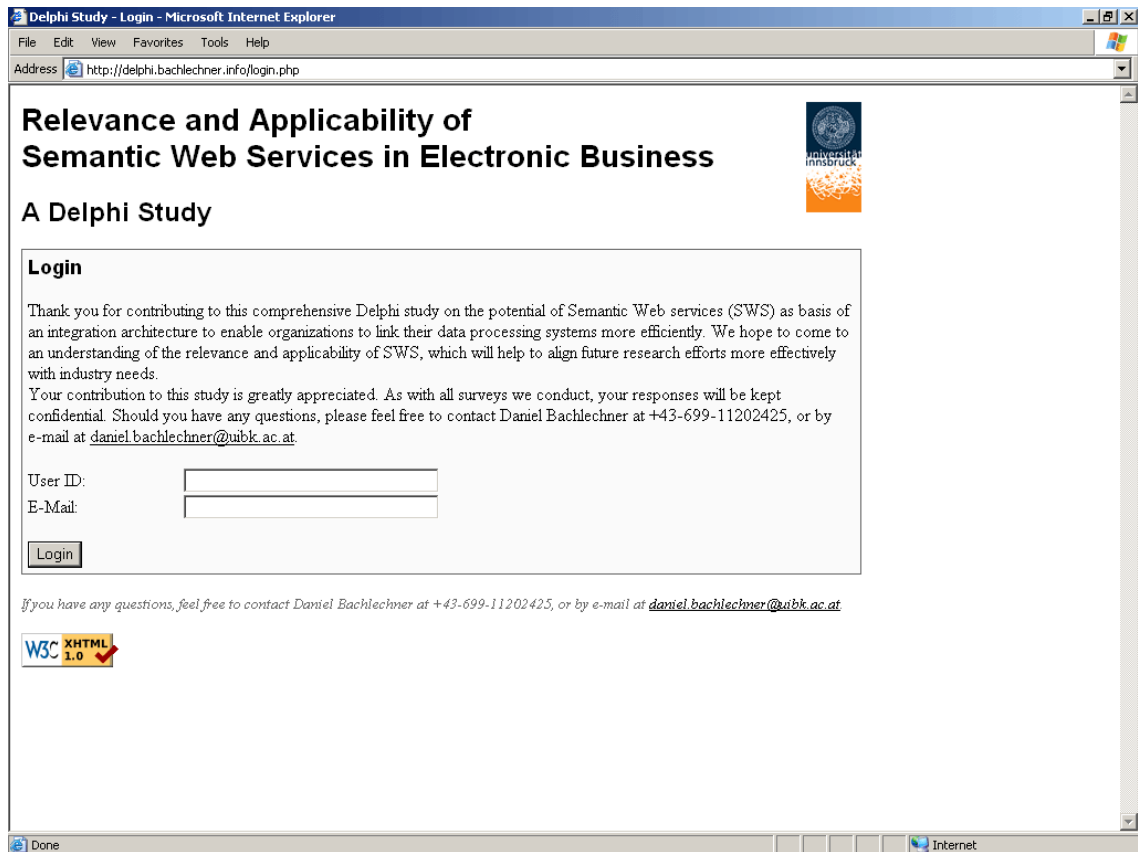


Figure 36 Login page.

During the four phases, the login page introduced the survey with a welcome page that had a primarily motivational character. Before the study began, in between the phases and after the study ended the login page had a different look; there was only a short message indicating what was to happen next.

One of the parameters encoded in the query component of the URI was a personal identification number (PIN), which was used to limit access to people in the sample. Using a PIN was necessary because the survey was conducted to investigate the opinions of a population of carefully selected experts. Otherwise, anyone who knew the website address could have accessed and answered the questionnaire. The randomly assigned PINs (numbers between 1 and 500) were part of the invitation e-mail sent to the candidates. The other field was the e-mail address of the sampled individual. The second parameter encoded in the query component of the URI was chosen to hinder people from guessing a PIN, even members of the sample. Because the survey system allowed experts to respond to the questionnaires in multiple sessions, an efficient identification was particularly important.

The validity of provided combinations of user IDs and e-mail addresses was not checked by the *login* file. This task was left to the subsequent files. If a combination passed was invalid the client was redirected back to the *login* file and a specific parameter was set indicating the error. If this parameter was set an error message was shown on the login page. Whether the login data was sent through the query component of the URI or typed into the form fields on the login page did not make a difference.

5.2.4.2 Registration and Personal Details

The *registration* file was the most important file for the registration of sampled individuals and for making changes to personal information during the first and second round of the survey. The files involved in the pre-registration phase are illustrated in Figure 37. The files involved in the two survey rounds are displayed in the next section.

The *registration* file was, besides the two *questionnaire* files and the *feedback* file, one of the files that validated login data. If the file received no login data or an invalid combination of user ID and e-mail address, the client was redirected back to the *login* file and a specific parameter was set in order to let the *login* file know that an error had occurred.

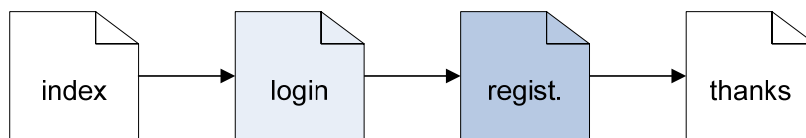


Figure 37 Files involved in the pre-registration phase.

During the pre-registration phase and the first round of the survey, sampled individuals who had not yet registered were directed to the registration page after successfully logging in. A large part of the registration page is shown in Figure 38. Optional information included the specification of the current affiliation, position and country of employment. The user ID could not be changed. The e-mail field and the name fields were preset but could be changed, if required. The collected data was sent to the *thanks* file, described in section 5.2.4.5, to be validated after clicking on the *Register* button.

A Delphi Study

Registration

Identification and Contact

User ID: 123

E-Mail: john@hancock.com

Name: Mr. John Hancock

Info: Please change the value of the e-mail field if you prefer correspondence via a different address. The address will not be published. Completion of the name fields is optional. Your responses will be kept confidential.

Professional Background

Background: ☐ Academia ☐ Industry

Affiliation:

Position:

Country:

Info: Please indicate whether you consider your background to be in academia or industry. Completion of the fields for current affiliation, position, and country of employment is optional.

Expertise

Expertise: ☐ Expert ☐ Proficient ☐ Competent ☐ Beginner ☐ Novice

Info: Please self-assess your expertise in the research area of this study.

If you have any questions, feel free to contact Daniel Bachlechner at +43-699-11202425, or by e-mail at daniel.bachlechner@subk.ac.at

Figure 38 Registration page.

If the dataset passed on was invalid, the client was redirected back to the *registration* file and a specific parameter was set indicating the error. If this parameter was set an error message was shown on the registration page. If a user who had already registered logged in another time during the pre-registration phase, an informative message was shown instead of the registration form.

After the successful completion of a registration process, the collected data was stored in the database and a confirmation e-mail was sent to the user and the survey administrator. During the pre-registration phase, a short message indicated what was to happen next. During the first round of the survey, the user was forwarded directly to the qualitative questionnaire as described in section 5.2.4.3.

During the two rounds of the survey, it was possible for registrants to change some personal details by clicking on *Personal Details* in the menu. The files involved in the first and second rounds of the survey are described, as previously mentioned, in section 5.2.4.3, and the personal details page is displayed Figure 39. Again, the *registration* file was used to create this page but specific parameters had to be set.

Delphi Study - Registration - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <http://delphi.bachlechner.info/registration.php?uid=123&email=john@hancock.com&details=true&referer=questionnaire1>

Relevance and Applicability of Semantic Web Services in Electronic Business

A Delphi Study

Personal Details

Identification and Contact

User ID: 123
 E-Mail: john@hancock.com
 Name: Mr. John Hancock
Info: Your responses will be kept confidential.

Professional Background

Background: ☐ Academia ☒ Industry
 Affiliation:
 Position: Governor
 Country: USA
Info: Please indicate whether you consider your background to be in academia or industry. Completion of the fields for current affiliation, position, and country of employment is optional.

Expertise

Expertise: ☒ Expert ☐ Proficient ☐ Competent ☐ Beginner ☐ Novice
Info: Please self-assess your expertise in the research area of this study.

If you have any questions, feel free to contact Daniel Bachlechner at +43-699-11202425 or by e-mail at daniel.bachlechner@unihk.ac.at

Done Internet

Figure 39 Personal details page.

Within the scope of this view on personal user details, only information related to the professional background and the expertise could be changed. The contact information was fixed after the registration. After clicking on the *Update* button, the data was sent to the respective *questionnaire* file and, subsequently, stored in the database.

5.2.4.3 Questionnaires, Questions and Topics

Web-based surveys can be constructed in a screen-by-screen manner so that each time a question is answered the user is directed to the next question on a new page. Alternatively, they can be constructed in a way that allows respondents to use the scroll bars to go anywhere in the questionnaire at any time [Dill00]. In the first round of our survey, questions were displayed on separate pages. In the second round the topics were presented on separate pages but respondents ranked various sets of statements on a single page. Hence, our survey system formed a hybrid approach.

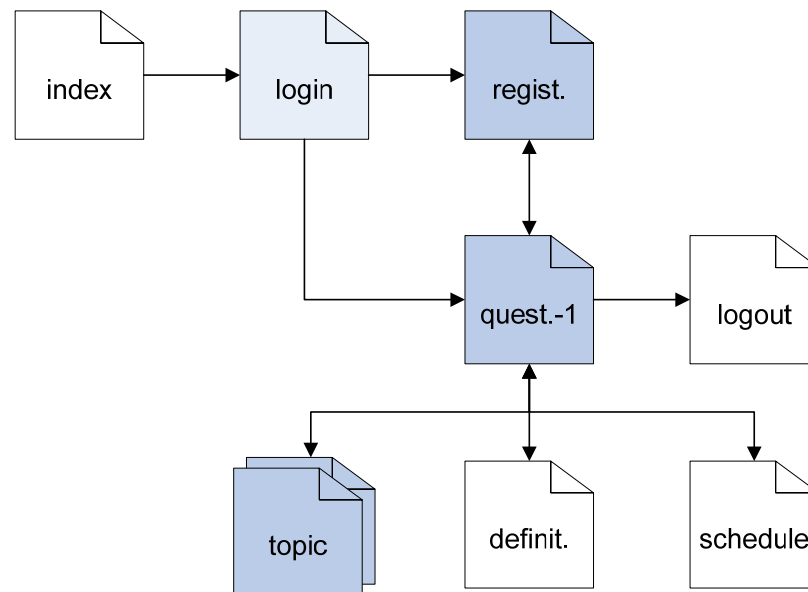


Figure 40 Files involved in the first round of the survey.

The files involved in the first round of the survey are shown in Figure 40 and the qualitative questionnaire itself appears in Figure 41.

Delphi Study - Questionnaire - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <http://delphi.bachlechner.info/questionnaire1.php>

Relevance and Applicability of Semantic Web Services in Electronic Business

A Delphi Study

[Personal Details](#) | [Definitions](#) | [Schedule](#) | [Logout](#)

User	Phase
User ID: 123	Round: 1
E-Mail: john@hancock.com	Start: January 22, 2007, 00:00 GMT
Name: Mr. John Hancock	End: February 7, 2007, 23:59 GMT

Questionnaire

Info: Please take a look at the [definitions](#) before you begin answering the questions. In order to work on a specific question, please click on it. A symbol on the right-hand side indicates whether a question has already been answered. You are free to skip any questions you do not wish to answer. Comments may be edited until the end of Round 1 on February 7, 2007.

SWOT Analysis

- Where do you see the strengths of integration architectures based on SWS? ✓
- Where do you see the weaknesses of integration architectures based on SWS? ✓
- What factors do you think will drive the use of integration architectures based on SWS in the future? ✓
- What factors do you think will restrict the use of integration architectures based on SWS in the future? ✓

Requirements

- What are, from your point of view, the functional requirements that integration architectures must fulfill? ✓
- What are, from your point of view, the qualitative requirements that integration architectures must fulfill? ✓
- Where do you see differences in the requirements for internal and external integration architectures? ✓

Expectations

- What positive effects of using an SWS-based integration architecture do you expect at the macro level? ✓
- What negative effects of using an SWS-based integration architecture do you expect at the macro level? ✓

Figure 41 Upper part of the qualitative questionnaire page.

The questionnaire page was created by the *questionnaire* file of the first round when an already registered user had logged in successfully or when any sampled individual had completed the registration.

As described in the previous section, on the qualitative questionnaire page, users could review personal details, take a look at important definitions, view the survey schedule, respond to selected questions and log out. Most functions could be accessed via the menu, but to respond to a question it was necessary to click on it directly.

The lower part of the questionnaire with a portion of the questions answered is depicted in Figure 42. According to Dillman, a sense of progress can be provided in an effort to avoid people quitting when they are only a few questions from the end [Dill00]. Symbols on the right-hand side indicated on the questionnaire page that a question had already been answered. A question was considered to be answered if either at least one of the text fields on the respective question page was completed or the *No Comment* checkbox was checked to indicate that a user did not want to answer a specific question. Comparable capabilities are typically not offered by self-administered survey methods that are not Web-based.

Delphi Study - Questionnaire - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <http://delphi.bachlechner.info/questionnaire1.php>

Name: Mr. John Hancock End: February 7, 2007, 23:59 GMT

Questionnaire

*Info: Please take a look at the **definitions** before you begin answering the questions. In order to work on a specific question, please click on it. A symbol on the right-hand side indicates whether a question has already been answered. You are free to skip any questions you do not wish to answer. Comments may be edited until the end of Round 1 on February 7, 2007.*

SWOT Analysis

1. Where do you see the strengths of integration architectures based on SWS? ✓
2. Where do you see the weaknesses of integration architectures based on SWS? ✓
3. What factors do you think will drive the use of integration architectures based on SWS in the future? ✓
4. What factors do you think will restrict the use of integration architectures based on SWS in the future? ✓

Requirements

5. What are, from your point of view, the functional requirements that integration architectures must fulfill? ✓
6. What are, from your point of view, the qualitative requirements that integration architectures must fulfill? ✓
7. Where do you see differences in the requirements for internal and external integration architectures? ✓

Expectations

8. What positive effects of using an SWS-based integration architecture do you expect at the macro level? ✓
9. What negative effects of using an SWS-based integration architecture do you expect at the macro level? ✓
10. What positive effects of using an SWS-based integration architecture do you expect at the micro level? ✓
11. What negative effects of using an SWS-based integration architecture do you expect at the micro level? ✓

Roadmap

12. What challenges do you know of that SWS research is facing today? ✓
13. What will be achieved in SWS research within the next five years? ✓
14. What are problems of current integration architectures that you believe can be solved with SWS? ✓
15. Do you know of any real-world case studies where SWS-based integration architectures have been used? ✓

If you have any questions, feel free to contact Daniel Bachlechner at +43-699-11202425, or by e-mail at daniel.bachlechner@uibk.ac.at

W3C XHTML 1.0

Done Internet

Figure 42 Lower part of the qualitative questionnaire page with some questions answered.

After clicking on one of the questions on the page of the qualitative questionnaire, the user was directed to the page of the respective question. For questions that had already been answered, the respective fields were preset but could be changed, if required. The question page is depicted in Figure 43. All questions were presented in a conventional format similar to that normally used for self-administered paper surveys. When the *Save and Return* button was clicked the user was directed from the *question* file back to the first *questionnaire* file. Later, the answer was stored in the database together with additional information such as date and time. For all questions in the first round of the survey, the same *question* file was used. The required data was taken from the database.

When clicking on *Definitions* in the menu or in the short message on the questionnaire page, definitions of important terms were displayed on a separate page. The definitions page was created by the *definitions* file and is displayed in Figure 44. After clicking on the *Close* button, the previous page was shown again.

A Delphi Study

Question 1

Where do you see the strengths of integration architectures based on SWS?

