Fraunhofer Institut
Arbeitswirtschaft und
Organisation

mediaVision

Joannis Vlachakis
Sascha Rex
Boris Otto
Markus Lebender
Tom Fleckstein

Web-Services

A look into quality and security aspects

OPENXCHANGE

Joannis Vlachakis
Sascha Rex
Boris Otto
Markus Lebender
Tom Fleckstein

Web Services

A look into quality and security aspects

Web Services

A look into quality and security aspects

Copyright, Disclaimer and Trademark Information

Editor
Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO, Stuttgart

Authors
Joannis Vlachakis
Sascha Rex
Boris Otto
Markus Lebender
Tom Fleckstein

Editorial notes
The present study was made on behalf of the European Commission. The European Commission holds all rights of the text.

The study contains information and research results from the CEN/ISSS initiative and from the RTD project openXchange, which is funded by the European Commission.

Layout
Stefanie von Lohr, lotsofdots

Imprint
Fraunhofer IRB Verlag, Stuttgart

Distribution
Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO
Nobelstraße 12, 70569 Stuttgart
Phone + 49(0)711/970-5120
Fax + 49(0)711/970-5111
Anja.Kirchhof@iao.fraunhofer.de
www.media-vision.iao.fraunhofer.de, www.iao.fraunhofer.de/d/shop

and
Fraunhofer IRB Verlag
Nobelstraße 12, 70569 Stuttgart
Phone + 49(0)711/970-2500
Fax + 49(0)711/970-2508
info@irb.fraunhofer.de
www.IRBbuch.de

Date of publication
October 2003

ISBN 3-8167-6364-2
ISSN 1612-7404

Table of Content

List of figures

Management Summary

Unlike other e-business integration concepts to date, Web Services promise to provide flexible cross platform communication, thereby easing integration issues between systems as well as reducing interoperability challenges for users. Web Services connect computers and devices with each other using the Internet to exchange data and combine data in new ways, thus enabling on-the-fly business relations to a much greater extent than before. It is no surprise that Web Services are said to have a considerable impact on the future development of the Internet as a means for conducting business transactions. The sheer prospect that Web Services may one day become the common way of enabling machine-to-machine interaction is influencing the strategic orientation of big software suppliers (Microsoft, IBM, SAP, among others) and start-ups. The possibility to be able to cost-efficiently and easily integrate heterogeneous IT systems seems to be very tempting, compared to 'traditional' integration methods where such systems could be interconnected only with huge development efforts.

What stands behind the term 'Web Services'? In short, Web Services are self-contained business functions that operate over the Internet. They are composed according to strict specifications to work together and exchange data. Among these functions are messaging, directories of business capabilities, and descriptions of technical services.

Since Web Services are built on a common set of technical specifications, they make it possible for many systems developers to enter the market, which increases competition and brings down costs. Competition among vendors also encourages the willingness to bring about more innovation in the products and services offered to business customers. And developing systems that are based on standards helps prevent being limited to a specific vendor or type of computer or software.

Also, compared to alternative approaches, Web Services technology promises easier implementation and application of cross-company business processes – as a result of the high degree of standardisation of application-to-application communications this technology offers. This enables business partners to implement cost-efficient cross-company business process automation. Therefore, the considerable potential of reducing process costs and organisational overhead can be fully realised.

Both business-to-consumer and business-to-business relationships are likely to change substantially as a result of the use of Web Services. By publishing services in a standardised form, demanders get the opportunity to quickly and efficiently find suitable business opportunities. It can be safely assumed that, in contrast to former approaches, with the deployment of Web Services technology ad-hoc business opportunities and »on-the-fly« service creation become feasible with reasonable effort in terms of monetary costs and time. This may lead to situations where organisations move away from one business partner to another more easily and quickly, which may result in a decrease of customer loyalty and – as a consequence – in an increase of competition.

Despite the great potential Web Services technology offers, there are still some issues to consider before Web Services will become successful.

First of all, the development of Web Services technology is still a work in progress. Some of the standards are still new and not fully tested, and many of the potential business applications are still getting started.

With the widespread dissemination of Web Services, quality requirements will become important selling and differentiating criteria. Quality determines service usability and utility, both of which influence the popularity and raise awareness and acceptance of Web Services. The issue of quality comprises a whole range of requirements that challenge service requesters and service providers. These requirements focus on issues such as bottlenecks affecting the performance of Web Services, approaches of providing service quality, transactional services, and a simple method of measuring the response time of Web Services.

A key aspect of the provision and adoption of commercial Web Services will be the ability to rate the quality of offered services. Methods of providing a rating and revenue chain management solution will be of great interest for providers of commercial Web Services. In this context, the challenge is to provide an objective rating mechanism of Web Services capabilities as well as the degree of these capabilities.

Where data processing is concerned, security issues always need to be addressed. Particularly with regard to e-business, meeting security requirements for privacy, integrity and confidentiality is essential. »No security« effectively means »No trust« and »No trust« means »No business«. With Web Services, it is just the same. If this technology should be adoptable for business purposes, security issues must be addressed as with any other technological approach. While Web Services might differ from competitive technologies, the security requirements do not.

Security is no quality characteristic that can be optimised and improved over time. Security is nothing less than a hard prerequisite for implementing Web Services technology in a business environment. While it is true that some business scenarios do not have equally strong security needs as others do, this does not change the fact that every scenario's security requirements have to be covered, or Web Services cannot be applied to this scenario at all. There is no such thing as »half-good« security. Either an application is sufficiently secure or it is not. There is no in-between.

The security need for Web Services has been addressed by a large number of technical security specifications for Web Services. By the use of these specifications the security need for Web Services can be covered – in theory. However, at this time, there are almost no products available on the market that implement these specifications in a complete manner. Some of these specifications are still too new to be considered stable. It is still possible that some of them will undergo significant changes in the future – what might be a reason that many companies are somewhat cautious not to implement a product based on these specifications too early. It has to be clearly stated that before products are not available on the market Web Services security will not happen at a large scale. In conjunction with the fact that lack of security is one of the largest obstacles for implementing Web Services in a business environment, it should be clear that resources must

be assigned to this matter with high priority. Further difficulties can be found in the missing legal regulations regarding Web Services. In order to be used for commercial purposes, Web Services need accepted and established models for contract completion and contract execution.

Last but not least, it should not remain unsaid that the promotion of further research in the field of Web Services technology and application of this technology is still needed. A considerable effort needs to be spent to close the existing gap between technological capabilities of Web Services technology and its current state of business applicability. A framework needs to be set up by public and non-profit organisations such as standardisation bodies in order to address legal and administrative issues to foster a wide-spread use and to ensure the availability of appropriate skills for the new technology. Moreover, guidance is required to enable the user community to leverage the benefits of Web Services to a degree that corresponds to the vast array of potentials ascribed to this promising technology.

1 Introduction

The situation

One of the most promising aspects of e-business lies in its potential to achieve and support integration on various levels. Both for enterprise application integration, business-to-business application integration and business-to-business process integration, many enterprises have implemented a multitude of differing technical solutions, each one designed to fulfil a certain business purpose, in order to cope with ever increasing global competition. At the same time, companies are undergoing various fundamental changes, manifesting themselves in a previously unknown intensity of company takeovers, mergers, spin-offs, outsourcing projects, and the like. In order to be able to electronically replicate those changing conditions, it is necessary to find mechanisms and technologies that allow flexible, efficient and effective integration of business partners, end consumers, governmental organisations and service providers. Major problems are of technical nature, caused by different communication protocols, different application server technologies, different interface formats and so forth. Therefore, a technology is necessary that helps overcome current obstacles of business-process and systems integration. It is true that several technical approaches, such as CORBA or DCOM, were being developed to address these needs. However, they all have some problems of their own which as yet have hampered widespread adoption of these technologies (refer to chapter 2 for details).

What are Web Services?

From a technical perspective, the Web Services concept seems to be a suitable solution to these problems. Web Services constitute the core of a new concept supposed to facilitate inner-company and cross-company integration of distributed applications by means of standard, XML-based Internet technology. The Web Services concept represents a 'software as services' approach. The basic idea is to loosely connect distributed, disparate applications independent of operating systems, hardware, or programming languages, in order to be able to take advantage of flexible business opportunities.

The W3C offers the following definition of Web Services:

»A Web Service is a software application identified by a URI, whose interfaces and binding are capable of being defined, described and discovered by XML artefacts and which supports direct interactions with other software applications using XML based messages via Internet based protocols.«

Since the W3C is one of the major standardisation organisations for Internet technologies – and thus Web Services technology as well – this definition will be taken as the foundation of this study.

The opportunities

Unlike other e-business integration concepts to date, Web Services promise to provide flexible cross-platform communication, thereby easing integration issues between systems as well as reducing interoperability challenges for users. Web Services connect computers and devices with each other using the Internet to exchange data and combine data in new ways, thus enabling on-the-fly business relations to a much greater extent than before. It is no surprise that Web Services are said to have a considerable

impact on the future development of the Internet as a means for conducting business transactions. The sheer prospect that Web Services may one day become the common way of enabling machine-to-machine interaction is influencing the strategic orientation of big software suppliers (Microsoft, IBM, SAP, among others) and start-ups. The possibility to be able to cost-efficiently and easily integrate heterogeneous IT systems seems to be very tempting, compared to 'traditional' integration methods where such systems could be interconnected only with huge development efforts.

On the user side, the Web Services concept is also causing excitement. Should the theoretical implications become a reality, this technological approach may very likely help businesses to address constantly changing demands much more effectively, thereby increasing productivity and profitability. As they think about the implementation of a Web Services strategy, company decision-makers intend to gain more freedom as to which electronic services should be made available to customers and partners, or – from a demanding point of view – which services should be used from Web Services suppliers. Companies would then be able to quickly create new business opportunities, establish flexible partnerships, speed application integration, and build up effective connections with their customers.

Ideally, Web Services provide both for a separation of functionality and usage and for a separation of business purpose and technical implementation. Web Services promise to increase process efficiency by eliminating media discontinuity and improving information flows, thereby accelerating workflows and lowering error rates. Moreover, Web Services are said to strengthen the orientation towards business processes when building up and extending an IT infrastructure; processes need no longer be pressed into the straitjacket of monolithic IT systems.

The challenges

From what has been outlined so far, it can be assumed that Web Services may serve as an excellent catalyst for the collaboration of systems and applications. Provided this potential can be largely exploited in reality, e-business would be confronted with a really novel concept for implementing distributed systems and applications. What has to be considered, however, is the degree to which Web Services contribute to the value creation within and between organisations. A deliberate strategy has to be developed which allows to fully take advantage of Web Services based applications. Sustained and economically reasonable framework conditions for the provision and use of Web Services must be created, in which profitable business models may yield the desired results.

But what are practicable solutions for the use of Web Services in an overall business context? There is a broad consensus among experts that e-business integration will gradually call for more flexible IT infrastructures, especially with regard to cross-company integration. In this context, Web Services are regarded as having the potential to achieve a degree of interoperability which allows companies to employ various software platforms, since Web Services standards (SOAP, WSDL, UDDI) are independent of underlying programming languages, operating systems and transport protocols. This fact lets us come to the conclusion that the number of business partners

to be linked electronically to a company's systems can be significantly increased, if XML based interfaces are implemented. However, the additional flexibility enabled by Web Services technology has a few problems of its own which need to be considered. For example, new laws and regulations are needed to cope with the specific requirements caused by conducting new ways of business. Also, from a more technical point of view, the higher degree of integration demands new approaches to connect these interfaces by business-process flows.

Compared to other infrastructure technologies, Web Services allow an evolutionary, step-by-step procedure, thereby improving the cost-benefit relation of application integration; legacy systems can be equipped quite easily with Web Services interfaces, and the flexibility of XML allows to modify interfaces without penetrating existing connections. Web Services are highly reusable, so each one can be used for several business purposes or combined with other Web Services to support entire process chains. Yet in order to fully exploit the potential of Web Services, new know-how and skills must be developed, not only on the part of Web Services developers but also on the part of Web Services providers and users. In highly integrated Web Services networks many applications will exchange function calls and parameters and process the results. Therefore Web Services enabled applications have to be developed more carefully than previously common forms of Web applications. In such an environment, processing errors could quickly propagate from one application to another and cause even more errors. And, last but not least, new security risks have to be addressed to enable the use of Web Services in a business environment.

Objectives of the study

The basic objective of the study is to discuss the potentials ascribed to Web Services. Expectations on Web Services technology are extremely high, although the concept is still young and largely unproven in real-existing business scenarios. Therefore, the study will at any stage take into account the critical aspects of the Web Services idea instead of ballyhooing about the revolutionary potentials of another killer concept.

- First of all, the basic technological and conceptual elements that constitute Web Services will be illustrated in order to provide a common understanding of the nature of this novel approach.

- The study will then elaborate on important security and quality issues related with the Web Services concept.

- Subsequently, possible application areas of Web Services will be examined. Three example scenarios will be developed to show how concrete Web Services based applications may look like, which previously problematic aspects may be improved and facilitated by Web Services, which critical issues still exist despite the use of Web Services, and what difficulties and obstacles actually result from the use of Web Services.

- Furthermore, the study will focus the actors on the developing/supplying side of Web Services and possible business models both for existing players (how will they be affected by the advent of Web Services?) and new players (what roles can they take over in the market?) etc.

– Also, the user side will be addressed. The central questions are: What impact will Web Services have on enterprises' business habits? What benefits can enterprises expect that take advantage of Web Services? What problems may arise?

– Finally, the study will address standardisation and regulation issues (which have always been an important aspect in the e-business and e-commerce context, but which experience a new level of complexity when it comes to promoting Web Services on a broad basis). Here, the role that superior authorities may play in establishing sustained framework conditions for Web Services will be of special interest.

Web Services technology can be provided and used by anybody, regardless of regional and national aspects. Because of this fact, the study did not undertake the effort to conduct market analyses and determine dissemination rates with regard to specific countries or regions.