Info: Please indicate and briefly describe up to five strengths to which you attach particular importance. Structure your answer by using the separate text areas. You may enter as much text in each area as you like.

- flexibility of the architecture
- loose coupling
- machine understandable and processable
-
-

☐ No Comment

Info: Please check the box above if you do not wish to answer this question. Note that all comments that you may have already made on this question will be lost.

Figure 43 Question page.



Figure 44 Definitions page.

The files involved during the second round of the survey are shown in Figure 45.

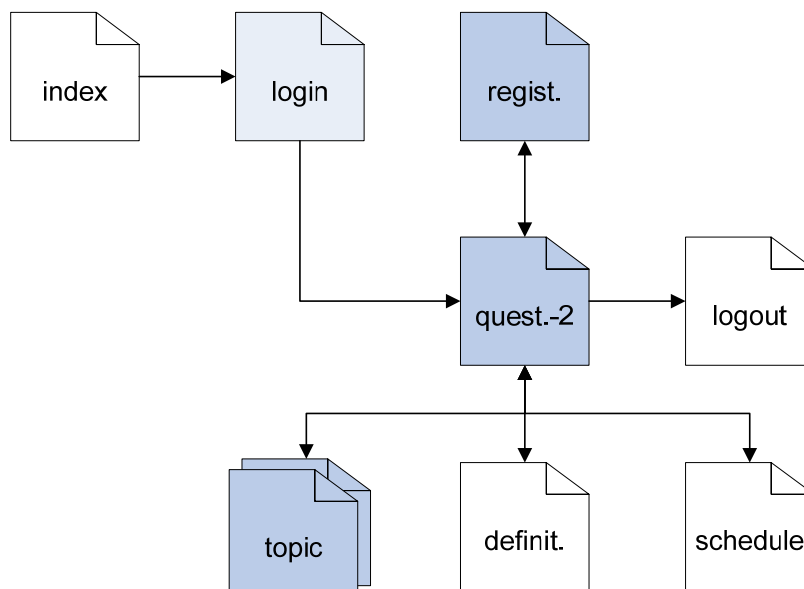


Figure 45 Files involved in the second round of the survey.

In this phase, the *registration* file lost one of its purposes, which was to handle user registrations. Figure 46 shows the quantitative questionnaire. In contrast to the first survey

round, the symbol indicating response progress had a finer granularity in this round in that it indicated the percentage of statements that had already been rated. When moving the cursor over the symbol, the exact percentage was displayed. Distinct symbols similar to those in the first round of the survey were used to indicate that either all or none of the statements of a specific topic had been rated. Furthermore, progress was divided into five levels of completeness with specific symbols that can be seen in Figure 46 assigned to topics 6 to 10.

In essence, the quantitative questionnaire page provided the same options as the qualitative questionnaire page. The topic page is shown in Figure 47. As already explained, the order of the statements was randomized to avoid order effects. To preserve clarity, statements that had already been rated were placed at the top. They could be changed, if required. The numeric codes on the right-hand side served as uniform identifiers of the statements. The explanation of the radio buttons was repeated after every ten statements to appear permanently visible also for users with a display resolution of 800 x 600 pixels. Again, the statements were presented in a conventional format similar to that typical for paper-based surveys.

Delphi Study - Questionnaire - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <http://delphi.bachlechner.info/questionnaire2.php>

E-Mail: john@hancock.com	Start: February 19, 2007, 00:00 GMT
Name: Mr. John Hancock	End: March 12, 2007, 23:59 GMT

Questionnaire

Info: In order to rate the statements related to a specific topic, please click on that topic. The symbol on the right-hand side indicates the percent of statements that have already been rated. You may make changes to ratings and comments until the end of Round 2 on March 4, 2007.

SWOT Analysis

1. [Strengths of Integration Architectures based on SWS](#) ✓
2. [Weaknesses of Integration Architectures based on SWS](#) ✓
3. [Factors driving the use of Integration Architectures based on SWS in the future](#) ✓
4. [Factors restricting the use of Integration Architectures based on SWS in the future](#) ✓

Requirements

5. [Functional Requirements that Integration Architectures must fulfil](#) ✓
6. [Qualitative Requirements that Integration Architectures must fulfil](#) ✓
7. [Differences in the Requirements for Internal and External Integration Architectures](#) ✓

Expectations

8. [Positive Effects of using an SWS-based Integration Architecture at the Macro Level](#) ✓
9. [Negative Effects of using an SWS-based Integration Architecture at the Macro Level](#) ✓
10. [Positive Effects of using an SWS-based Integration Architecture at the Micro Level](#) ✓
11. [Negative Effects of using an SWS-based Integration Architecture at the Micro Level](#) ○

Roadmap

12. [Challenges SWS Research is facing today](#) ○
13. [Achievements in SWS Research within the next five years](#) ○
14. [Problems of current Integration Architectures that can be solved with SWS](#) ○

If you have any questions, feel free to contact Daniel Bachlechner at +43-699-11202425, or by e-mail at daniel.bachlechner@sabk.ac.at

W3C XHTML 1.0 ✓

Done Internet

Figure 46 Quantitative questionnaire page.

Delphi Study - Questionnaire - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <http://delphi.bachlechner.info/topic.php?uid=1&tid=1>

Relevance and Applicability of Semantic Web Services in Electronic Business

A Delphi Study

Topic 1

Statements

Strengths of Integration Architectures based on SWS (as compared with state-of-the-art architectures):

Info: Please rate how strongly you agree or disagree with each of the following statements. Note that the order of the statements is randomized in order to avoid biases. To preserve clarity, statements that have already been rated are placed at the top. The numeric code on the right-hand side serves for uniform identification of specific statements only. If you do not wish to rate a statement, please select "No Comment."

	Strongly Agree	Rather Agree	Undecided	Rather Disagree	Strongly Disagree	No Comment	
Improved service validation capability.	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.36)
Reduced error proneness.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.11)
Robustness against changing environments.	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.23)
Accelerated integration.	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.10)
Relief from repetitive tasks.	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.9)
Use of open standards.	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.22)
Improved mediation between services.	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.37)

Done Internet

Figure 47 Topic page.

Delphi Study - Questionnaire - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <http://delphi.bachlechner.info/topic.php?uid=1&tid=1>

Improved service composition capability.	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.34)
Facilitated error treatment.	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.12)
Compliance with business and legal rules.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	(1.39)
Improved service orchestration capability.	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.32)
Platform independence.	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.26)
Facilitated customization of systems.	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	(1.17)

Comments

Any additional comments you wish to add on existing or missing statements or on the topic in general are greatly appreciated.

Info: If you comment on an existing statement, please indicate its numeric code. You may enter as much text as you like.

Save and Return

If you have any questions, feel free to contact Daniel Bachlechner at +43-699-11202425, or by e-mail at daniel.bachlechner@uibk.ac.at

W3C XHTML 1.0

Done Internet

Figure 48 Text field for comments on a topic page.

The lower part of every topic page contained a text field for comments. The field is shown in Figure 48. When the *Save and Return* button was clicked, the user was directed from the *topic* file back to the second *questionnaire* file. Later, the ratings were stored in the database together with additional information. For all topics of the second round of the survey the same *topic* file was used. As in the first round, the required data was taken from the database.

The schedule page that was accessible from both questionnaire pages by clicking on *Schedule* in the menu is displayed in Figure 49. After clicking on the *Close* button, the previous page was shown again. A symbol on the right-hand side indicated the progress of the survey. The page created by the *schedule* file was always kept up to date.

The process of logging out is described in section 5.2.4.5.

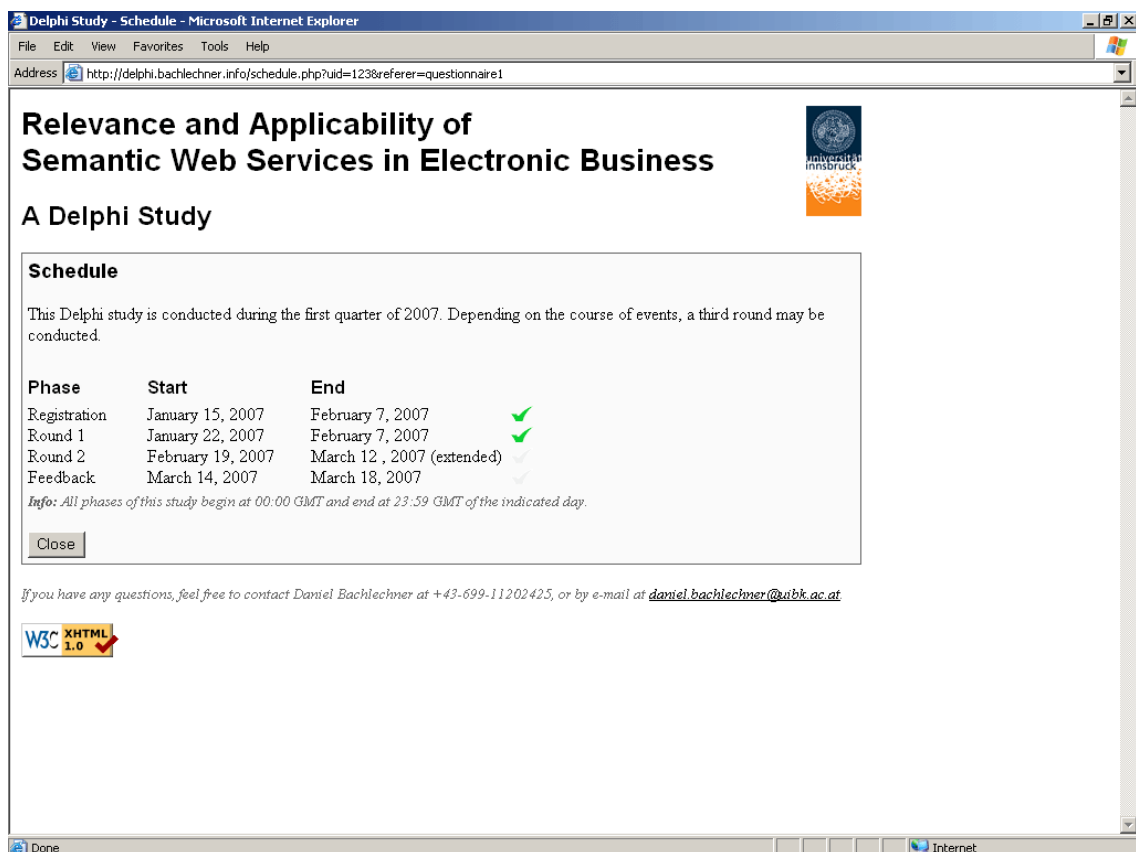


Figure 49 Schedule page.

5.2.4.4 Feedback

During the feedback phase the *login* file forwarded directly to the *feedback* file. The files involved in the feedback phase are illustrated in Figure 50.

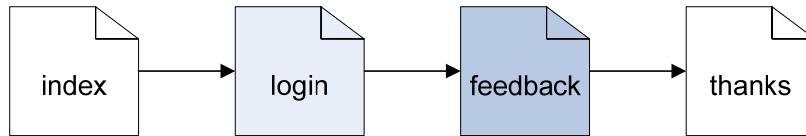


Figure 50 Files involved in the feedback phase.

If the login was successful, the feedback page shown in Figure 51 was displayed.

The feedback consisted of comments about the satisfaction with the study in general and ratings of the perceived functionality and usability of the Web-based survey system.

Delphi Study - Feedback - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <http://delphi.bachlechner.info/feedback.php>

Relevance and Applicability of Semantic Web Services in Electronic Business

A Delphi Study

Feedback

Was the participation in this Delphi study a worthwhile experience?

Info: There is nothing as useful as the ideas, opinions, and questions of participants. We welcome all feedback, and we will consider it in future studies. Completion of all fields is optional. Comments may be edited until March 18, 2007.

the results of this comprehensive study could really be useful and could significantly help to assign next priorities to requirements on SWS

Survey System

	Excellent	Good	Satisfactory	Fair	Poor
Functionality:	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Usability:	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Info: Please rate the functionality and usability of the Web-based survey system used for this Delphi study.

If you have any questions, feel free to contact Daniel Bachlechner at +43-699-11202425, or by e-mail at daniel.bachlechner@uibk.ac.at

W3C XHTML 1.0

Figure 51 Feedback page.

The collected data was sent to the *thanks* file and stored in the database with statistical information after the user clicked on the *Save* button.

5.2.4.5 Thanks and Logout

In all phases either the *thanks* or the *logout* file played a role. The *thanks* file stored personal details in the database after registration during the pre-registration phase. During the first round of the survey the first *questionnaire* file took over this task. In the feedback phase, again, the *thanks* file stored the collected data in the database. Furthermore, the file

always displayed a short confirmation message to inform users of the success of their actions.

The *logout* file was used for similar tasks during the two rounds of the survey. Unlike the *thanks* file, the *logout* file did not write to the database. Storing data was the responsibility of the *questionnaire* files during both survey rounds.

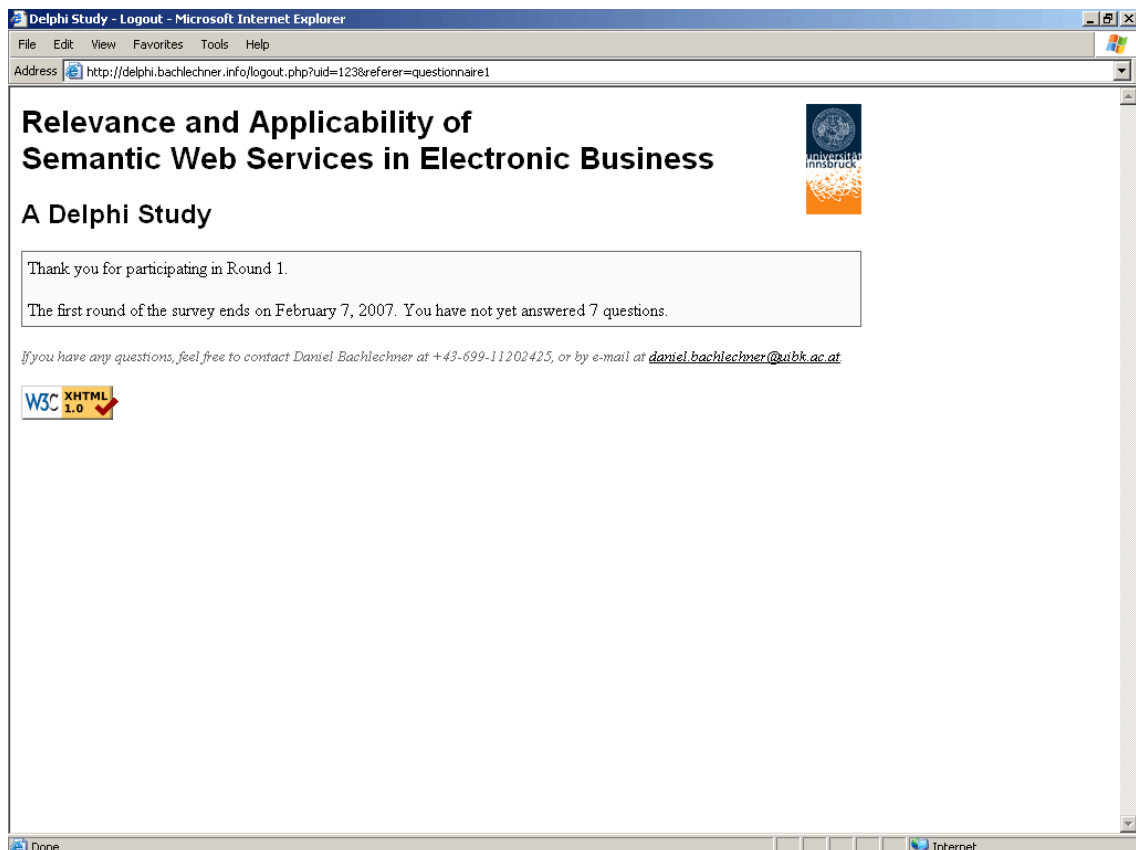


Figure 52 Logout page during the first round.