Instead, the study examined Web Services-related market opportunities and possible business models of relevant actors in the e-business market and outlined the changing conditions that are likely to occur if Web Services technology would be established in the market. On the side of potential Web Services users it was examined what potentials and benefits Web Services could yield and how Web Services could affect the value-adding chain.

Methodology of the study

The objective of the present study was to conduct both primary and secondary data analysis in order to gather information about Web Services. With regard to primary data collection, it was decided to follow a qualitative research approach and to conduct exploratory (in-depth) interviews with e-business/Web Services experts both of organizations acting on the supplier/developer side and organizations on the user side. The intention of the secondary data collection was both to gather qualitative information (from statements of experts in specialized magazines, in specialized books and on relevant websites) and to search for quantitative and quantifiable data provided by surveys of opinion poll researchers.

The overall aim of the primary data collection was to make implicit knowledge explicit in a very young thematic field where interviewees could not necessarily be expected to have profound knowledge or justifiable assumptions. The intention was to conduct in-depth interviews with an openly designed topic guide in order to encourage two-way communication between the interviewers and the interviewees. To this purpose, a semi-standardized, semi-structured questionnaire was constructed which allowed the interviewers to be flexible with the order of questions, to leave out questions or entire blocks of questions (in case the interviewees would not feel competent or willing to provide information) or to go more into detail in thematic blocks where the interviewees would be very keen to talk about.

Due to the fact that Web Services are still a quite unknown phenomenon and hardly implemented in real-world cross-company business scenarios, efforts to gain interview partners from different industries in order to collect primary data on the user side largely resulted in failure. Persons initially willing to take part in an interview later became very reluctant and sceptical as to whether they would be able to really provide expert information about the topic. Although these persons were told that the interviews will have exploratory character (means that the interview would rather be a talk inter pares about basic issues concerning Web Services in particular and IT/e-business issues in general) and that the questionnaire would rather address generalists (with background knowledge both in IT and business issues), not Web Services experts, most of the interview commitments were withdrawn.

However, three interviews took place with experts on the supplier/developer side (RosettaNet, OASIS and Fraunhofer). A great deal of the findings of these interviews were implicitly (means not explicitly marked as the opinion of an individual or group of individuals) included into the corresponding sections of the study.
In order to take into account the fact that primary data collection could not be conducted to the desired extent, the secondary data analysis was further extended to provide a sound empirical basis for the study. Since a very lively discussion about Web Services has taken place for some time, the authors of the study had no difficulties in finding comprehensive qualitative data in specialized magazines, in specialized books and on relevant websites.

The situation with quantitative and quantifiable data, however, is different. The present study does not include, build upon or refer to statistical data regarding the future market potential of Web Services. Since Web Services represent a very young technological approach, reliable and valid statistical data is rare. No statistical survey of a renowned supplier of statistical data could be found that offers a quantitative prediction of the future development and dissemination of Web Services in Europe and the U.S. Other data by less known opinion research institutes surveying the acceptance and dissemination of Web Services were taken note of. Since these researchers fail to provide a transparent understanding of their methodology and used terminology, however, the authors of the present study came to the conclusion that these surveys do not take into account the obvious, inevitable inability of people on the user side to provide usable information. Very frequently, it appears that the notion of Web Services is mixed up with what must be considered Web-based services – and more often than not the survey designers do little to clear this misunderstanding.

Besides, the quality and accuracy of many statistical analyses resulting from opinion polls and interviews about e-business issues has proven to be not very convincing, to say that in moderate words. In issuing far too optimistic statistical forecasts, many opinion research experts completely went astray in their efforts to predict e-business market trends correctly, both with regard to the dissemination of individual technologies and applications, and with regard to estimated sales figures and revenues. The authors of the study decided not to fall back on questiona-

ble statistical data but to stick to a serious analysis of the core aspects of Web Services technology and Web Services applicability.

2 Technological aspects

It is no secret that Web Services are not the first technological approach to address the issue of effectively implementing distributed applications. Several different approaches deal with remote function calls (the three most important approaches are very briefly illustrated below). Particularly the use of XML and standard Internet protocols is what differs Web Services from older approaches for distributed applications. The latter often use proprietary interfaces or run only on certain operating systems. As default, Web Services use the HTTP protocol – the same protocol as the WWW itself. However, it is also possible to use other protocols, such as SMTP.
As for the interactions with Web Services, it is possible to call a service via a direct remote procedure call (RPC) or alternatively, to work document-oriented. When using RPCs, the methods or procedures on the target system are directly called using their respective parameters and the answer is sent back in real time (which means, it is a synchronous method). The document-oriented approach is about parsing and processing entire documents (e.g. business transactions). Here, the response might be generated much later (that means, that it is an asynchronous method).

2.1 Technologies for implementing distributed applications

2.1.1 Distributed Computing Environment (DCE)

Behind DCE is the idea to create a standardised programming interface for all platforms, from super computers to small Windows based PCs. DCE was developed by the Open Software Foundation (OSF). Apparently, the most important element of DCE is its Remote Procedure Call (RPC) mechanism, which allows to call up and execute functions on remote computing systems. DCE must be considered a failure, however. A main reason for this negative judgement is the high complexity of DCE, which comprises about 600 API functions. Besides, DCE was knocked out at an early stage by approaches of higher technical quality, such as CORBA or RMI.

2.1.2 Common Object Request Broker Architecture (CORBA)

CORBA was defined as an architecture for distributed object-oriented applications. A crucial aspect of this approach is its independence of a certain programming language. A runtime component, the Object Request Broker (ORB), allows to call objects remotely by means of an RPC mechanism. CORBA uses the Internet Inter-ORB Protocol (IIOP).
Besides its complexity, one major problem of CORBA is its lack of compatibility regarding different ORB implementations.

2.1.3 Distributed Component Object Model (DCOM)

DCOM, developed by Microsoft, is a technological approach which is very similar to CORBA. The RPC mechanism, however, was deducted from DCE. Another origin of DCOM lies in the Object Linking and Embedding (OLE) architecture, which

allows communication among different, independent Windows applications (e.g. OLE enables to integrate an MS Excel table into a MS Word document). Since it was the objective to not only facilitate communication among applications running on the same computer but to accomplish cross-computer communication of applications, OLE was extended to DCOM. Like with CORBA, access to objects is made possible over interfaces. DCOM is not affected if objects are realised in different programming languages. The definition of the interfaces is done by a proprietary programming language, Microsoft Interface Definition Language (MIDL). In order to be able to locate objects, these objects must be registered in the system. Network nodes exchange information about objects which they accommodate, so that this information is also locally available and may be accessed.
A major disadvantage of DCOM is the fact that this technology can only be used in connection with MS Windows operating systems.