The logout page for the first round of the survey is displayed in Figure 52. During the first round, the number of open questions was indicated in the message displayed.

5.3 Data Analysis

Several techniques were used to process and analyze the data collected in our Delphi study. On the one hand, the responses from the first survey round had to be aggregated and transformed into statements to be rated in the second round. On the other hand, the quantitative data collected in the second round of the survey and in the feedback phase had to be analyzed.

The aggregation of the qualitative data gathered in the first round of the survey was a quite elaborate task. The facet theory was helpful in formulating sets of statements for the second round of the survey by reducing the complexity of the responses from the first round. First, the responses were broken into unit statements. Then, every statement was related to the corresponding responses and vice versa to ensure that it originated from at least one first round response and that every response was transformed into a statement. We tried to make every statement self-contained and mutually exclusive to others, but this goal could not be achieved completely.

In the second round of the survey and during the feedback phase, both quantitative and qualitative data was collected. Statistical Program for the Social Sciences¹ (SPSS) was used to analyze the quantitative data. The quantitative data consisted of ratings on bipolar rating scales. The mean and the standard deviation were computed with SPSS for each statement. We arranged the statements in the order of their average ratings and took a number of statements from the top. The standard deviations of the ratings were computed to discover the polarization of opinions.

5.4 Results

The expertise of the participants in the area of research was gathered to evaluate their qualification for the study. However, no experts were excluded from the study because of lack of sufficient expertise. The expertise distribution grouped by background is shown in Figure 53. The scale ranged from 1 to 5, with 1 representing *Novice* and 5 representing *Expert*. None of the participants ranked in the lowest category.

The 38 experts who participated in both rounds of the study were from all parts of the world and were employed at major universities and enterprises. While 21 of the experts had academic backgrounds, 17 had industrial ones. Gender information was gathered only to improve the personalization of the communication process. We proceeded on the assumption that gender has no influence on the responses of experts with respect to the objective of the study. Providing information about affiliation, position and country of work was optional. The values were used to check whether the participants were distributed more or less equally.

¹ For more information on SPSS, see <http://www.spss.com/>.

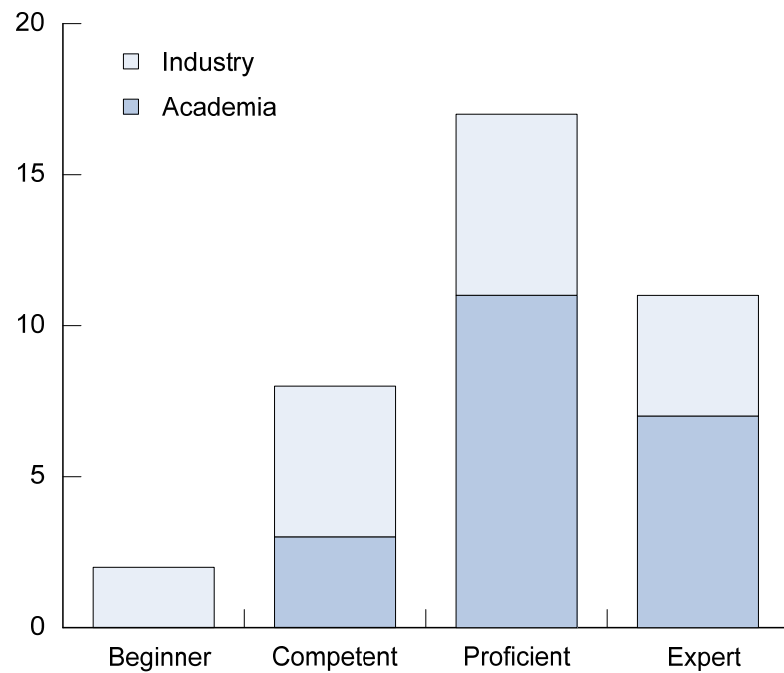


Figure 53 Expertise distribution subdivided in backgrounds.

Below, we describe the results based on the entire group of respondents and compare the results of the respondents having academic and industrial backgrounds, respectively. The scale ranged from 1 to 5, with 1 representing *Strong Disagreement* and 5 representing *Strong Agreement*. The number of respondents varies because respondents were free to leave statements unrated or to check the *No Comment* box.

In section 5.4.1 the results of the SWOT analysis are described. The results related to requirements are discussed in section 5.4.2 and the ones related to expectations in section 5.4.3. Finally, in section 5.4.4, we describe the results related to future developments in SWS research.

5.4.1 SWOT Analysis

In section 5.4.1.1, we describe the results with regard to the strengths of integration architectures based on SWSs. In section 5.4.1.2 the weaknesses are outlined. In sections 5.4.1.3 and 5.4.1.4, we discuss future opportunities and threats. The respondents rated 40 statements with regards to each of the elements of the SWOT analysis.

5.4.1.1 Strengths

Table 1 lists the most important strengths of integration architectures based on SWSs.

Table 1 Most important strengths of SWS-based integration architectures.

	N	Mean	Std. Dev.
Improved service discovery capability.	36	4.28	0.659
Facilitated interoperability.	34	4.26	0.864
Facilitated reuse of services.	34	4.00	0.853
Improved mediation between services.	34	4.00	0.953
Explicit definitions of conditions and functionalities.	34	3.94	0.886
Improved service composition capability.	35	3.89	0.993
Enhanced process and term definitions.	33	3.88	1.023
Use of ontologies.	35	3.83	0.891
Service-orientation.	33	3.82	0.950
Improved service validation capability.	36	3.75	0.806

While Table 2 lists the most agreed strengths of SWS-based integration architectures, Table 3 lists the most controversial strengths.

Table 2 Most agreed strengths of SWS-based integration architectures.

	N	Mean	Std. Dev.
Improved service discovery capability.	36	4.28	0.659
Improved service validation capability.	36	3.75	0.806
Improved service invocation capability.	34	3.59	0.821
Facilitated system extensions.	32	3.53	0.842
Facilitated error treatment.	36	3.03	0.845

Table 3 Most controversial strengths of SWS-based integration architectures.

	N	Mean	Std. Dev.
Use of open standards.	34	3.59	1.234
Compliance with business and legal rules.	32	3.44	1.190
Lightweight approach.	36	2.44	1.182
Well-accepted communication infrastructure.	34	3.21	1.149
Improved service orchestration capability.	34	3.71	1.142

Improved service discovery capability represents not only the most important strength of integration architectures based on SWSs but it is also the most agreed aspect in this regard. The second most agreed aspect, improved service validation capability, is also among the ten key strengths. With an average rating greater than or equal to 4, facilitated

interoperability, facilitated reuse of services and the improved mediation between services also play key roles with respect to strengths of integration architectures based on SWSs. Further strengths worth mentioning are explicit definitions of conditions and functionalities, and enhanced process and term definitions. Finally, improved service composition capabilities, the use of ontologies, and service orientation were perceived as rather important strengths.

The respondents were most discordant with respect to the statement that the use of open standards represents an important strength of integration architectures based on SWSs. There was also some dissent about the compliance of SWS-based integration architectures with business and legal rules. Furthermore, it is noteworthy that there is no clear agreement on the weight of the approach. By tendency, the respondents rather disagree with the statement that the approach is lightweight.

While Table 4 lists the most important strengths of integration architectures based on SWS from an academic perspective, Table 5 lists the most important strengths from an industrial perspective.

Table 4 Most important strengths of SWS-based integration architectures from an academic perspective.

	N	Mean	Std. Dev.
Improved service discovery capability.	21	4.24	0.700
Facilitated interoperability.	20	4.20	0.894
Improved mediation between services.	20	4.10	0.968
Enhanced process and term definitions.	18	4.06	1.056
Formalization of systems.	21	4.00	0.837
Facilitated reuse of services.	20	3.95	0.945
Use of ontologies.	20	3.95	0.945
Explicit definitions of conditions and functionalities.	20	3.95	0.686
Improved service validation capability.	21	3.81	0.814
Improved service composition capability.	21	3.81	1.030

The academic and industrial points of view are generally in accord concerning the strengths of integration architectures based on SWSs. However, respondents with industrial backgrounds perceive the goal-based paradigm as a clear strength while respondents with academic backgrounds rated the statement rather neutral. Conversely, academic respondents see the formalization of systems as a key strength but respondents with

industrial backgrounds rated it only slightly above average. Loose coupling and increased flexibility play a key role particularly for respondents with industrial backgrounds.

Table 5 Most important strengths of SWS-based integration architectures from an industrial perspective.

	N	Mean	Std. Dev.
Facilitated interoperability.	14	4.36	0.842
Improved service discovery capability.	15	4.33	0.617
Facilitated reuse of services.	14	4.07	0.730
Improved service composition capability.	14	4.00	0.961
Goal-based paradigm.	15	3.93	0.704
Explicit definitions of conditions and functionalities.	14	3.93	1.141
Service-orientation.	14	3.93	0.917
Loose coupling.	15	3.87	0.640
Improved mediation between services.	14	3.86	0.949
Increased flexibility.	14	3.79	1.122

Table 6 lists the most controversial strengths of SWS-based integration architectures comparing the two groups of respondents.

Table 6 Most controversial strengths of SWS-based integration architectures comparing the two groups of respondents.

	Mean	
	Academia	Industry
Goal-based paradigm.	3.16	3.93
Formalization of systems.	4.00	3.40
Compliance with business and legal rules.	3.67	3.14
Improved service choreography capability.	3.81	3.29
Facilitated system upgrades.	3.47	3.00

5.4.1.2 Weaknesses

Table 7 lists the most important weaknesses of integration architectures based on SWSs.

Table 7 Most important weaknesses of SWS-based integration architectures.

	N	Mean	Std. Dev.
Use of immature technologies.	32	4.25	0.718
Description overhead.	32	4.09	0.856
High initial start-up costs.	32	4.06	0.948
Labor-intensive service specification.	32	4.00	0.984
Software engineers are not ontology experts.	32	3.97	1.062
Lack of agreement on description depth.	29	3.90	1.012
Not yet adopted.	32	3.88	0.976
Unsatisfactory support of change management.	31	3.87	0.957
Lack of effective tools.	31	3.87	1.088
High system complexity.	32	3.84	0.954

While Table 8 lists the most agreed weaknesses of SWS-based integration architectures, Table 9 lists the most controversial weaknesses.

Table 8 Most agreed weaknesses of SWS-based integration architectures.

	N	Mean	Std. Dev.
Unsatisfactory service validation features.	29	3.14	0.693
Use of immature technologies.	32	4.25	0.718
Contradiction to REST.	26	2.62	0.852
Description overhead.	32	4.09	0.856
Lack of a dominant design.	32	3.50	0.880

Table 9 Most controversial weaknesses of SWS-based integration architectures.

	N	Mean	Std. Dev.
Lack of standards.	32	3.59	1.316
Unintuitive concepts.	31	3.35	1.305
Lack of semantic annotations.	32	3.44	1.243
Lack of service registration features.	27	3.00	1.240
Performance.	32	3.28	1.198

Interestingly, none of the statements to which the respondents attached much importance, is among the most agreed statements. This suggests that there is some uncertainty with regard to weaknesses. Nevertheless, the use of immature technologies, description overhead and high initial start-up costs represent clear and more or less agreed weaknesses

with average ratings greater than 4. Furthermore, the facts that software engineers are usually not ontology experts and that integration architectures based on SWSs have not yet been adopted seem to be serious issues. The respondents also sense a lack of effective tools. The fact that high system complexity is among the most important weaknesses is in line with the controversy with regard to the statement that using an SWS-based integration architecture represents a lightweight approach.

While Table 10 lists the most important weaknesses of integration architectures based on SWS from an academic perspective, Table 11 lists the most important weaknesses from an industrial perspective.

Table 10 Most important weaknesses of SWS-based integration architectures from an academic perspective.

	N	Mean	Std. Dev.
High initial start-up costs.	19	4.42	0.607
Lack of agreement on description depth.	16	4.38	0.885
Use of immature technologies.	19	4.26	0.806
High system complexity.	19	4.21	0.855
Description overhead.	19	4.16	1.068
High service development costs.	19	4.05	0.780
Unsatisfactory support of change management.	19	4.05	0.848
Labor-intensive service specification.	19	4.05	1.026
Software engineers are not ontology experts.	19	3.89	1.049
Unsatisfactory life-cycle support.	18	3.89	0.963

While respondents from both groups agree on weaknesses such as the use of immature technologies, description overhead and labor-intensive service specification, there are also many controversies. Respondents with academic backgrounds, for instance, rated the lack of agreement on description depth, high system complexity and high service development costs as major weaknesses. From an industrial perspective, these aspects were rated only slightly above average. Instead, respondents with industrial backgrounds perceive the fact that integration architectures based on SWSs have not yet been adopted as a key weakness. Furthermore, the lack of a dominant design seems to be a serious issue from an industrial viewpoint.

Table 11 Most important weaknesses of SWS-based integration architectures from an industrial perspective.

	N	Mean	Std. Dev.
Use of immature technologies.	13	4.23	0.599
Not yet adopted.	13	4.15	0.689
Software engineers are not ontology experts.	13	4.08	1.115
Description overhead.	13	4.00	0.408
Labor-intensive service specification.	13	3.92	0.954
Unsatisfactory security features.	13	3.92	0.641
Lack of effective tools.	12	3.92	0.996
Lack of standards.	13	3.77	1.301
Unsatisfactory management capability.	11	3.64	0.674
Lack of a dominant design.	13	3.62	0.768

As per one of the experts, there are numerous competing SWS frameworks available at the moment and misinformation regarding the differences is widely spread. While some frameworks are aimed at lightweight solutions, others also support more complicated tasks such as service composition but rely on much less mature technologies. Consensus on one of the frameworks is probably unavoidable to fully enable interoperability.

Table 12 lists the most controversial weaknesses of SWS-based integration architectures comparing the two groups of respondents.

Table 12 Most controversial weaknesses of SWS-based integration architectures comparing the two groups of respondents.

	Mean	
	Academia	Industry
High degree of formality.	3.79	2.54
Unintuitive concepts.	3.83	2.69
Lack of agreement on description depth.	4.38	3.31
High system complexity.	4.21	3.31
High service development costs.	4.05	3.17

5.4.1.3 Opportunities

Table 13 lists the most important factors driving the use of integration architectures based on SWSs in the future.

Table 13 Most important opportunities of SWS-based integration architectures.

	N	Mean	Std. Dev.
Availability of business cases.	32	4.313	0.931
Need for service interoperability.	32	4.219	0.792
Availability of compliant middleware implementations.	32	4.188	0.738
Availability of effective tools.	32	4.125	1.100
Preceding agreement on standards.	31	4.097	0.908
Proven cost effectiveness.	31	4.065	1.063
Need for flexible integration.	32	4.000	0.880
Increasing dynamics of cooperations.	30	4.000	0.871
Compelling value proposition.	29	3.966	0.944
Availability of best practices.	31	3.935	0.929

While Table 14 lists the most agreed factors driving the use of SWS-based integration architectures in the future, Table 15 lists the most controversial factors.

Table 14 Most agreed opportunities of SWS-based integration architectures.

	N	Mean	Std. Dev.
Availability of compliant middleware implementations.	32	4.188	0.738
Incorporation of qualitative specifications.	29	3.552	0.783
Need for service interoperability.	32	4.219	0.792
Availability of methodologies.	32	3.906	0.818
Increasing usability of services.	31	3.710	0.824

Table 15 Most controversial opportunities of SWS-based integration architectures.

	N	Mean	Std. Dev.
Consolidated pattern algebra.	26	2.808	1.327
Availability of integrated development environments.	32	3.875	1.129
Failure of existing integration architectures.	29	3.448	1.121
Preceding globalization.	27	3.000	1.109
Availability of effective tools.	32	4.125	1.100

The availability of compliant middleware implementations and the need for service interoperability are among both the most important factors driving the use of SWS-based integration architectures and the most agreed statements with regard to opportunities. This,

together with both also having an average rating significantly above 4, suggests a general agreement concerning the fact that they will play a major role in the future of SWS-based integration architectures. The most important opportunity, however, lies in the availability of business cases. Furthermore, the availability of effective tools, preceding agreement on standards and proven cost effectiveness exhibit an average rating greater than 4. Interestingly, the availability of effective tools is among the five most controversial statements with regard to factors driving the use of integration architectures based on SWSs. Nevertheless, the need for flexible integration and increasing dynamics of cooperations are recognized as opportunities.

While Table 16 lists the most important factors driving the use of integration architectures based on SWS in the future from an academic perspective, Table 17 lists the most important factors from an industrial perspective.

Table 16 Most important opportunities of SWS-based integration architectures from an academic perspective.

	N	Mean	Std. Dev.
Availability of business cases.	19	4.421	0.769
Proven cost effectiveness.	18	4.333	0.767
Availability of compliant middleware implementations.	19	4.211	0.713
Increasing dynamics of cooperations.	19	4.158	0.765
Availability of best practices.	19	4.158	0.898
Need for service interoperability.	19	4.053	0.911
Compelling value proposition.	19	4.053	0.911
Increasing support from standardization bodies.	19	4.000	0.882
Availability of effective tools.	19	4.000	1.247
Proliferation of services.	18	4.000	0.907

From an academic point of view, proven cost effectiveness and cooperation across industries, academia and other interest organizations play much more important roles than they do from an industrial perspective.

Table 17 Most important opportunities of SWS-based integration architectures from an industrial perspective.

	N	Mean	Std. Dev.
Need for service interoperability.	13	4.462	0.519
Preceding agreement on standards.	12	4.333	0.985
Availability of effective tools.	13	4.308	0.855
Buy-in from large integration players.	12	4.250	0.866
Potential savings.	11	4.182	1.079
Increasing dynamics of systems.	11	4.182	0.874
Availability of business cases.	13	4.154	1.144
Availability of compliant middleware implementations.	13	4.154	0.801
Availability of integrated development environments.	13	4.154	1.144
Need for effective collaboration.	12	4.083	0.996

In contrast, the increasing dynamics of systems are perceived as an important opportunity only from respondents with industrial backgrounds. Furthermore, buy-ins from large integration players and potential savings are of major importance only from an industrial viewpoint. With regard to the majority of the opportunities there is quite an accord between the two groups of respondents.

Table 18 lists the most controversial factors driving the use of SWS-based integration architectures in the future comparing the two groups of respondents.

Table 18 Most controversial opportunities of SWS-based integration architectures comparing the two groups of respondents.

	Mean	
	Academia	Industry
Proven cost effectiveness.	4.333	3.692
Cooperation across industries, academia and interest organizations.	3.947	3.308
Preceding globalization.	3.235	2.600
Consolidated pattern algebra.	2.588	3.222
Increasing dynamics of systems.	3.579	4.182

5.4.1.4 Threats

Table 19 lists the most important factors restricting the use of integration architectures based on SWSs in the future.

Table 19 Most important threats to SWS-based integration architectures.

	N	Mean	Std. Dev.
Difficulty of describing semantics.	31	4.097	0.700
Lack of effective tools.	31	4.065	0.964
Unproven cost effectiveness.	31	3.968	0.752
Limited consideration of business needs.	31	3.968	0.912
Unavailability of convincing case studies.	30	3.967	1.033
Lack of integration into middleware technologies.	29	3.931	0.884
Market does not understand values and capabilities.	32	3.906	1.088
Increasing complexity.	31	3.903	0.944
Lack of compelling value proposition.	31	3.871	1.204
Lack of skilled developers.	29	3.862	0.990

While Table 20 lists the most agreed factors restricting the use of SWS-based integration architectures in the future, Table 21 lists the most controversial factors.

Table 20 Most agreed threats to SWS-based integration architectures.

	N	Mean	Std. Dev.
No broad scale adoption.	31	3.806	0.543
Difficulty of describing semantics.	31	4.097	0.700
Unproven cost effectiveness.	31	3.968	0.752
High costs.	30	3.767	0.858
Lack of integration into middleware technologies.	29	3.931	0.884

Table 21 Most controversial threats to SWS-based integration architectures.

	N	Mean	Std. Dev.
Old-fashioned thinking in industries.	29	3.207	1.497
Heterogeneity of workflows and business processes.	30	3.067	1.363
Lack of funding.	29	3.069	1.307
Fragmentation of research.	29	3.034	1.295
Difficulty of ontology learning.	31	3.355	1.253

Difficulty in describing semantics, unproven cost effectiveness and lack of integration into middleware technologies are three of the most important factors restricting the use of SWS-based integration architectures that are also among the most agreed statements in this

regard. The lack of effective tools, limited consideration of business needs and unavailability of convincing case studies also play key roles with regard to threats to integration architectures based on SWSs. Furthermore, SWS-based integration architectures can be threatened by a market not understanding the associated values and capabilities. Finally, according to the respondents, the lack of a compelling value proposition as well as the lack of skilled developers are also serious threats.

While Table 22 lists the most important factors restricting the use of integration architectures based on SWS in the future from an academic perspective, Table 23 lists the most important factors from an industrial perspective.

Table 22 Most important threats to SWS-based integration architectures from an academic perspective.

	N	Mean	Std. Dev.
Difficulty of describing semantics.	18	4.389	0.502
Unavailability of convincing case studies.	18	4.222	0.878
Increasing complexity.	18	4.111	0.900
Unproven cost effectiveness.	18	4.111	0.832
High costs.	18	4.056	0.639
Failure to reach critical mass.	18	4.056	0.873
Limited consideration of business needs.	18	4.056	0.873
Lack of skilled developers.	16	4.000	1.033
Lack of integration into middleware technologies.	17	4.000	0.935
Lack of effective tools.	18	3.944	1.056

The two groups of respondents regard quite different aspects of the threats to integration architectures based on SWSs as important. For respondents with industrial backgrounds the limited interest of vendors and the lack of industrial commitment represent key threats, while respondents with academic backgrounds rated the respective statements only slightly above average. The most important threat from an academic perspective is the difficulty in describing semantics, while the lack of effective tools is the most critical aspect from an industrial point of view.

Table 23 Most important threats to SWS-based integration architectures from an industrial perspective.

	N	Mean	Std. Dev.
Lack of effective tools.	13	4.231	0.832
Lack of industrial commitment.	13	4.077	1.038
Limited interest of vendors.	13	4.077	0.954
Market does not understand values and capabilities.	13	4.000	0.816
Difficulty of catalyzing the market.	12	4.000	0.739
Dominant vendors use own technology.	12	4.000	1.044
Lack of common terminology for service description.	13	3.846	1.068
Inability to communicate strengths.	13	3.846	0.987
Limited consideration of business needs.	13	3.846	0.987
Lack of compelling value proposition.	13	3.846	1.345

Both groups sense a significant threat either in the difficulty in catalyzing the market or the failure to reach critical mass. Respondents from industry fear an inability to communicate the strengths of integration architectures based on SWSs more than respondents with academic backgrounds. The limited consideration of business needs is perceived as a noteworthy threat by both groups. While the lack of semantic annotations and the heterogeneity of workflows and business processes pose serious threats for respondents with academic backgrounds, these concerns are not shared by participants with industrial backgrounds. The situation is the other way round with regard to the lack of funding.

Table 24 lists the most controversial factors restricting the use of SWS-based integration architectures in the future comparing the two groups of respondents.

Table 24 Most controversial threats to SWS-based integration architectures comparing the two groups of respondents.

	Mean	
	Academia	Industry
Lack of semantic annotations.	3.889	2.583
Heterogeneity of workflows and business processes.	3.529	2.462
Limited interest of vendors.	3.167	4.077
Lack of funding.	2.765	3.500
Lack of industrial commitment.	3.353	4.077

5.4.2 Requirements

In sections 5.4.2.1 and 5.4.2.2, the functional and qualitative requirements that integration architectures must fulfill, are discussed. In section 5.4.2.3, we outline the results related to the differences between internal and external integration architectures.

5.4.2.1 Functional Requirements

The respondents rated 36 statements related to functional requirements that integration architectures must fulfill. Table 25 lists the most important functional requirements.

Table 25 Most important functional requirements that integration architectures must fulfill.

	N	Mean	Std. Dev.
Clear definitions of interfaces.	31	4.516	0.626
Semantic interoperability among services.	32	4.375	0.707
Support of service reuse.	31	4.323	0.702
Support of service registration.	31	4.290	0.693
Support of workflow definitions.	31	4.258	0.773
Support of service life-cycles.	31	4.258	0.729
Incorporation of service level agreements.	30	4.200	0.805
Support of interface modifications.	30	4.167	0.592
Mediation between services.	31	4.161	0.820
Validation of services.	31	4.129	0.718

While Table 26 lists the most agreed functional requirements that integration architectures must fulfill, Table 27 lists the most controversial functional requirements.

Table 26 Most agreed functional requirements that integration architectures must fulfill.

	N	Mean	Std. Dev.
Support of interface modifications.	30	4.167	0.592
Clear definitions of interfaces.	31	4.516	0.626
Support of configuration and customization of services.	32	4.094	0.641
Support of service registration.	31	4.290	0.693
Support of service reuse.	31	4.323	0.702

Table 27 Most controversial functional requirements that integration architectures must fulfill.

	N	Mean	Std. Dev.
Automatic resolution of semantic conflicts.	32	3.469	1.367
Automatic service selection.	30	3.600	1.276
Automatic service discovery.	32	3.844	1.247
Dynamic composition of services.	32	3.719	1.170
Provision of security features.	32	3.906	1.146

Interestingly, all ten most important functional requirements that integration architectures must fulfill have an average rating significantly above 4 and all of the most agreed statements with regard to functional requirements, except the support of configuration and customization of services, are also among the most important functional requirements. This suggests that the respondents are generally in accord with regard to the functional requirements integration architectures must fulfill. The most important functional requirements are the clear definition of interfaces, semantic interoperability among services and the support of service reuse. Furthermore, support of service registration, workflow definitions and service life-cycles play key roles. The incorporation of service level agreements was also rated quite high.

The respondents were most discordant with respect to aspects such as automatic resolution of semantic conflicts, automatic service selection and automatic service discovery. It is remarkable that automatic services discovery and dynamic composition of services are also among the most controversial aspects with regard to functional requirements because improved service discovery and composition are among the most important strengths with regard to SWS-based integration architectures. However, it is worth mentioning that all functional requirements, including the ones discussed above, were rated rather high by the majority of respondents. Both have average ratings greater than 3.7.

While Table 28 lists the most important functional requirements that integration architectures must fulfill from an academic perspective, Table 29 lists the most important functional requirements from an industrial perspective.

Table 28 Most important functional requirements that integration architectures must fulfill from an academic perspective.

	N	Mean	Std. Dev.
Support of service registration.	18	4.444	0.856
Clear definitions of interfaces.	18	4.389	0.698
Support of service life-cycles.	18	4.333	0.767
Mediation between services.	19	4.316	0.582
Support of service reuse.	18	4.278	0.752
Support of workflow definitions.	18	4.278	0.752
Incorporation of service level agreements.	18	4.278	0.826
Semantic interoperability among services.	19	4.263	0.562
Support of configuration and customization of services.	19	4.263	0.562
Support of service adaptation.	18	4.167	0.786

Table 29 Most important functional requirements that integration architectures must fulfill from an industrial perspective.

	N	Mean	Std. Dev.
Clear definitions of interfaces.	13	4.692	0.480
Semantic interoperability among services.	13	4.538	0.877
Validation of services.	12	4.500	0.522
Support of service reuse.	13	4.385	0.650
Support of static and dynamic relationship management.	11	4.364	0.505
Support of interface modifications.	12	4.333	0.492
Provision of unambiguous semantics.	12	4.333	0.778
Support of the implementation of business models.	12	4.333	0.492
Support of workflow definitions.	13	4.231	0.832
Provision of trust and reputation mechanisms.	13	4.231	0.599

It is remarkable that the ratings of statements related to functional requirements such as the support of the implementation of business models, automatic service selection, provision of trust and reputation mechanisms, validation of services and provision of unambiguous semantics are all rated significantly higher by respondents with industrial backgrounds as compared with respondents with academic backgrounds.

Table 30 lists the most controversial qualitative requirements that integration architectures must fulfill comparing the two groups of respondents.

Table 30 Most controversial functional requirements that integration architectures must fulfill comparing the two groups of respondents.

	Mean	
	Academia	Industry
Support of the implementation of business models.	3.471	4.333
Automatic service selection.	3.278	4.083
Provision of trust and reputation mechanisms.	3.500	4.231
Validation of services.	3.895	4.500
Provision of unambiguous semantics.	3.737	4.333

5.4.2.2 Qualitative Requirements

The respondents rated 18 statements related to qualitative requirements that integration architectures must fulfill, Table 25 lists the most important qualitative requirements.

Table 31 Most important qualitative requirements that integration architectures must fulfill.

	N	Mean	Std. Dev.
Reliability.	30	4.600	0.498
Availability.	30	4.467	0.776
Robustness.	30	4.467	0.860
Understandability.	30	4.433	0.626
Extensibility.	30	4.433	0.626
Adaptability.	30	4.400	0.724
Manageability.	30	4.400	0.621
Stability.	30	4.367	0.890
Scalability.	30	4.333	0.711
User-friendliness.	29	4.310	0.604

While Table 32 lists the most agreed qualitative requirements that integration architectures must fulfill, Table 34 lists the most controversial qualitative requirements.

Table 32 Most agreed qualitative requirements that integration architectures must fulfill.

	N	Mean	Std. Dev.
Reliability.	30	4.600	0.498
User-friendliness.	29	4.310	0.604
Manageability.	30	4.400	0.621
Understandability.	30	4.433	0.626
Extensibility.	30	4.433	0.626

Table 33 Most controversial qualitative requirements that integration architectures must fulfill.

	N	Mean	Std. Dev.
Ease of use.	30	3.900	1.029
Universal applicability.	29	3.448	0.948
Stability.	30	4.367	0.890
Robustness.	30	4.467	0.860
Safety.	26	4.231	0.815

All of the most agreed statements with regard to qualitative requirements are also among the most important qualitative requirements. According to the study's results, reliability, availability and robustness play key roles. Participants were most discordant with respect to the statement that ease of use is an important qualitative requirement that integration architectures must fulfill. However, ease of use is not among the most important aspects, from neither an academic nor an industrial perspective. Robustness and stability, which are among the most controversial qualitative requirements, are also among the most important qualitative requirements.

While Table 34 lists the most important qualitative requirements that integration architectures must fulfill from an academic perspective, Table 35 lists the most important qualitative requirements from an industrial perspective.

Table 34 Most important qualitative requirements that integration architectures must fulfill from an academic perspective.

	N	Mean	Std. Dev.
Robustness.	19	4.579	0.607
Reliability.	19	4.579	0.507
Extensibility.	19	4.526	0.697
Understandability.	19	4.474	0.697
Adaptability.	19	4.474	0.841
Manageability.	19	4.421	0.692
Stability.	19	4.421	0.692
User-friendliness.	19	4.316	0.671
Modular architecture.	19	4.316	0.820
Adherence to standards.	19	4.316	0.582

Table 35 Most important qualitative requirements that integration architectures must fulfill from an industrial perspective.