2.2 Web Services

So, what is new about Web Services? Like all approaches mentioned above, Web Services provide a way to call functions remotely. The end user of Web Services based applications might not even notice the difference between such an application and a conventional Web application (e.g. based on JSP technology). What is actually new about Web Services is »under the hood«: Web Services based distributed applications use a common set of protocols. A major advantage of the Web Services concept is that this technology takes into consideration the structure of the WWW, which is not a network of fixed, permanently active connections. On the contrary, the WWW is characterised by ad-hoc connection and disconnection of participants. Usually, communicating participants do not know each other before a connection is set up. Therefore, by means of hyperlinks and the HTTP protocol, the WWW offers a technology by which systems can set up connections and exchange data over a defined mechanism. Web Services follow exactly this philosophy. Users can use the Universal Description, Discovery and Integration Registry (UDDI) in order to find the Web Services they need on the WWW. Furthermore, with the Web Services Description Language (WDSL) a description language is provided which determines how Web Services must be addressed and what structure a Web Service's response is made of. Eventually, the Simple Object Access Protocol (SOAP) provides a mechanism which first transmits the actual request to the Web Service and then returns the response, if any, to the calling system.

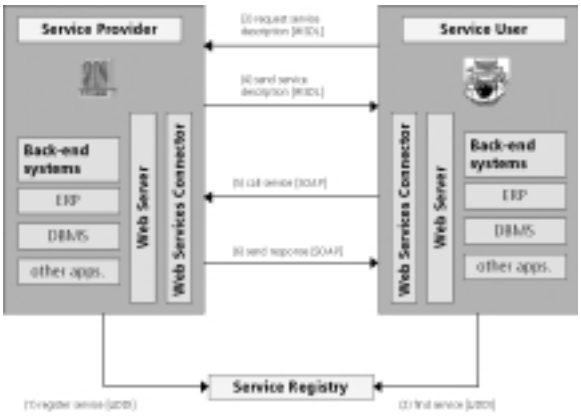


Figure 1:
Structure of a typical Web Services architecture in a business-to-business environment

The above figure illustrates the basic idea of making available and using a Web Service (please note that not all components depicted in the figure are mandatory at all times, e.g. the Web Server or the ERP at the Service user side). First, the provider of a Web Service must register this Web Service with an UDDI operator (step 1) in order to make information available about the specific Web Service. The information on this Web Service can now be delivered as the result of a search request to the UDDI operator (step 2). After that, the requestor must get necessary information as to how the Web Service has to be addressed. For this purpose, a WSDL description is used (step 3) which is not accommodated at the UDDI operator's site but at some other location of the Internet. After the requester has downloaded the WSDL document (step 4), the Web Service can be called under consideration of the specific information which the WSDL document contains. The actual calling of the Web Service is achieved by a SOAP document (step 5), which transmits the information about the desired functions and their calling parameters to the Web Services provider. Having received the request, the Web Service provider sends a response to the demander, also by means of a SOAP document (step 6).

The three specifications will be further illustrated in the following sections. Example source code for these specifications that might be helpful in understanding this technology is also available in the appendix.

2.2.1 Universal Description, Discovery and Integration (UDDI)

Universal Description, Discovery and Integration (UDDI) is a repository services which allows to locate information about specific Web Services. UDDI provides a possibility to search for Web Services according to certain criteria. UDDI is a technological specification and a real-existing service at the same time. This service is offered on the WWW by corporations such as IBM or Microsoft, of which both substantially contributed – together with Ariba – to the development of the UDDI specification.

The way UDDI works slightly resembles the operation mode of the Internet Domain Name Service (DNS). Web Services suppliers can register their services with a UDDI registry provider. Relevant information about specific Web Services is then included into the respective UDDI database. It makes no difference which UDDI registrar is provided with that Information, since the data pools are being mapped with each other on a regular basis. UDDI also comprises mechanisms that are supposed to guarantee data security and data quality. For example, organisations that want to make information available over UDDI first must acquire entitlement from a UDDI registrar. Also, during the registration procedure an encrypted connection (SSL) must be used. As a result of the registration, each newly registered organisation is given a Universally Unique Identifier (‚UUID'). Such a UUID is also allocated for each individual UDDI registered Web Service. After the data has been integrated into the UDDI database, they are made available by every registrar.

UDDI provides three categories of information:

– **White Pages** contain the basic contact information about a supplier of a Web Service, such as company name, address, URL of the homepage etc.

– **Yellow Pages** provide information about the supplier portfolio, products, locations etc.; for this purpose, classification systems (e.g. UN/SPSC[1] , NAICS[2] and ISO 3166[3]) are used, which offer requesters support for the searching process.

– **Green Pages** accommodate technical information about the type and functionality of each Web Service; here, also, a reference to the respective WSDL document, if any, is given.

UDDI is capable of processing any Web Service description, i.e. proprietary formats can be used, too. This means that the use of WSDL is possible but not mandatory. Also, the concrete data management of the individual UDDI registrars may vary, as long as the generation of XML Schema documents as defined by UDDI is guaranteed.

For the use of UDDI, three Application Programming Interfaces (APIs) are available: the 'Publisher API' serves for registering information about Web Services, the 'Inquiry API' is used for searching and retrieving this information. Finally, the 'Subscriber API' can be used to notify users of changes in the registry. The functions of these APIs are called by means of a SOAP document.[4]

UDDI provides a flexible, powerful and at the same time easy-to-use mechanism for storing, locating and calling information about Web Services and their suppliers in a standardised way. However, UDDI does not solve all the problems connected with searching for Web Services. The lack of universally unique systems for classifying organisations, products and services and the difficulties resulting from that fact with regard to the search process cannot be eliminated by UDDI.

2.2.2 Web Services Description Language (WSDL)

Although with UDDI an elaborated repository service for finding Web Services has been created, successfully locating a Web Service does not mean that this Web Service is ready to be used. By means of the Web Services Description Language (WSDL), which was mainly developed by Microsoft, Ariba and IBM, the interfaces, data formats and protocol bindings used for a specific Web Service can be described. So the demander of a Web Service is enabled to correctly address the specific service and interpret the returned answer. In order to use the functionality of a Web Service, the requester must only have access to the WSDL description of the respective service from which all relevant information can be taken. WSDL itself is independent of specific data for-

[1] United Nations Standards Products and Services Classification
[2] North American Industry Classification System
[3] A classification for geographical regions and countries

[4] A full presentation of all API functions would exceed this paper. For further information please see http://www.uddi.org/specification.html

mats and network protocols, but mainly it is used with SOAP, MIME and HTTP GET/POST.

WSDL defines a service as a set of abstract »end points« of network connections exchanging messages. For WSDL, the messages, representing a description of the data exchanged, are also an abstract themselves. Furthermore, WSDL includes operations which describe single sub-actions of the service. All these definitions are executed in an abstract fashion. It is not until a later stage in the procedure that WSDL puts this information into a concrete form for certain network protocols, data formats and URLs.

The elements used by WSDL are illustrated in the following picture. (For further information, please see the selected bibliography in the appendix.)