	N	Mean	Std. Dev.
Availability.	11	4.818	0.405
Reliability.	11	4.636	0.505
Performance.	11	4.545	0.522
Understandability.	11	4.364	0.505
Manageability.	11	4.364	0.505
Scalability.	11	4.364	0.505
User-friendliness.	10	4.300	0.483
Short response time.	10	4.300	0.483
Adaptability.	11	4.273	0.467
Error tolerance.	11	4.273	0.467

During the second round of the survey it was proposed to add a statement specific to integration such as interoperability. Among services, semantic interoperability represents the core idea behind SWS-based integration architectures and was ranked second among the functional requirements, right after clear interface definitions. We believe that interoperability can be rated as an implicit feature of integration architectures based on SWSs.

The academic and industrial viewpoints are in general accord with regard to the qualitative requirements that integration architectures must fulfill. However, robustness and availability were seen as different representations of the same issue by the two groups of

respondents. On the one hand, it is interesting that performance, scalability and short response time were rated higher by respondents with industrial backgrounds. Extensibility, modularity of the architecture and adherence to standards, on the other hand, were more important for respondents with academic backgrounds.

Table 36 lists the most controversial qualitative requirements that integration architectures must fulfill comparing the two groups of respondents.

Table 36 Most controversial qualitative requirements that integration architectures must fulfill comparing the two groups of respondents.

	Mean	
	Academia	Industry
Availability.	4.263	4.818
Performance.	4.158	4.545
Modular architecture.	4.316	4.000
Adherence to standards.	4.316	4.000
Robustness.	4.579	4.273

5.4.2.3 Differences between Internal and External Integration Architectures

The respondents rated 37 statements with regard to differences between internal and external integration architectures. Table 37 lists the most important differences.

Table 37 Most important differences between internal and external integration architectures.

	N	Mean	Std. Dev.
Access policies.	28	3.964	1.071
Number of clients.	26	3.885	1.071
Semantic standards versus home-grown semantics.	29	3.862	1.125
Requirement of security mechanisms.	29	3.862	0.915
Necessity of agreement.	29	3.793	0.978
Heterogeneity of business processes.	28	3.786	1.031
Use of proprietary versus public elements.	26	3.692	0.970
Flexibility in customizing services.	29	3.690	1.039
Requirement of standardization.	29	3.655	0.897
Requirement of trust mechanisms.	29	3.655	0.857

While Table 38 lists the most agreed differences between internal and external integration architectures, Table 39 lists the most controversial differences.

Table 38 Most agreed differences between internal and external integration architectures.

	N	Mean	Std. Dev.
System complexity.	28	3.500	0.793
Requirement of trust mechanisms.	29	3.655	0.857
Requirement of standardization.	29	3.655	0.897
Requirement of security mechanisms.	29	3.862	0.915
Use of event- versus message-based communication.	26	3.077	0.935

Table 39 Most controversial differences between internal and external integration architectures.

	N	Mean	Std. Dev.
Integration of legacy systems.	30	3.100	1.373
Scalability.	29	3.138	1.274
Ontology mapping efforts.	29	3.241	1.272
Support of business rules.	29	3.000	1.254
Requirement of stability.	27	3.296	1.203

According to the study's results, the requirements for internal and external integration architectures differ particularly with regard to access policies, the number of clients, security mechanisms and the general necessity of agreement. There are also issues such as semantic standards versus home-grown semantics and use of proprietary versus public elements.

There were no significant differences in the ratings of the two groups of respondents.

5.4.3 Expectations

The expected effects were grouped according to their planning and control level. At the macro level, the focus was on long-term or strategic effects and at the micro level, the focus was on day-to-day or operational effects. In sections 5.4.3.1 and 5.4.3.2, we analyze the positive and negative effects of integration architectures based on SWSs on the macro level. In sections 5.4.3.3 and 5.4.3.4, the focus is on the micro level.

5.4.3.1 Positive Effects at the Macro Level

The respondents rated 34 statements related to positive effects of using SWS-based integration architectures at the macro level.

Table 40 lists the most important positive effects.

Table 40 Most important positive effects of using SWS-based integration architectures at the macro level.

	N	Mean	Std. Dev.
Reduced interoperability problems.	29	4.069	0.842
Higher flexibility.	28	4.036	0.922
Conformance to standards.	29	3.966	1.052
Higher adaptability.	29	3.897	0.860
Better understanding of processes.	28	3.893	1.166
Improved business process integration.	29	3.862	0.915
Increased number of possible cooperations.	29	3.862	0.990
New business models.	29	3.759	1.057
Higher business efficiency and agility.	29	3.724	1.162
Agile collaborations.	29	3.724	1.032

While Table 41 lists the most agreed positive effects of using integration architectures based on SWSs at the macro level, Table 42 lists the most controversial positive effects.

Table 41 Most agreed positive effects of using SWS-based integration architectures at the macro level.

	N	Mean	Std. Dev.
Reduced interoperability problems.	29	4.069	0.842
Higher adaptability.	29	3.897	0.860
Good predictability of advancements.	28	3.143	0.891
Improved business process integration.	29	3.862	0.915
Higher flexibility.	28	4.036	0.922

Table 42 Most controversial positive effects of using SWS-based integration architectures at the macro level.

	N	Mean	Std. Dev.
Improved user experience.	28	3.321	1.389
Facilitated corporate planning.	28	3.357	1.224
Shift of costs from integrators to service providers.	29	3.345	1.203
Higher robustness.	28	3.214	1.197
Lower vendor power.	28	3.143	1.177

Reduced interoperability problems, improved business process integration and higher flexibility and adaptability are present in both the list of the most important positive effects of using SWS-based integration architectures at the macro level and the list of the most agreed statements in this regard. Respondents also rated conformance to standards, better understanding of processes and increased number of possible cooperations as important. Finally, respondents sense new business models, higher business efficiency and agility, and agile collaborations as positive outcomes of the use of SWS-based integration architectures.

While Table 43 lists the most important positive effects of using integration architectures based on SWS at the macro level from an academic perspective, Table 44 lists the most important positive effects from an industrial perspective.

Table 43 Most important positive effects of using SWS-based integration architectures at the macro level from an academic perspective.

	N	Mean	Std. Dev.
Reduced interoperability problems.	18	4.222	0.943
Increased number of possible cooperations.	18	4.056	0.725
Improved business process integration.	18	3.944	0.725
Better understanding of processes.	18	3.944	1.110
Higher adaptability.	18	3.944	0.802
Higher flexibility.	18	3.889	0.900
Agile collaborations.	18	3.833	0.924
Conformance to standards.	18	3.833	1.150
Higher business efficiency and agility.	18	3.778	1.114
Agile business processes.	18	3.722	1.074

Table 44 Most important positive effects of using SWS-based integration architectures at the macro level from an industrial perspective.

	N	Mean	Std. Dev.
Higher flexibility.	10	4.300	0.949
Conformance to standards.	11	4.182	0.874
New business models.	11	3.909	1.044
Reduced time to market.	11	3.909	0.944
Reduced interoperability problems.	11	3.818	0.603
Higher adaptability.	11	3.818	0.982
Better understanding of processes.	10	3.800	1.317
Improved business process integration.	11	3.727	1.191
Improved value creation capability.	11	3.727	1.348
Higher business efficiency and agility.	11	3.636	1.286

With average ratings greater than 4, respondents with academic backgrounds perceive reduced interoperability problems and the increased number of possible cooperations as the most important positive effects at the macro level. Respondents with industrial backgrounds, in comparison, sense higher flexibility and conformance to standards as most relevant. The expectation of a reduced time to market is a further important, positive effect from an industrial perspective. Both groups attached particular importance to statements related to efficiency and agility.

While lower vendor power and increased throughput are important positive effects for respondents with industrial backgrounds, these concerns are not shared by participants with academic backgrounds. The situation is the other way round with regard to the compliance with business and legal requirements as well as the facilitation of payment transactions.

Table 45 lists the most controversial positive effects of using SWS-based integration architectures at the macro level comparing the two groups of respondents.

Table 45 Most controversial positive effects of using SWS-based integration architectures at the macro level comparing the two groups of respondents.

	Mean	
	Academia	Industry
Lower vendor power.	2.889	3.600
Compliance with business and legal requirements.	3.556	2.889
Increased through-put.	2.882	3.400
Increased number of possible cooperations.	4.056	3.545
Facilitated payment transactions.	3.235	2.778

5.4.3.2 Negative Effects at the Macro Level

The respondents rated 20 statements related to negative effects of using SWS-based integration architectures at the macro level. Table 46 lists the most important negative effects.

Table 46 Most important negative effects of using SWS-based integration architectures at the macro level.

	N	Mean	Std. Dev.
Unfulfilled promises.	26	3.769	0.992
Dependency on good services.	26	3.346	1.231
Unpredictability of results.	28	3.286	1.013
Higher manpower costs.	27	3.259	1.163
Higher development costs.	28	3.250	1.041
Stakeholder dissatisfaction.	26	3.192	0.981
No vendor support.	27	3.111	1.155
Excessive growth of services.	27	3.074	0.917
High investments and little gain.	27	3.074	1.072
Focus on short term partnerships.	28	3.000	1.054

While Table 47 lists the most agreed negative effects of using integration architectures based on SWSs at the macro level, Table 48 lists the most controversial negative effects.

Table 47 Most agreed negative effects of using SWS-based integration architectures at the macro level.

	N	Mean	Std. Dev.
Abundance of transparency.	24	2.667	0.816
Continuous adaptation to changing standards.	28	2.893	0.832
Excessive growth of services.	27	3.074	0.917
Creation of non-viable cooperations.	23	2.870	0.920
Limited business value.	28	2.964	0.962

Table 48 Most controversial negative effects of using SWS-based integration architectures at the macro level.

	N	Mean	Std. Dev.
Dependency on good services.	26	3.346	1.231
Neglect of non-technical aspects.	27	2.667	1.209
Asset erosion.	23	2.913	1.203
Higher maintenance costs.	27	2.815	1.178
Increased divide between business and IT stakeholders.	25	2.760	1.165

The only widely agreed negative effect of using SWS-based integration architectures at the macro level is related to the fear of being confronted with unfulfilled promises. All others exhibit an average rating only slightly above 3. Interestingly, dependency on good services is the second most important negative effect of using integration architectures based on SWSs but also the most controversial one.

While Table 49 lists the most important negative effects of using integration architectures based on SWS at the macro level from an academic perspective, Table 50 lists the most important negative effects from an industrial perspective.

Table 49 Most important negative effects of using SWS-based integration architectures at the macro level from an academic perspective.

	N	Mean	Std. Dev.
Unfulfilled promises.	17	3.706	1.047
Unpredictability of results.	18	3.500	0.985
Dependency on good services.	17	3.471	1.007
Stakeholder dissatisfaction.	17	3.176	1.074
Higher manpower costs.	18	3.167	1.295
Higher development costs.	18	3.167	1.150
High investments and little gain.	18	3.167	1.098
Low cost effectiveness.	17	3.059	1.029
Continuous adaptation to changing standards.	18	3.056	0.873
Increased divide between business and IT stakeholders.	16	3.000	1.033

Table 50 Most important negative effects of using SWS-based integration architectures at the macro level from an industrial perspective.

	N	Mean	Std. Dev.
Unfulfilled promises.	9	3.889	0.928
Focus on short term partnerships.	10	3.500	0.850
Higher manpower costs.	9	3.444	0.882
No vendor support.	9	3.444	1.130
Excessive growth of services.	9	3.444	1.236
Asset erosion.	9	3.444	1.236
Higher development costs.	10	3.400	0.843
Stakeholder dissatisfaction.	9	3.222	0.833
Dependency on good services.	9	3.111	1.616
Limited business value.	10	3.000	0.816

Separating the ratings of the two groups of respondents gives greater insight into expected negative effects of using SWS-based integration architectures at the macro level. On the one hand, respondents with academic backgrounds see among unfulfilled promises, the unpredictability of results and the dependency on good services as negative effects. On the other hand, respondents with industrial backgrounds expect among unfulfilled promises, the focus on short-term partnerships, higher manpower costs, no vendor support, excessive growth of services and asset erosion. Asset erosion, together with the focus on short-term

partnerships, is also one of the most controversial aspects, comparing the average ratings of the two groups of respondents.

Table 51 lists the most controversial negative effects of using SWS-based integration architectures at the macro level comparing the two groups of respondents.

Table 51 Most controversial negative effects of using SWS-based integration architectures at the macro level comparing the two groups of respondents.

	Mean	
	Academia	Industry
Asset erosion.	2.571	3.444
Focus on short term partnerships.	2.722	3.500
Low cost effectiveness.	3.059	2.333
Neglection of non-technical aspects.	2.889	2.222
Increased divide between business and IT stakeholders.	3.000	2.333

5.4.3.3 Positive Effects at the Micro Level

The respondents rated 35 statements related to positive effects of using SWS-based integration architectures at the micro level. Table 52 lists the most important positive effects.

Table 52 Most important positive effects of using SWS-based integration architectures at the micro level.

	N	Mean	Std. Dev.
Reuse of system components.	26	4.154	0.967
Improved service discovery capability.	27	4.148	0.818
Better integration quality.	27	3.889	1.188
Easier service integration.	26	3.808	0.801
Facilitated construction of large systems.	26	3.808	0.895
Easier data access.	26	3.808	0.939
Availability of unambiguous descriptions.	26	3.769	1.243
Faster reorganization.	26	3.769	1.142
Automation of time-consuming tasks.	26	3.731	0.919
Improved correctness and consistency.	26	3.731	1.079

While Table 53 lists the most agreed positive effects of using integration architectures based on SWSs at the micro level, Table 54 lists the most controversial positive effects.

Table 53 Most agreed positive effects of using SWS-based integration architectures at the micro level.

	N	Mean	Std. Dev.
Clearer structure of interfaces.	26	3.654	0.797
Easier service integration.	26	3.808	0.801
Improved service discovery capability.	27	4.148	0.818
Reduction of security problems.	27	2.481	0.893
Facilitated construction of large systems.	26	3.808	0.895

Table 54 Most controversial positive effects of using SWS-based integration architectures at the micro level.

	N	Mean	Std. Dev.
More flexible supply chain management.	25	3.520	1.388
Shorter lead times.	25	3.240	1.363
Improved decision support.	25	3.520	1.262
Increased availability of services.	25	3.440	1.261
Availability of unambiguous descriptions.	26	3.769	1.243

With average ratings significantly greater than 4, reuse of system components and improved service discovery capability represent the most important positive effects at the micro level. Furthermore, better integration quality, easier service integration and data access, and facilitated construction of large systems are expected to be positive effects of using SWS-based integration architectures at the micro level.

The facilitated construction of large systems is also among the aspects, most respondents agreed on. Quite controversial was the debate about a more flexible supply chain management and shorter lead times.

While Table 55 lists the most important positive effects of using integration architectures based on SWS at the micro level from an academic perspective, Table 56 lists the most important positive effects from an industrial perspective.

Table 55 Most important positive effects of using SWS-based integration architectures at the micro level from an academic perspective.

	N	Mean	Std. Dev.
Improved service discovery capability.	17	4.235	0.752
Reuse of system components.	17	4.118	0.993
Better integration quality.	17	4.059	1.029
Facilitated construction of large systems.	16	3.938	0.929
Easier data access.	17	3.882	0.928
Improved service documentation.	16	3.875	0.957
Easier service integration.	16	3.813	0.911
Standardized service composition.	17	3.765	1.091
Improved correctness and consistency.	16	3.750	1.000
Availability of unambiguous descriptions.	16	3.750	1.238

Table 56 Most important positive effects of using SWS-based integration architectures at the micro level from an industrial perspective.

	N	Mean	Std. Dev.
Reuse of system components.	9	4.222	0.972
Improved service discovery capability.	10	4.000	0.943
Faster reorganization.	10	4.000	1.155
Reduced time to deployment.	10	4.000	0.816
Seamless interoperability.	10	3.900	1.101
More flexible supply chain management.	10	3.900	1.101
Shorter lead times.	10	3.900	1.101
Increased availability of services.	9	3.889	1.167
Easier service integration.	10	3.800	0.632
Availability of unambiguous descriptions.	10	3.800	1.317

While from an industrial point of view, faster reorganization, reduced time to deployment and shorter lead times play key roles, better integration quality, easier data access and improved service documentation are the most important aspects with regard to positive effects at the micro level when using SWS-based integration architectures from an academic perspective. Respondents with academic backgrounds further expect clear methodologies for maintenance to be a positive effect at the micro level, while respondents with industrial backgrounds disagree. Instead, they see positive effects in shorter lead times and higher performance.