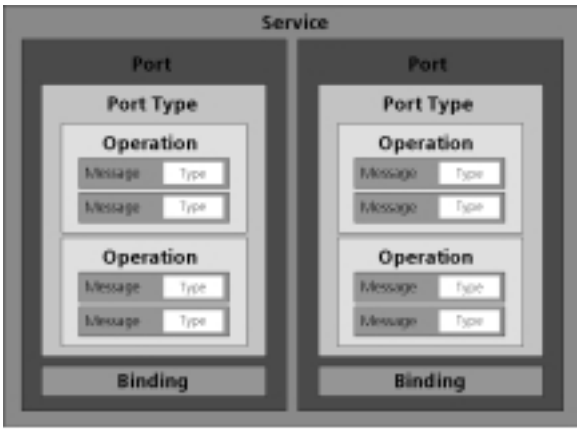


Figure 2:
Structure of WSDL

To sum it up, it can be said that WSDL provides an elaborated mechanism for defining Web Services. Due to their complexity, WSDL documents are difficult to read and comprehend for humans, but this fact is not problematic since for creating and interpreting WSDL documents mostly software tools are used.

2.2.3 SOAP

SOAP is probably the most important component of Web Services technology. SOAP is responsible for the actual data transfer. It provides a standard mechanism both for the sender and the addressee.

Like WSDL, SOAP is based on XML, representing an extension of the XML-RPC specification. SOAP was first published in 1999, and it was submitted to the W3C one year later. Main developers of SOAP are Microsoft and IBM, but with regard to an extension of SOAP other well-known organisations have taken part in the collaboration meanwhile. The latest version, SOAP 1.2, has now received a very high status (»recommendation«) by the W3C, implying that this specification is recommended to be used as a standard. Today, SOAP implementations exist for nearly all important application platforms, e.g. for Java 2 Enterprise Edition, Microsoft .NET, and for all major programming languages. Besides the actual structure of the SOAP message, the SOAP specification also contains rules for the serialisation and deserialisation of data types and a binding for the HTTP protocol.

In order to be able to use SOAP in connection with a Web-based application, besides a conventional Web server (such as Apache or IIS) a SOAP processor is required for the parsing and interpreting of the SOAP documents. However, SOAP is not restricted to be applied for the Internet but can be used in any network.

A SOAP message basically comprises three main elements, called envelope, header and body. The envelope is the root element, defining the beginning and the end of a SOAP message. The header is optional and may consist of one or more blocks which may contain meta-information about the message itself. The body, eventually, contains the actual message.

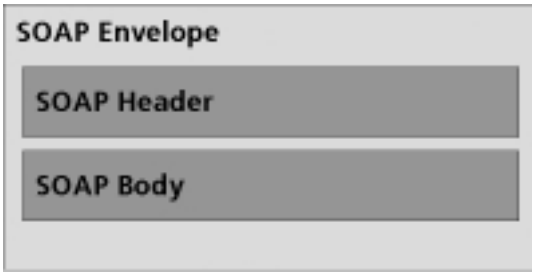


Figure 3:
Structure of a SOAP document

The HTTP protocol itself is not modified in any way. Rather, it can be considered – in a metaphorical sense – as a »beast of burden« that carries SOAP.

SOAP provides a technology for exchanging any kind of data in networks. Crucial advantages of SOAP are its simplicity and extensibility. Another characteristic of SOAP is that this protocol, being basically nothing else but usual HTTP, can pass through most firewalls without difficulties. Reconfiguration of security systems is therefore not necessary. Of course, this advantage may also be turned into a disadvantage, for SOAP has the ability to trigger application functionality behind the firewall, thus touching basic security issues. As a consequence, system administrators have the duty to very carefully inspect the services offered with regard to security gaps. Finally, it must be said that SOAP does not include many of the functions desired for the exchange of data in networks, such as security mechanisms, process support, and transaction management. If such functions are required, the user has to implement or apply additional technology (some of which are described later).

2.3 Development platforms for Web Services

SOAP, WSDL, UDDI and other Web Services technologies are still not directly available products which can be applied at once. Rather, these formats must be considered as technical specifications which remain to be implemented in concrete software architectures before they can be used. There are currently several endeavours under way that deal with that challenge. Two approaches are very likely to achieve broad dissemination: Microsoft .NET and Java 2 Enterprise Edition. Fortunately, both architectures are basically compatible amongst each other, at least in that both are built upon the same specifications and use these specifications in the same way.

2.3.1
Microsoft .NET

Microsoft .NET is primarily based upon Windows operating systems and their functionality. For example, the development tools and server platforms offered by Microsoft are very closely linked with Windows while not being available for alternative operating systems. Like with JAVA, however, the .NET software itself is executed by a virtual machine, so that .NET software can basically run on any platform for which a virtual machine exists. The .NET applications are therefore not translated into a code for a specific processor (e.g. Intel x86) but into an intermediate code, Microsoft Intermediate Language (MSIL). The architecture of the virtual machine for .NET is called Common Language Infrastructure (CLI). The concrete implementation of this architecture is the Common Language Runtime (CLR). The latter translates the MSIL intermediate code into the actual code for a certain processor.

Unlike JAVA, however, .NET is independent of the used programming language. It is therefore possible to implement an application with various languages, for example a combination of Visual Basic and C++. All programming languages supported by .NET fall back on a uniform class library and uniform interfaces, so that single sub-applications may communicate with each other without difficulties. The Windows API and the Microsoft Foundation Classes (MFC), previously common under Windows, are entirely replaced by the .NET variant CLR.

2.3.2
Java2 Enterprise Edition

Compared to Microsoft .NET, J2EE is the older approach. The J2EE technology spectrum, which is primarily coined by Sun Microsystems, comprises well-known technologies such as Java Servlets, Java Server Pages (JSP) and Enterprise Java Beans (EJB). Also, APIs for security and transaction management as well as for the use of directory services belong to the J2EE environment. As of the publication of .NET, J2EE was the only serious approach for implementing modern Web applications which achieved a broader dissemination in the market.

In the wake of Web Services technology, J2EE was extended with further functionality, e.g. comprehensive support of XML and XML-based technologies.

2.3.3
Open-source initiatives

In addition to commercial approaches, several open-source initiatives (such as Apache Axis) exist in the Web Services context which have the potential to provide serious alternatives to commercial products and to exert a strong impact on the acceptance of Web Services. More and more open-source products are in the development phase, which will most certainly provide all necessary means to implement Web Services technology based applications solely built upon open-source products.

It can be safely assumed that all advantages of the open-source philosophy well known from

such successful software as LINUX or the Apache Web Server will also apply to the Web Services environment. Beside the fact that open-source software is free of license fees in most cases, the possibility to read and alter the source code to fit specific needs of the users is probably the most significant advantage of open-source software.

3 Web Services-enabled business processes

3.1 Introduction and scope

Given the functional interplay of SOAP, WSDL and UDDI, Web Services are primarily a technology for implementing application-to-application communication, i.e. a software application may directly use the functionality and services of another software application, provided that both applications support Web Services technology. By using simple Internet technology, such as http for transmitting data, the implementation of distributed systems across company boundaries is facilitated. Software systems directly access the functionality of external applications, resulting in process simplification due to the fact that no common runtime environment, such as an ORB (see chapter 2.1.2), needs to be at hand in each participating company.