Table 57 lists the most controversial positive effects of using SWS-based integration architectures at the micro level comparing the two groups of respondents.

Table 57 Most controversial positive effects of using SWS-based integration architectures at the micro level comparing the two groups of respondents.

	Mean	
	Academia	Industry
Shorter lead times.	2.800	3.900
Clear methodologies for maintenance.	3.588	2.600
Higher performance.	2.500	3.300
Increased availability of services.	3.188	3.889
More flexible supply chain management.	3.267	3.900

5.4.3.4 Negative Effects at the Micro Level

The respondents rated 24 statements related to negative effects of using SWS-based integration architectures at the micro level. Table 58 lists the most important negative effects.

Table 58 Most important negative effects of using SWS-based integration architectures at the micro level.

	N	Mean	Std. Dev.
High knowledge requirements.	26	4.231	0.863
High initial setup efforts.	27	4.037	0.759
Immature tools.	26	3.962	1.076
High modeling efforts.	26	3.923	0.977
Additional documentation efforts.	27	3.778	0.934
Increased complexity.	25	3.680	1.215
Lack of ontological commitment.	24	3.625	1.173
Low scalability.	25	3.440	1.121
System verification problems.	26	3.423	1.332
Increased maintenance workload.	25	3.360	1.036

While Table 59 lists the most agreed negative effects of using integration architectures based on SWSs at the micro level, Table 60 lists the most controversial negative effects.

Table 59 Most agreed negative effects of using SWS-based integration architectures at the micro level.

	N	Mean	Std. Dev.
High initial setup efforts.	27	4.037	0.759
Low reliability.	26	2.654	0.846
High knowledge requirements.	26	4.231	0.863
Long response times.	27	3.296	0.912
Additional documentation efforts.	27	3.778	0.934

Table 60 Most controversial negative effects of using SWS-based integration architectures at the micro level.

	N	Mean	Std. Dev.
Lack of reusability.	26	2.115	1.336
System verification problems.	26	3.423	1.332
Complex service composition.	25	3.280	1.242
Increased error tracking workload.	23	3.304	1.222
Distraction from important tasks.	24	3.000	1.216

The most important negative effects of using integration architectures based on SWSs are high knowledge requirements, initial setup efforts as well as modeling efforts and immature tools. The high knowledge requirements and setup efforts are, together with additional documentation efforts, not only among the most important negative effects but also among the most agreed statements with regard to negative effects of using SWS-based integration architectures at the micro level. Further negative effects are increased complexity, lack of ontological commitment and low scalability.

One expert's comment put it in a nutshell: the options particularly for complex systems are to either pay costs up front and live with a higher perceived complexity or pay much more at a later point in time. The expert argues that if semantic and representational problems are not addressed, errors and unintended behavior will be the consequence.

While Table 61 lists the most important negative effects of using integration architectures based on SWS at the macro level from an academic perspective, Table 62 lists the most important negative effects from an industrial perspective.

Table 61 Most important negative effects of using SWS-based integration architectures at the micro level from an academic perspective.

	N	Mean	Std. Dev.
High knowledge requirements.	17	4.412	0.618
High initial setup efforts.	17	4.118	0.781
High modeling efforts.	17	4.118	0.697
Immature tools.	17	4.000	1.118
Lack of ontological commitment.	15	3.867	1.125
Increased complexity.	16	3.813	1.276
Long response times.	17	3.529	0.874
Additional documentation efforts.	17	3.529	1.068
Increased maintenance workload.	15	3.467	1.060
Distraction from important tasks.	14	3.429	1.284

Table 62 Most important negative effects of using SWS-based integration architectures at the micro level from an industrial perspective.

	N	Mean	Std. Dev.
Additional documentation efforts.	10	4.200	0.422
High initial setup efforts.	10	3.900	0.738
High knowledge requirements.	9	3.889	1.167
Immature tools.	9	3.889	1.054
System verification problems.	10	3.800	1.033
Increased error tracking workload.	9	3.778	1.202
Complex service composition.	9	3.667	1.323
High modeling efforts.	9	3.556	1.333
Low scalability.	9	3.556	1.130
Lack of security features.	9	3.556	1.014

The academic and industrial viewpoints are quite in accord with regard to the expected negative effects of using integration architectures based on SWSs at the micro level. However, from an academic perspective the distraction from important tasks also plays an important role. Conversely, the increased error tracking workload and lack of trust features are negative effects from an industrial perspective.

Table 63 lists the most controversial negative effects of using SWS-based integration architectures at the micro level comparing the two groups of respondents.

Table 63 Most controversial negative effects of using SWS-based integration architectures at the micro level comparing the two groups of respondents.

	Mean	
	Academia	Industry
Distraction from important tasks.	3.429	2.400
Increased error tracking workload.	3.000	3.778
Lack of trust features.	2.647	3.333
Lack of reusability.	1.882	2.556
Additional documentation efforts.	3.529	4.200

5.4.4 Roadmap

In section 5.4.4.1, we describe the results related to challenges SWS research faces today, while section 5.4.4.2 deals with achievements expected within the next five years. In section 5.4.4.3, we discuss the problems of current integration architectures that could probably be solved with SWSs. Finally, in section 5.4.4.4, we briefly summarize the responses related to real-world case studies in which SWS-based integration architectures have been used.

5.4.4.1 SWS Research Challenges

The respondents rated 35 statements related to challenges SWS research is facing today. Table 64 lists the most important challenges.

Table 64 Most important challenges SWS research is facing today.

	N	Mean	Std. Dev.
Immature technologies.	28	4.071	0.940
Grounding research vision into reality.	28	4.000	1.089
Lack of skilled developers.	27	4.000	1.074
High complexity.	28	3.964	0.962
Lack of effective tools.	28	3.964	0.922
Integration into integrated development environments.	26	3.962	0.774
Proof of cost effectiveness.	28	3.929	1.052
Market and vendor apathy.	28	3.893	1.066
Lack of suitable test environments.	28	3.857	1.079

While Table 65 lists the most agreed challenges SWS research is facing today, Table 66 lists the most controversial challenges.

Table 65 Most agreed challenges SWS research is facing today.

	N	Mean	Std. Dev.
Integration into integrated development environments.	26	3.962	0.774
Lack of a visual knowledge architecture.	27	3.333	0.784
Slow process of adoption.	28	3.643	0.826
Poor comparability of models.	25	3.520	0.918
Lack of effective tools.	28	3.964	0.922

Table 66 Most controversial challenges SWS research is facing today.

	N	Mean	Std. Dev.
Unsatisfactory service discovery capability.	28	3.286	1.384
Agreement on standards.	28	3.643	1.311
Limited coordination with related research communities.	27	3.593	1.309
Scalability of reasoning.	28	3.464	1.261
Unsatisfactory inconsistency detection.	28	3.286	1.243

The most important SWS research challenges, with average ratings equal to or greater than 4, are immature technologies, the grounding of the research vision in reality and the lack of skilled developers. The high complexity, the lack of effective tools and the integration into integrated environments are further challenges. The latter two are also among the most agreed statements with regard to challenges SWS research is facing today. Finally, the proof of cost effectiveness must be brought forward, market and vendor apathy must be attacked, and the lack of suitable test environments must be eradicated. The high rating of the unavailability of convincing case studies is in line with the responses to question 15.

One expert commented during the second survey round that one of the biggest challenges for SWS research is to show that the technologies work in practice. This is in line with challenges ranked in the top third, such as the grounding of the research vision in reality, the proof of cost effectiveness, and market and vendor apathy. Furthermore, the present circumstances were compared with the situation 30 years ago when the logic programming community was trying to sell programming tools based on Prolog¹. It was alleged that SWSs could only capture small niche markets in specific application domains. This

¹ Prolog is a logic programming language often associated with artificial intelligence and computational linguistics.

concern, however, could not be brought in line with the overall impression gathered from the results of our study.

While Table 67 lists the most important challenges SWS research is facing today from an academic perspective, Table 68 lists the most important challenges from an industrial perspective.

Table 67 Most important challenges SWS research is facing today from an academic perspective.

	N	Mean	Std. Dev.
Grounding research vision into reality.	18	4.111	0.963
Integration into a broader set of technologies.	18	4.111	0.758
Unavailability of convincing case studies.	18	4.056	0.802
Lack of skilled developers.	18	4.056	1.056
Immature technologies.	18	4.000	1.138
Lack of effective tools.	18	3.944	1.056
High complexity.	18	3.889	1.079
Unclear benefit over traditional approaches.	18	3.889	1.023
Proof of cost effectiveness.	18	3.889	1.231
Integration into integrated development environments.	16	3.875	0.885

Table 68 Most important challenges SWS research is facing today from an industrial perspective.

	N	Mean	Std. Dev.
Market and vendor apathy.	10	4.500	0.527
Lack of semantically-enabled services.	10	4.400	0.699
Immature technologies.	10	4.200	0.422
High complexity.	10	4.100	0.738
Integration into integrated development environments.	10	4.100	0.568
Agreement on standards.	10	4.100	1.197
Lack of effective tools.	10	4.000	0.667
Proof of cost effectiveness.	10	4.000	0.667
Scalability of ontologies.	10	4.000	0.943
Lack of suitable test environments.	10	3.900	0.994

From the industrial perspective, the most important challenges are market and vendor apathy and the lack of semantically enabled services. Both statements are not considered quite important by respondents with academic backgrounds. From the academic

perspective, integration into a broader set of technologies is worth mentioning. Respondents with industrial backgrounds perceive the agreement on standards, scalability of ontologies and the lack of suitable test environments as particularly important research challenges. All these statements are not among the most important ones for respondents with academic backgrounds. Instead, they sense challenges in the grounding of the research vision in reality and the unclear benefit of SWSs over traditional approaches.

Table 69 lists the most controversial challenges SWS research is facing today comparing the two groups of respondents.

Table 69 Most controversial challenges SWS research is facing today comparing the two groups of respondents.

	Mean	
	Academia	Industry
Unsatisfactory trust and reputation features.	2.611	3.900
Lack of semantically-enabled services.	3.375	4.400
Market and vendor apathy.	3.556	4.500
Integration into a broader set of technologies.	4.111	3.300
Unsatisfactory inconsistency detection.	3.000	3.800

5.4.4.2 Near- and Medium-Term Achievements of SWS Research

The respondents rated 28 statements related to achievements of SWS research to be made within the next five years. Table 70 lists the achievements most likely to be made within the next five years.

Table 70 Achievements of SWS research to be made most likely within the next five years.

	N	Mean	Std. Dev.
Availability of convincing cases studies.	28	3.964	0.793
Improved service discovery capability.	28	3.929	0.813
Availability of suitable languages.	28	3.750	0.967
Availability of open workbenches.	28	3.750	0.967
Availability of integrated methodologies.	28	3.750	0.887
Improved reasoning capability.	28	3.750	0.752
Availability of advanced ontology management systems.	27	3.741	0.944
Limited-scale industrial application.	28	3.714	0.937
Availability of open system architectures.	26	3.692	0.970
Availability of open toolsets.	28	3.679	0.983

While Table 71 lists the most agreed achievements of SWS research to be made within the next five years, Table 72 lists the most controversial achievements.

Table 71 Most agreed achievements of SWS research to be made within the next five years.

	N	Mean	Std. Dev.
Improved reasoning capability.	28	3.750	0.752
Availability of effective tools.	27	3.667	0.784
Availability of convincing cases studies.	28	3.964	0.793
Improved service discovery capability.	28	3.929	0.813
Emergence of model generated workplaces.	25	3.480	0.823

Table 72 Most controversial achievements of SWS research to be made within the next five years.

	N	Mean	Std. Dev.
Agreement on standards.	28	3.607	1.257
Availability of semantically-enabled services.	28	3.679	1.156
Maturity of used technologies.	28	3.286	1.084
Availability of efficient ontologies.	25	3.320	1.069
Higher awareness in industry.	28	3.643	1.062

Respondents see the availability of convincing case studies and the improved service discovery capability as the key achievements of SWS research to be made within the next five years. Both are also among the most agreed statements in this regard. The availability of suitable languages, open workbenches and integrated methodologies are also among the

most important achievements. Improved reasoning capability is the most agreed aspect which is also among the most important achievements in this regard. Finally, advanced ontology management systems, open system architectures and open toolset will be available, according to the respondents. Interestingly, limited-scale industrial application is also in the achievements to be made within the next five years.

However, respondents were discordant with regard to the forecast of the achievements with regard to the agreement on standards, the availability of semantically-enabled services and the general maturity of the technologies used.

While Table 73 lists achievements of SWS research to be made most likely within the next five years from an academic perspective, Table 74 lists the achievements from an industrial perspective.

Table 73 Achievements of SWS research to be made most likely within the next five years from an academic perspective.

	N	Mean	Std. Dev.
Improved service discovery capability.	18	3.944	0.938
Availability of integrated methodologies.	18	3.889	1.023
Availability of convincing cases studies.	18	3.833	0.786
Availability of open workbenches.	18	3.833	1.150
Availability of advanced ontology management systems.	17	3.824	0.951
Improved reasoning capability.	18	3.778	0.878
Availability of open toolsets.	18	3.722	1.018
Availability of effective tools.	17	3.706	0.686
Limited-scale industrial application.	18	3.667	0.907
Agreement on standards.	18	3.667	1.237

One expert doubted that SWS-based integration architectures will support a large degree of automation in design and mediation-related activities in the near-term. He argued that in the initial phase of SWS adoption and deployment, it is neither necessary nor technically possible to support a large degree of automation. In the near- to medium-term, from his point of view, human involvement will remain necessary and desirable for making final choices. This estimation is definitely in line with the results of our study.

Table 74 Achievements of SWS research to be made most likely within the next five years from an industrial perspective.

	N	Mean	Std. Dev.
Availability of convincing cases studies.	10	4.200	0.789
Higher awareness in industry.	10	4.100	1.197
Availability of suitable languages.	10	4.000	0.471
Improved service choreography capability.	10	4.000	0.667
Improved service orchestration capability.	10	4.000	0.000
Improved service discovery capability.	10	3.900	0.568
Limited-scale industrial application.	10	3.800	1.033
Availability of open system architectures.	10	3.800	0.632
Availability of semantically-enabled services.	10	3.800	1.033
Improved reasoning capability.	10	3.700	0.483

Table 75 lists the most controversial statements related to achievements in SWS research within the next five years comparing the two groups of respondents.

Table 75 Most controversial statements related to achievements in SWS research within the next five years comparing the two groups of respondents.

	Mean	
	Academia	Industry
Improved service orchestration capability.	3.167	4.000
Higher awareness in industry.	3.389	4.100
Split-up into independent communities.	2.688	3.375
Improved service choreography capability.	3.389	4.000
Availability of integrated methodologies.	3.889	3.500

5.4.4.3 Problems of Current Integration Architectures

The respondents rated 27 statements related to problems of current integration architectures that can be solved with SWSs. Table 76 lists the most important problems.

Table 76 Most important problems of current integration architectures that can be solved with SWSs.

	N	Mean	Std. Dev.
Automation of service discovery.	28	4.036	1.071
Lack of semantic interoperability.	28	4.000	1.054
Lack of semantic service descriptions.	28	3.964	0.922
Data mapping and sharing.	28	3.964	0.922
Matching of services from heterogeneous sources.	28	3.929	1.016
Reuse of components.	27	3.741	0.984
Limited flexibility.	26	3.692	1.158
Replacement of hardwired adapters and converters.	28	3.643	1.062
Format and data type conversion.	28	3.643	1.162
Manageability.	26	3.538	1.140

While Table 77 lists the most agreed problems of current integration architectures that can be solved with SWSs, Table 78 lists the most controversial problems.