Web Services get further significance when they are used on a business-process level. A business process consists of a set of sub-processes that are related to each other. Some sub-processes may only be executed sequentially, since their inputs and outputs are interdependend. Other sub-processes may be executed in parallel, what may require synchronization within the overall process.

In order to define and execute business processes within companies, suitable software systems are used, such as document management systems or workflow systems. If a company wants to integrate business partners directly with their processes, these business partners usually must use an equal or compatible system, since the systems partially use proprietary formats and structures.

In the field of Web Services technology, specifications are currently being developed that deal with that aspect. The objective is that companies willing to participate in cross-company business processes use Web Services technology in order to make the business functionality available which they want to offer partners as single sub-processes. By means of certain specifications, such Web Services can be combined to form one overall process.

Process descriptions, which relate Web Services to each other, are then used as input for process engines responsible for correctly calling the Web Services and transporting the data. As illustrated in Figure 4, Enterprise A uses a sub-process provided by Enterprise B by means of a Web Service. The cross-hatched areas represent the process steps that form the enterprise's interfaces with external partners for exchanging data. The cross-hatched process step in Enterprise A is a representative of the sub-process running in Enterprise B. Enterprise A does not need to know any details about this sub-process; only the interface needs to be defined.

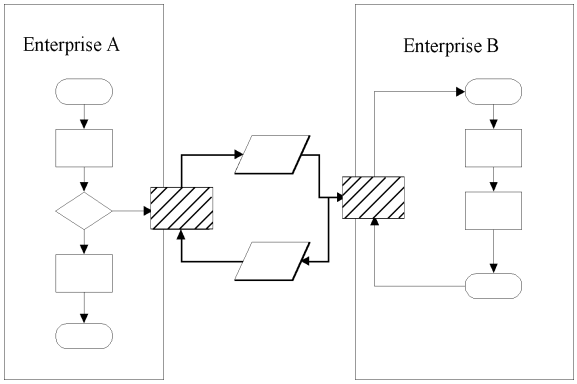


Figure 4:
Cross-company business processes

There are a great number of different specifications which are dealing with business processes or the coordination and orchestration of web services, every specification with a special main focus but naturally there are also common aspects in these specifications. A not complete list of specifications is:

- The Workflow Process Definition Language / XML Process Definition Language (WPDL/XPDL) of the Workflow Management Coalition (http://www.wfmc.org);
- The Business Process Modelling Language (BPML) of the Business Process Modelling Initative (http://www.bpmi.org);
- The Web Service Conversation Language (WSCL) hosted by the W3C and initially developed by Hewlett Packard (http://www.w3.org/TR/2002/NOTE-wscl10-20020314/);
- The Web Service Choreography Interface (WSCI) hosted by the W3C (http://www.w3.org/TR/wsci/);
- XLANG by Microsoft;
- The Web Service Flow Language (WSFL) by IBM;
- The Business Process Execution Language for web services (BPEL4WS);
- The Web Services Composite Application Framework (WS-CAF) by Sun, Oracle, IONA, Fujitsu and Arjuna (http://www.arjuna.com/standards/ws-caf/).

In the following sections, we will give a short description of four of theses specifications: one process-oriented specification, namely BPML (see 3.2), one more technical specification not focussed on web services, namely XLANG (see 3.3) and two technical specification addressing especially web services, namely WSFL (see 3.4) and BPEL4WS (see 3.5).

3.2 Business Process Markup Language (BPML)

Business Process Markup Language (BPML) is a specification developed by an industry consortium, the Business Process Management Initiative (BPMI). The current version of the specification was published Nov. 13, 2002 and is labeled »Last Call Draft«, meaning that the specification's content has been fixed and that inconsistencies, if any, will be eliminated by Dec. 13, 2002 (see Arkin, 2002). Companies that have substantially contributed to BPMI are CSC, Intalio, SAP, SeeBeyond, Sun and Versata.

BPMI defines the focus of BPML as follows (see BPMI, 2000):

»The Business Process Modeling Language (BPML) is a meta-language for the modeling of business processes, just as XML is a meta-language for the modeling of business data. BPML provides an abstracted execution model for collaborative & transactional business processes based on the concept of a transactional finite-state machine.«

BPML provides constructs that allow the definition and manipulation of the data flow as well as the definition of single process steps and of the control flow of these process steps. With regard to control flow, four types are distinguished (see Thiagarajan et.al., 2002):

– **»value based«**: The dependence of single process steps (activities) is determined by the values of the process data at runtime.

– **»state based«**: The state of the process determines the dependencies.

– **»time based«**: The control flow is limited by total runtimes or is subject to a timetable.

– **»cycle based«**: The control flow is done by repetition of one or more activities (similar to loop constructs in programming languages).

BPML also offers the opportunity to use nested activities and to define the way (sequential or parallel) a group of activities is to be executed. Furthermore, BPML offers transaction support for both ACID (coordinated) and long-running (extended) transactions as well as methods and constructs for exception-handling.

Figure 5 gives an overview of the different activities that may occur within a process or a group of activities. In order to coordinate single process steps within a group or a process, contexts can be defined that pass information about the actual state of the process between single activities.

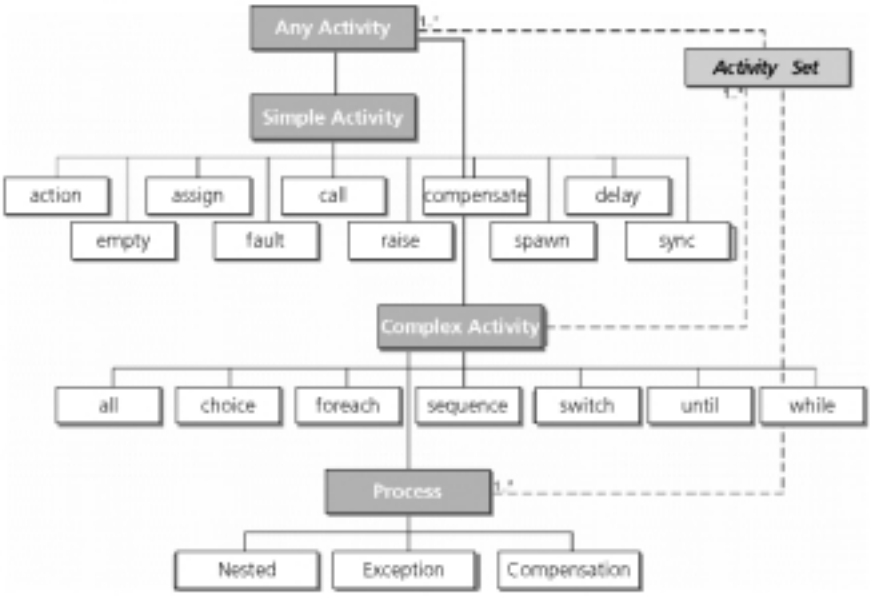


Figure 5:
Different activities
of the BPML specification

BPMI is currently working on a specification for graphically representing business processes, called Business Process Modeling Notation (BPMN). Based on the graphical representation, it becomes possible to automatically generate documents that define the business processes. At present, BPML and BPEL4WS are supported.