Table 77 Most agreed problems of current integration architectures that can be solved with SWSs.

	N	Mean	Std. Dev.
Difficulty of conformance testing.	28	3.500	0.839
Changeability of large systems.	28	3.500	0.839
Understanding of interfaces.	28	3.071	0.900
Monitoring.	28	3.321	0.905
Lack of semantic service descriptions.	28	3.964	0.922

Table 78 Most controversial problems of current integration architectures that can be solved with SWSs.

	N	Mean	Std. Dev.
Agreement on standards.	28	3.393	1.397
Automation of service composition.	28	3.179	1.335
Consistency and completeness proofs.	28	3.464	1.232
Integration of legacy systems.	28	3.214	1.228
Subjectivity of semantic descriptions.	28	3.107	1.227

The respondents revealed many aspects of current integration architectures that could be improved by using SWSs. The automation of service discovery is the most important issue that could be affected positively by the application of SWSs in integration architectures. Respondents also see a potential in SWSs with respect to the lack of semantic

interoperability and semantic service descriptions. Furthermore, data mapping and sharing, matching of services from heterogeneous sources, and reuse of components might be facilitated. The statement about the lack of semantic service descriptions is also among the most agreed statements concerning problems of current integration architectures that could be solved by using SWS technologies.

According to one expert's comment, the SWS approach departs considerably from existing development paradigms without providing any clear and proven additional benefits to justify its use. The expert was also apprehensive about the fact that SWSs are confusing documentations created for human consumption with metadata. Many of the examples SWS researchers use are rather artificial because they try to capture things that are intended for human consumption as metadata. Nevertheless, because the availability of convincing case studies leads the list of the most important achievements in SWS research within the next five years, it seems agreed that it is possible and of major importance to make convincing case studies available.

While Table 79 lists the most important problems of current integration architectures that can be solved with SWSs from an academic perspective, Table 80 lists the most important problems from an industrial perspective.

Table 79 Most important problems of current integration architectures that can be solved with SWSs from an academic perspective.

	N	Mean	Std. Dev.
Lack of semantic interoperability.	18	4.222	0.878
Automation of service discovery.	18	4.056	1.162
Matching of services from heterogeneous sources.	18	4.000	0.970
Data mapping and sharing.	18	4.000	0.907
Lack of semantic service descriptions.	18	3.944	0.938
Reuse of components.	17	3.824	0.883
Limited flexibility.	16	3.813	1.167
Difficulty of configuration.	18	3.722	1.227
Replacement of hardwired adapters and converters.	18	3.667	1.138
Format and data type conversion.	18	3.667	1.237

Table 80 Most important problems of current integration architectures that can be solved with SWSs from an industrial perspective.

	N	Mean	Std. Dev.
Automation of service discovery.	10	4.000	0.943
Lack of semantic service descriptions.	10	4.000	0.943
Agreement on standards.	10	4.000	1.414
Data mapping and sharing.	10	3.900	0.994
Matching of services from heterogeneous sources.	10	3.800	1.135
Lack of semantic interoperability.	10	3.600	1.265
Reuse of components.	10	3.600	1.174
Replacement of hardwired adapters and converters.	10	3.600	0.966
Format and data type conversion.	10	3.600	1.075
Manageability.	10	3.600	1.075

The academic and industrial viewpoints are quite in accord with regard to the current problems of integration architectures that could be solved by using SWSs. From an industrial perspective, the agreement on standards represents a problem of the kind discussed. Respondents with academic backgrounds did not state explicitly that they agree with the statement that the agreement on standards is a current problem that could be solved by using SWSs.

Table 81 lists the most controversial problems of current integration architectures that can be solved with SWSs comparing the two groups of respondents.

Table 81 Most controversial problems of current integration architectures that can be solved with SWSs comparing the two groups of respondents.

	Mean	
	Academia	Industry
Agreement on standards.	3.056	4.000
Market and vendor apathy.	2.438	3.200
Invalid implementation of standards.	2.889	3.600
Lack of semantic interoperability.	4.222	3.600
Lack of effective tools.	2.625	3.200

5.4.4.4 Real-life Cases

Most respondents who answered question 15, which was related to real-world case studies in which SWS-based integration architectures were used, declared that they did not know of any. Others declared that they knew only of pilots and prototypes in which integration architectures based on SWSs were used in experimental environments. One expert stated explicitly that up to now, SWS technologies have failed to show any practical applications in industry, except for some specific cases around service discovery. According to him, the reason is that many SWS prototypes either do not take scalability into account or try to capture semantics in a purely syntactic manner. No question, scalability plays a key role in integration architectures. It is also among the most important qualitative requirements identified within the scope of this study. It seems plausible that most current integration architectures capture semantics, if they are available, in a purely syntactic manner. The results of the study, however, do not substantiate this assumption specifically.

Some respondents referred to research projects in which SWS technologies were used. The ATHENA and DIP projects, which are described in sections 4.2.1 and 4.2.2, respectively, were mentioned most often in this regard. Other projects such as SWAD-Europe¹, SWWS (Semantic Web-enabled Web Services) and MAPPER² (Model-based Adaptive Product and Process Engineering) were also mentioned. The SWAD-Europe project ran from May 2002 to October 2004 and was aimed at moving Semantic Web technologies into the mainstream of networked computing. By the end of the SWWS project in February 2005, the project partners had developed several prototype applications based on SWSs. MAPPER started in September 2005 and aims at supporting the increased cooperation and collaboration among enterprises during the product life-cycle. Many more projects focusing on SWSs exist but are not listed here.

Finally, respondents also stated that SWS-based integration architectures have already been introduced into bioinformatics. Projects such as BioMOBY³ and EMBRACE⁴ are beginning to adopt SWS-based methods. BioMOBY is a research project intended to generate an architecture for the discovery and distribution of biological data through Web services. The objective of EMBRACE is to draw together a broad group of experts

¹ For more information on SWAD-Europe, see <http://www.w3.org/2001/sw/Europe/>.

² For more information on MAPPER, see <http://mapper.troux.com/>.

³ For more information on BioMOBY, see <http://www.biomoby.org/>.

⁴ For more information on EMBRACE, see <http://www.embracegrid.info/>.

throughout Europe who are involved in the use of IT in the biomolecular sciences to optimize information exploitation.

5.5 Feedback and Comments

The results of the feedback phase and general comments collected during the survey rounds show that the respondents were quite satisfied with the study in general and the survey system in particular. The main question was whether the Delphi study was a worthwhile experience. Respondents paid many compliments on the organization of the study but also expressed much constructive criticism. All of the feedback received was very welcome and will be of great value for future studies. Some experts stated that it was interesting to read the contributions and views of the other survey respondents on a heretofore unexplored topic such as SWSs in e-business.

A few experts indicated that they had problems rating some of the statements. They brought the argument forward that selected statements seemed to be dependent on the implementation of the technology and the environmental circumstances determining the limitations of SWS-based integration architectures. Some stated that it was too early to predict how well SWSs will perform from a business perspective. According to them it was easier to answer questions related to technology and standard issues.

Others stated that there were too many overlapping questions to answer in the first round of the survey and too many different statements to rate in the second round. This suggests that the study was perceived to be a bit long, in particular in the second round. The suggestion was also made that the questions be about more concrete aspects. In general it was agreed that most of the questions were crucial but some experts would have liked them to be somewhat more simplified. It was also stated that there were too many questions and statements to be rated that could not be processed well without having specific use cases in mind. Experts also missed a check box similar to the *No Comment* box indicating that a question or a statement was ambiguous. Others would have liked to see more emphasis on the instruction that one should not answer questions about which one did not have sufficient knowledge.

The respondents further suggested adding a complete overview of all questions and statements in a printable form. Furthermore, a printable overview showing the questions together with the personal responses would have been desired by some experts. Finally, the

participants would have liked to see an overview showing the averaged responses compared with the personal responses in a way that allows finding significant differences easily. This would have been a particularly important point for a possible third round of the survey. Furthermore, some respondents were surprised that the study concluded after only two rounds.

Numerous participants declared in the feedback phase that they were very interested in the conclusions from the study. Furthermore, they stated that the results could definitely be useful and could significantly help to assign neat priorities to SWS requirements. They also stated that the study helped in particular to formulate ideas about the many issues that must be discussed with respect to SWSs.

Table 82 lists the ratings related to the survey system's functionality and usability. The scale ranged from 1 to 5, with 1 representing *Poor* and 5 representing *Excellent*.

Table 82 Functionality and usability of the survey system.

	N	Mean		Std. Dev.
Functionality.	17	4.24	0.136	0.567
Usability.	17	4.06	0.160	0.659

The functionality of the Web-based survey system was rated slightly higher than its usability. With average ratings above 4, however, both aspects seem to be handled quite well by the survey system.

Chapter 6

Discussion

In this chapter the results of the Delphi study are discussed and integrated with information from current literature. In section 6.1, we provide a general vision deduced from the results of the study, in section 6.2, we outline essential business implications and, in section 6.3, we discuss the gap between research trends and industrial needs. We would like to emphasize once again that the Delphi method has the basic limitation that it cannot make complex forecasts with multiple factors. Potential future outcomes are considered as if they have no effect on each other. Because most events and developments are in some way connected to each other, these interdependencies must be kept in mind when interpreting the results.

6.1 General Vision

Based on the results of the Delphi study and on current literature, it seems plausible that SWSs have significant potential as a scalable and cost-effective solution to many integration problems, thereby dealing with one of the key bottlenecks in modern networked society. We expect that there will be a strong demand for Web services and integration technologies as businesses react to the need for a higher level of integration and more agility. Making disparate systems share information cost-effectively is a key problem for companies and represents billions of Euros in technology spending, with a high percentage of worldwide IT budgets dedicated to enterprise integration projects. SWSs promise to move beyond the simple exchange of information, the dominant mechanism for application

integration today, to access application services that are encapsulated in both old and new applications. This means enterprises will not only be able to move information from application to application, but also to create composite applications by combining services found in any number of different local or remote application systems.

The challenges described in [Wieh04] are mostly in line with the ones identified within the scope of our study. In particular with regard to issues such as flexibility, adaptation and maintenance costs, and dependence on software vendors, integration architectures based on SWSs are expected to be superior as compared with traditional integration approaches.

The cost-effectiveness of SWS-based integration architectures still has to be proved. It is agreed that SWS-based integration architectures imply high initial set-up efforts. Most respondents of our study agreed that the abundance of transparency is not among the most important weaknesses of integration architectures based on SWSs.

According to the results of our study, security mechanisms are among the most important functional requirements for B2B integration but not for A2A integration. Particularly respondents with industrial backgrounds perceived unsatisfactory security features as an important weakness of SWS-based integration architectures. Generally, it is not expected that SWSs will reduce security problems significantly.

The results of our Delphi study are ambiguous with regard to complexity and its implications. On the one hand, high system complexity is among the most important weaknesses with regard to integration architectures based on SWSs. On the other hand, it is perceived that SWS-based integration architectures represent a lightweight approach.

6.2 Business Implications

In essence, the use of integration architectures based on SWSs has a variety of profound business impacts. According to the results of our study, SWS-based integration architectures have a dramatic impact on the economics of enterprise integration.

On the one hand, SWS-based architectures allow an increased number of possible cooperations and facilitate the construction of large systems in particular. Adopters are also expected to benefit significantly from higher flexibility and adaptability of their architectures. Service orientation, reuse of existing system components and easy data access are further positive effects of the application of SWS-based integration architectures. The advanced integration architecture reduces interoperability problems and

facilitates enterprise integration by improving selected SWS usage activities such as discovery, mediation and composition. Enhanced process and term definitions as well as explicit definitions of service conditions and functionalities based on ontologies allow for a better understanding of systems.

On the other hand, high start-up costs, software engineers' lack of ontology expertise and unsatisfactory support of change management must be considered with regard to SWS-based integration architectures. Because these architectures have not yet been adopted on an industrial scale, the exercise of caution is advisable. Besides high system complexity, the use of immature technologies and the lack of effective tools pose problems. It is not surprising that improved definitions do not come without labor-intensive specification and modeling tasks. Furthermore, it is clear that there is still lack of agreement on the depth of descriptions. Finally, it is important to find the right balance while satisfying high knowledge requirements and avoiding description overhead.

6.3 Research Trends and Industry Needs

As described in sections 4.1 and 4.2, current literature and the latest international research projects put much effort into closing the gap between academic research and industrial needs.

By comparing the ratings of the functional and qualitative requirements, we found that academic and industrial experts prioritize requirements quite differently. The prioritizations of the top 15 functional and top 10 qualitative requirements are shown in descending order in Table 83 and Table 84, respectively.

Independent from their backgrounds respondents agree that functional requirements such as clear interface definitions and support of service reuse, service life-cycles and workflow definitions are important. There is also accord about the importance of the incorporation of service level agreements and the semantic interoperability among services in general.

However, it is clear that adaptability plays a key role from a research point of view. This proves true not just when looking at qualitative requirements. Functional requirements such as support of configuration and customization of services as well as service adaptation and the support of interface modifications also suggest that. The provision of best practice guidance and recommendations for workflow compositions indicate that research is concerned with user-friendliness and understandability of the architecture.

Table 83 Comparison of the most important functional requirements of the two groups of respondents.

Academia	Industry
Support of service registration.	Clear definitions of interfaces.
Clear definitions of interfaces.	Semantic interoperability among services.
Support of service life-cycles.	Validation of services.
Mediation between services.	Support of service reuse.
Support of service reuse.	Support of static and dynamic relationship management.
Support of workflow definitions.	Support of interface modifications.
Incorporation of service level agreements.	Provision of unambiguous semantics.
Semantic interoperability among services.	Support of the implementation of business models.
Support of configuration and customization of services.	Support of workflow definitions.
Support of service adaptation.	Provision of trust and reputation mechanisms.
Provision of best practice guidance.	Support of service life-cycles.
Support of interface modifications.	Incorporation of service level agreements.
Automatic invocation of services.	Automatic service selection.
Support of service role definitions.	Provision of security features.
Recommendations for workflow compositions.	Support of service registration.

Table 84 Comparison of the most important qualitative requirements of the two groups of respondents.

Academia	Industry
Robustness.	Availability.
Reliability.	Reliability.
Extensibility.	Performance.
Adaptability.	Scalability.
Understandability.	Manageability.
Manageability.	Understandability.
Stability.	User-friendliness.
User-friendliness.	Short response time.
Scalability.	Error tolerance.
Modular architecture.	Adaptability.

Finally, functional requirements such as mediation between services and their automatic invocation point to the importance of the robustness, extensibility and stability of SWS-based integration architectures. The modularity of the architecture is also a key qualitative requirement from an academic perspective.

Besides reliability, availability, performance and error tolerance also play major roles from an industrial perspective. Functional requirements such as the validation of services and the provision of trust and reputation mechanisms as well as security features support this assumption. The support of static and dynamic relationship management and the support of the implementation of business models result in improved scalability and manageability. Finally, automatic service selection and the provision of unambiguous semantics lead to shorter response times and increased understandability, respectively.

Chapter 7

Conclusion

The main focus of this work was on the analysis of the relevance and applicability of SWSs in solving integration problems in today's e-business. The main strengths and weaknesses of SWSs in organizational environments were highlighted and – based on a Delphi study – ideal features of SWS-based integration architectures were exemplified. Our goal was to deduce critical success factors for the future use of SWSs in business environments from literature and the findings of our study. We identified key trends and opportunities and discussed the potential future impact of SWSs. Additionally, major challenges were described and the underlying problems analyzed.