3.3 XLANG

Microsoft's product in the field of process integration is the Microsoft BizTalk Server. This product is complemented by a set of tools providing support in process integration, e.g. the »BizTalk Orchestration Designer«. This tool, which is based on Microsoft Viso, allows to model the process in order to couple single process steps directly with adequate software fragments, e.g. Web Services (see Figure 6). However, Web Services technology is only one option among others in this context.

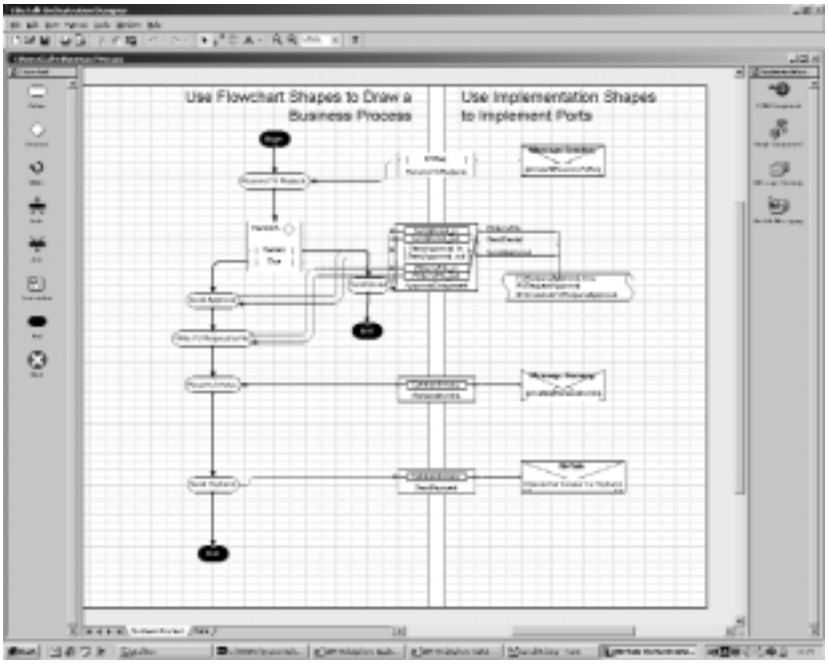


Figure 6:
Screenshot of the Microsoft BizTalk Orchestration Designer

Based on the so-called »XLANG Scheduler Drawing«, an »XLANG Schedule« is generated by the Orchestration Designer. The basis of XLANG Schedule is »XLANG«, a language defined by Microsoft (see Thatte, 2001). Microsoft defines the focus of XLANG as follows:

»A language that describes the logical sequencing of business processes, as well as the implementation of the business process by using various application services.« (see Microsoft, 2002)

Execution of a process defined by an XLANG Schedule is then done over the »XLANG Scheduler« of the Microsoft BizTalk Server.

Microsoft will not continue the development of XLANG, because Microsoft is participating since August 2002 in the development of the Business Process Execution Language for Web Services (BPEL4WS), which is described in chapter 3.5

3.4 Web Services Flow Language (WSFL)

Web Services Flow Language (WSFL) (see Leymann, 2001) is a language developed by IBM for composing Web Services. The first version of the specification (Version 1.0) was published in May 2001. IBM is also participating in the development of the BPEL4WS, so that WSFL will be replaced by BPEL4WS. Therefore only a few basic principles of WSFL will be presented here.

WSFL distinguishes between two kinds of models, each focusing a certain aspect of the composition:

- The **»flow model«** primarily describes business processes. The roles of the partners involved in a certain process are identified, and single activities (process steps) within the process are defined and related to each other (business rules), e.g. by defining the control flow (»control links«). Besides, data flows are modeled, i.e. it is defined which input and output data correspond and must be transformed, if need be (»data links«). The flow model defines a »public interface« representing the exterior view of the new, composed Web Service.
- **»Global models«** do not define any control flows or other process rules. Global models only define the interactions among all Web Services involved, including the overall Web Service which is composed of other Web Services. By means of »plug links«, the operations of the public interface are combined with operations of the providers of the subprocesses.

Two brief scenarios might provide for a better understanding of the models:

Scenario 1: An enterprise wants to implement a certain business process by coupling various (both own and external) Web Services. In this case, a flow model is used in order to relate the process steps (i.e. the single Web Services) to each other, thereby defining the control flow, transitioning conditions, data flow etc.

Scenario 2: An enterprise wants to offer business partners an integrated Web Service (which is the result of a combination of several Web Services). Like in the first scenario, the flow model is used, but there is no fixation of certain service providers. Instead, roles and activities are defined on an abstract level. The concrete involvement of the service providers is now defined by the global model. Service providers may be selected according to certain conditions and rules by WSFL Service Provider Locators.

Figure 7 shows the relations among flow models, public interfaces and global models. A flow model defines activities and exports a public interface, which is described by WSDL (see chapter 2.2.2), for the combined Web Service. Here, selected activities correspond to operations. By means of the plug link, in the global model the operations of the public interface are then mapped onto the operations of the used services.

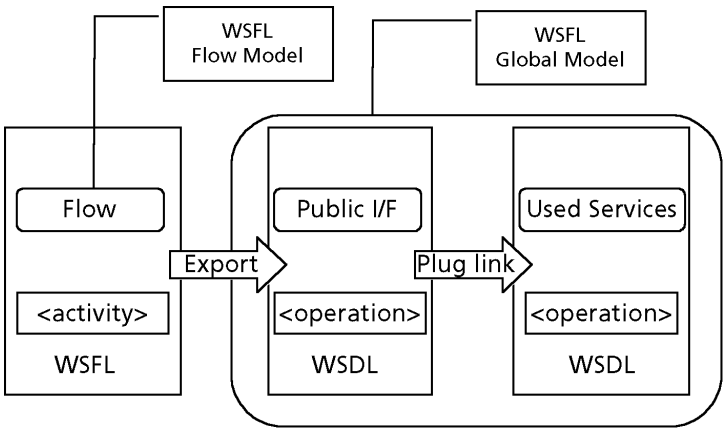


Figure 7:
Relations between flow models, public Interfaces and global models (see Leymann, 2001)

WSFL is already supported by several software tools, which are not being developed by IBM solely. Information about an open-source initiative can be found, for example, under http://flowrider.sourceforge.net/.

3.5 Business Process Execution Language for Web Services (BPEL4WS)

As already mentioned, BPEL4WS is the official successor of XLANG and WSFL. Business Process Execution Language for Web Services (BPEL4WS) was published in August 2002 by Microsoft, IBM and BEA (see Curbera, 2002) with version 1. The current version 1.1 was published in May 2003. The further development of this specification will be hosted by OASIS (http://www.oasis-open.org), more exact the OASIS Web Services Business Process Execution Language TC.

Similar to WSFL distinguishing between flow model and global model, BPEL4WS makes a difference between »abstract processes« and »executable business processes«. For example, during a usual procurement process, the buyer and the seller each take up a certain role which are both characterized by an abstract process: a buying process and a selling process. In the description of an abstract process only the data are used which are necessary for the respective business partner. The connection of the two processes is done by a service link.