The main purpose of the Delphi study was to collect and quantify the opinions of a clearly defined group of experts about the potential of SWSs in e-business. Of particular interest were deficiencies in SWS frameworks and the requirements that must be fulfilled to enable service automation on a scale required by today's connected enterprises. The results have the potential to help align research efforts and market demands. The survey also made participating decision makers more sensitive to the future role of SWS technologies in general.

The Delphi study revealed many interesting aspects of the relevance and applicability of integration architectures based on SWSs. The study had two rounds and the questionnaires consisted of four parts structured and formalized in a way that allowed various analyses: a SWOT analysis, a requirements analysis, an analysis of expected effects and a technology roadmap.

Internal	Strengths	Weaknesses
	<ul style="list-style-type: none"> Improved service discovery capability. Facilitated interoperability. Facilitated reuse of services. Improved mediation between services. Explicit definitions of conditions and functionalities. 	<ul style="list-style-type: none"> Use of immature technologies. Description overhead. High initial start-up costs. Labor-intensive service specification. Software engineers are not ontology experts.
External	Opportunities	Threats
	<ul style="list-style-type: none"> Availability of business cases. Need for service interoperability. Availability of compliant middleware implementations. Availability of effective tools. Preceding agreement on standards. 	<ul style="list-style-type: none"> Difficulty of describing semantics. Lack of effective tools. Unproven cost effectiveness. Limited consideration of business needs. Unavailability of convincing case studies.

Figure 54 SWOT matrix.

The results of the SWOT analysis are summarized in Figure 54. Enterprises considering the application of SWSs within their integration architectures are constrained to maintain, build and leverage strengths, remedy and stop weaknesses, prioritize and optimize opportunities, and counter threats. The most important strength is, according to the results of our Delphi study, the improved service discovery capability. The use of immature technologies was perceived as the major weakness. For the respondents, the availability of business cases represented the most important opportunity and the difficulty of describing semantics, the key threat.

The requirements analysis helped to identify functional and qualitative requirements that integration architectures must fulfill. The major functional requirements are led by clear definitions of interfaces, semantic interoperability among services and support of service reuse. Reliability, availability and robustness are the key qualitative requirements identified within the scope of our study.

The identified expectations were categorized into two groups: expected effects at the macro level and expected effects at the micro level. At the macro level, respondents expected reduced interoperability problems, higher flexibility and conformance to standards. It was also revealed that respondents with both, academic and industrial backgrounds fear unfulfilled promises as well as the dependency on good services. At the micro level, the reuse of system components, improved service discovery capability and better integration quality were expected. However, respondents were also afraid that using

SWS-based integration architectures could imply high knowledge requirements, high initial setup efforts and immature tools.

The analysis of the data collected within the scope of the technology roadmap led to the point of view that SWSs decidedly have the potential to solve a variety of the problems faced by current integration architectures. Among these problems are the automation of service discovery, the lack of semantic interoperability and the lack of semantic service descriptions. The key challenges for SWS research are the maturity of technologies and the grounding of the research vision in reality. Within the next five years, a greater availability of convincing case studies is expected. Currently, integration architectures based on SWSs are hardly used outside of experimental environments.

We conclude from the results of the study that SWS-based integration architectures are relevant to the integration market and will be applicable within a reasonable time. Without doubt, the majority of the used technologies is not yet satisfactorily mature. To bring SWS-based integration architectures further, the problems with respect to service description have to be addressed in particular. These problems are responsible for many weaknesses such as the high initial start-up efforts.

We showed in this work that with respect to many aspects the picture of integration architectures based on SWSs looks quite different from an academic point of view as compared with an industrial viewpoint. We hope that our study helps to reduce the gap between research trends and industrial needs and, subsequently, to exploit the full potential of e-business. However, there is more that could be done in the area of advanced enterprise integration. For instance, it would be interesting to evaluate specific SWS frameworks such as OWL-S and WSMO with respect to their relevance and applicability for integration architectures. Furthermore, based on the results of this work an ideal integration architecture could be modeled. The priorities could be taken directly from the results presented within the scope of this work.

Bibliography

[AdZi96]

Adler, M.; Ziglio, E.: Gazing into the oracle. Jessica Kingsley Publishers, Bristol 1996.

[Aich05]

Aichholzer, G.: Das ExpertInnen-Delphi: methodische Grundlagen und Anwendungsfeld Technology Foresight. In: *Bogner, A. (ed.)*: Das Experteninterview, Verlag für Sozialwissenschaften, Wiesbaden 2005.

[ARBR04]

Al-Naeem, T.; Rabhi, F.; Benatallah, B.; Ray, P.: Systematic Approaches for Designing B2B Applications. In *International Journal of Electronic Commerce* 9 (2004) 2, pp. 41–70.

[AlSm05]

Alesso, H.P.; Smith, C.F.: Developing Semantic Web Services. A K Peters, Ltd., Wellesley 2005.

[AnHa04]

Antoniou, G.; van Harmelen, F.: A Semantic Web Primer. The MIT Press, Boston 2004.

[ArTC63]

Aronson, E.; Turner, J.; Carlsmith, J.: Communication credibility and communication discrepancy as determinants of opinion change. In: *Journal of abnormal Social Psychology* 67 (1963), pp. 31–36.

[ArLó06]

Arroyo, S.; López-Cobo, J.M.: Describing Web Services with Semantic Metadata. In: *International Journal on Metadata, Semantics and Ontologies* 1 (2006) 1, pp. 76–82.

[BeGV00]

Beck, K.; Glotz, P.; Vogelsang, G.: Die Zukunft des Internet. UVK-Medien, Konstanz 2000.

[BeHL01]

Berners-Lee, T.; Hendler, J.; Lassila, O.: The Semantic Web. In *Scientific American* 279 (2001) 5, pp. 34–43.

[Buss03]

Bussler, C.: B2B Integration: Concepts and Architecture. Springer, Berlin 2003.

[BuFS05]

Bussler, C.; Fensel, D.; Sadeh, N.M.: The Role of Semantic Web Services in Enterprise Application Integration and E-Commerce. In *International Journal of Electronic Commerce* 9 (2005) 2, p. 7.

[CDMP04]

Cabral, L.; Domingue, J.; Motta, E.; Payne, T.; Hakimpour, F.: Approaches to Semantic Web Services: An Overview and Comparison. In: *Proceedings of the European Semantic Web Conference* (2004).

[Come06]

Comergent Technologies Inc.: What's driving eBusiness? White Paper (2006).

[Cuhl00]

Cuhls, K.: Wie kann ein Foresight-Prozess in Deutschland organisiert werden? Gutachten. Friedrich-Ebert-Stiftung, Düsseldorf 2000.

[Dill00]

Dillman, D.A.: Mail and Internet Surveys. John Wiley & Sons Inc., New York 2000.

[DiCS95]

Dillman, D.A.; Clark, J.R.; Sinclair, M.A.: How prenotice letters, stamped return envelopes, and reminder postcards affect mailback response rates for census questionnaires. In: *Survey Methodology* 21 (1995), pp. 1–7.

[Duff93]

Duffield, C.: The Delphi technique: a comparison of results obtaining using two expert panels. In: *International Journal of Nursing Studies* 30 (1993) 3, pp. 227–237.

[EsPW04]

Esplugas-Cuadrado, J.; Preist, C.; Williams, S.: Integration of B2B Logistics Using Semantic Web Services. In: *Proceedings of the 11th International Conference on Artificial Intelligence: Methodology, Systems, and Applications* (2004).

[FRPD05]

Feier, C.; Roman, D.; Polleres, A.; Domingue, J.; Stollberg, M.; Fensel, D.: Towards Intelligent Web Services: Web Service Modeling Ontology (WSMO). In: *Proceedings of the International Conference on Intelligent Computing* (2005).

[Fens03]

Fensel, D.: *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*, 2nd edition, Springer, Berlin 2003.

[FeBu02]

Fensel, D.; Bussler, C.: The Web Service Modeling Framework WSMF. In: *Electronic Commerce Research and Applications*, 1 (2002) 2, pp. 113–137.

[FHLW05]

Fensel, D.; Hendler, J.A.; Lieberman, H.; Wahlster, W.: *Spinning the Semantic Web*. MIT Press, Boston 2005.

[FoBY06]

Foster, J.; Barkus, E.; Yavorsky, C.: *Understanding and Using Advanced Statistics*. Sage Publications, London 2006.

[FrNa06]

Friesen, A.; Namiri, K.: Towards semantic service selection for B2B integration. In: Workshop Proceedings of the 6th International Conference on Web Engineering (2006)

[GrCo98]

Groves, R.M.; Couper M.P.: Nonresponse in household interview surveys. Wiley-Interscience, New York 1998.

[GrBC00]

Grupp, H.; Blind, K.; Cuhls, K.: Die Delphi-Technik in den Sozialwissenschaften Analyse von Meinungsdisparitäten in der Technikbewertung mit der Delphi-Methode. Westdeutscher Verlag, Wiesbaden 2000.

[Häde02]

Häder, M.: Delphi-Befragungen. Westdeutscher Verlag, Wiesbaden 2002.

[HCMO05]

Haller, A.; Cimpian, E.; Mocan, A.; Oren, E.; Bussler, C.: WSMX - A Semantic Service-Oriented Architecture. In: Proceedings of the International Conference on Web Service (2005).

[HaGB05]

Haller, A.; Gómez, J.M.; Bussler, C.: Exposing Semantic Web Service principles in SOA to solve EAI scenarios. In: Proceedings of the 14th International World Wide Web Conference (2005).

[HKMV06]

Haselwanter, T.; Kotinurmi, P.; Moran, M.; Vitvar, T.; Zaremba, M.: WSMX: A Semantic Service Oriented Middleware for B2B Integration. In: Proceedings of the International Conference on Service Oriented Computing (2006).

[HeHR04]

Heinrich, L.J.; Heinzl, A.; Roithmayr, F.: Wirtschaftsinformatik-Lexikon. Oldenbourg, Munich 2004.

[HoWo05]

Hohpe, G.; Woolf, B.: Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley, Boston 2005.

[KiCL05]

Kim, W.; Chung, M.-J.; Lloyd, J.: Automated Outsourcing Partnership Management Using Semantic Web Services. In: Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design (2005).

[KöSc07]

König, O.; Schattenhofer, K.: Gruppendynamik. Carl-Auer-Systeme, Heidelberg 2007.

[KVHR06]

Kotinurmi, P.; Vitvar, T.; Haller, A.; Richardson, R.; Boran, A.: Semantic Web Services Enabled B2B Integration. In: Proceedings of the 2nd International Workshop on Data Engineering Issues in E-Commerce and Services (2006).

[LRPF04]

Lara, R.; Roman, D.; Polleres, A.; Fensel, D.: A Conceptual Comparison of WSMO and OWL-S. In: Proceedings of the European Conference on Web Services (2004).

[LiTu75]

Linstone, H.A.; Turoff, M.: The Delphi Method: Techniques and Applications. Addison-Wesley, London 1975.

[Lint03]

Linthicum, D.S.: Next Generation Application Integration: From Simple Information to Web Services, Addison-Wesley, Boston 2003.

[LKBC06]

Losada, S.; Keczek, D.; Renjamins, R.; Contreras; Corcho, O.; Bas, J.L.; Bellido, S.: How to Make It Faster and at Lower Cost? B2B Integration with Semantic Web Services. In: Proceedings of the 17th International Conference on Database and Expert Systems Applications (2006).

[Mahm04]

Mahmoud, Q.H.: Middleware for Communications. John Wiley & Sons Inc., West Sussex 2004.

[Mane03]

Manes, A.T.: Web Services: A Manager's Guide. Addison-Wesley, Boston 2003.

[MaBe06]

Marks, E.A.; Bell, M.: Service Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology. John Wiley & Sons Inc., New York 2006.

[MPMB04]

Martin, D.; Paolucci, M.; McIlraith, S.; Burstein, M.; McDermott, D.; McGuinness, D.; Parsia, B.; Payne, T.; Sabou, M.; Solanki, M.; Srinivasan, N.; Sycara, K.: Bringing Semantics to Web Services: The OWL-S Approach. In: Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition (2004).

[Masa05]

Masak, D.: Moderne Enterprise Architekturen. Springer, Berlin 2005.

[McSZ01]

McIlraith, S.A.; Son, T.C.; Zeng, H.: Semantic Web Services. In: IEEE Intelligent Systems 16 (2001) 2, pp. 46–52.

[MDCG03]

Motta, E.; Domingue, J.; Cabral, L.; Gaspari, M.: IRS-III: A framework and infrastructure for Semantic Web services. In: Proceeding of the 2nd International Semantic Web Conference (2003).

[Myer02]

Myerson, J.M.: The Complete Book of Middleware. CRC Press, Boca Raton 2002.

[Newc02]

Newcomer, E.: Understanding Web Services: XML, WSDL, SOAP and UDDI. Addison-Wesley, Boston 2002.

[Oasi00]

OASIS Standards Consortium: Universal Description, Discovery and Integration. Technical White Paper (2000).

[Ober06]

Oberle, D.: The Semantic Management of Middleware. Springer, Berlin 2006.

[PoTJ06]

Podobnik, V.; Trzec, K.; Jezic, G.: An Auction-Based Semantic Service Discovery Model for E-Commerce Applications. In: OTM Workshops (2006).

[PoHo04]

Pollock, J.T.; Hodgson, R.H.: Adaptive Information. John Wiley & Sons Inc., New York 2004.

[PEBG05]

Preist, C.; Esplugas-Cuadrado, J.; Battle, S.A.; Grimm, S.; Williams, S.K.: Automated Business-to-Business Integration of a Logistics Supply Chain Using Semantic Web Services Technology. In: Proceedings of the 4th International Semantic Web Conference (2005).

[RiMH85]

Richey, J.S.; Mar, B.W.; Horner, R.R.: The Delphi Technique in Environmental Assessment. In: Journal of Environmental Management 2 (1985) 21, pp. 135–159.

[RoWB91]

Rowe, G.; Wright, G.; Bolger, F.: Delphi: A reevaluation of Research and Theory. In: Technological Forecasting a Social Change 39, pp. 235–251.

[ScDi98]

Schaefer, D.; Dillman, D.A.: Development of standard e-mail methodology: Results of an experiment. In Public Opinion Quarterly 62, pp. 378–397.

[ShEH94]

Shye, S.; Elizur, D.; Hoffman, M.: Introduction to Facet Theory. Sage Publications, Thousand Oaks 1994.

[SMSV04]

Sivashanmugam, K.; Miller, J.; Sheth, A.; Verma, K.: Framework for Semantic Web Process Composition. In: *International Journal of Electronic Commerce* 9 (2004) 2, pp. 71–106.

[ThSH04]

Thome, R.; Schinzer, H.; Hepp, M.: *Electronic Commerce und Electronic Business*. Verlag Vahlen, Munich 2005.

[VGSM05]

Verma, K.; Gomadam, K.; Sheth, A.P.; Miller, J.A.; Wu, Z.: The METEOR-S Approach for Configuring and Executing Dynamic Web Processes. Technical Report (2005).

[Vino02]

Vinoski, S.: Where is Middleware? In: *IEEE Internet Computing* 6 (2002) 2, pp 83–85.

[Whyt01]

Whyte, W.S.: *Enabling eBusiness: Integrating Technologies, Architectures and Applications*. John Wiley & Sons Inc., New York 2001.

[Wieh04]

Wiehler, G.: *Mobility, Security und Web Services*. Publicis Corporate Publishing, Erlangen 2004.

[Wood06]

Wood, J.T.: *Communication in Our Lives*. 4th edition, Thomson Wadsworth, Belmont 2006.

[YZGZ05]

Yang, Z.; Zhang, J.-B.; Gay, R.; Zhuang, L.; Lee, H.M.: Building a Semantic-Rich Service-Oriented Manufacturing Environment. In: *Proceedings of the 6th International Conference on Web Information Systems Engineering* (2005).

[Zura05]

Zurawski, R.: *The Industrial Communication Technology Handbook*. CRC Press, Boca Raton 2005.