This degree of abstraction, however, is not sufficient to actually execute the procurement process. To do so, both sides may define private,

executable processes which implement the public interface. Here again, aspects such as the control flow of process steps and data manipulation is of relevance. BPEL4WS offers constructs for both types of the process view.

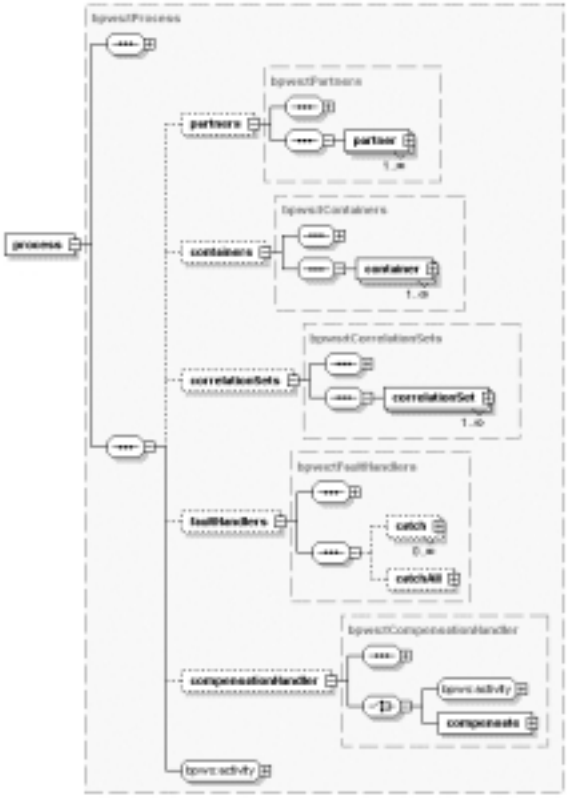


Figure 8:
Schema of a BPEL4WS process

The schema of a process definition in BPEL4WS is shown in Figure 8. First, the various roles of the business partners involved in the process are defined and containers are provided which include the data that are exchanged among the individual process steps. For handling invalid actions during the process, specific handlers and compensators are defined. Correlation sets offer a possibility to relate and identify exchanged data. These data are required because the individual process steps of the modeling at runtime are realized in different, mutually independent instances. For the overall execution of a process, however, it must be possible not only to identify the data of one process step but of an instance of a process step. In BPEL4WS, this is achieved by correlation sets.

The following further activities are defined – the meaning of which can be identified by their names –, which can be used for the above scheme:

– Receive
– Reply
– Invoke
– Assign
– Throw
– Terminate
– Wait
– Empty
– Sequence
– Switch
– While
– Pick
– Flow
– Scope
– Compensate

The »scope«-activity allows to use own error-handling procedures for a defined number of process steps instead of using the procedures of the overall process. The activites »sequence« and »flow« are used to define the execution order of sub-processes: activities within a flow activity may be executed in parallel, activities within a sequence activity only one after another. In order to get control over the sequencing of parallel activities anyhow, the link construct allows to relate two activities to each other.

3.6 Business process related activities in ebXML

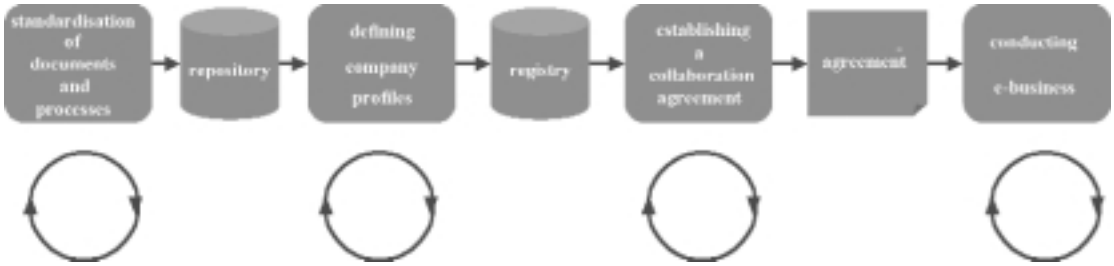
ebXML is meant to be the overall framework of B2B electronic commerce. Though still under development, ebXML will offer a complete set of standards to be used by businesses, organisations and authorities for their mutual electronic communication. ebXML comprises both technical standards for registries, repositories, protocols, profiles, agreements, etc. and standards for business modelling, information components and entities, registration procedures, etc. ebXML is a set of specifications that together enable a modular electronic business framework. ebXML enables a global electronic marketplace where enterprises of any size and in any geographical location can meet and conduct business with each other through the exchange of XML-based messages. ebXML is jointly sponsored by the United Nations (UN/CEFACT) and the Organisation for Structured Information Standards (OASIS).

ebXML is composed of four infrastructure components as depicted in the following figure (see OPENXCHANGE, 2002).

The general ebXML architecture addresses business process integration and automation not only at run-time, but also at design time. It provides specifications to automate the description and matching of process profiles of different companies as well as for the execution of business process instances.

Whilst Web Services technologies provide technical oriented solutions to overcome the obstacles of business process integration, ebXML provides the business process oriented input and provides process specifications that could be based on Web Services technologies. Web Services and ebXML are therefore complementary in the area of business process automation and integration. However, overlappings can be identified in the area of registry and messaging specifications.

Figure 9:
ebXML Infrastructure Components



3.7 Conclusion

Four specifications were illustrated which aim at modeling business processes, both inner-company and cross-company, by means of XML. All four specifications require that each respective process step is implemented as a Web Service – a requirement which is still far from reality.
Of the four specifications outlined, two will not be developed further: XLANG and WSFL. The initiators of these two specifications, IBM and Microsoft, have recognized that it makes little sense to work on different specifications for the modeling of business processes and the implementation of those processes as Web Services.

Although the approaches differ only slightly from each other, adaptation efforts in business-to-business integration projects are still very time-consuming and cost-intensive. However, this contrasts with the basic idea of Web Services: to enable application-to-application integration as simple and cost-efficient as possible. There are enough problems to deal with on a semantic level so that additional problems on the level of technical integration, caused by diversified specifications, would entirely question the Web Services approach.

The willingness to enable standardization, or at least interoperability, exists, which becomes obvious best by the fact that BPMN supports both BPML and BPEL4WS, and that BPMI also has published a position paper on BPEL4WS.

4 Quality aspects

With the widespread dissemination of Web Services, quality requirements will become important selling and differentiating criteria. Quality determines service usability and utility, both of which influence the popularity and raise awareness and acceptance of Web Services.

The issue of quality comprises a whole range of requirements that challenge service requesters and service providers. These requirements focus on issues such as bottlenecks affecting the performance of Web Services, approaches of providing service quality, transactional services, and a simple method of measuring the response time of Web Services.

A key aspect of the provision and adoption of commercial Web Services will be the ability to rate the quality of offered services. Methods of providing a rating and revenue chain management solution will be of great interest for providers of commercial Web Services. In this context, the challenge is to provide an objective rating mechanism of Web Services capabilities as well as the degree of these capabilities.

Although crucial for the dissemination and adoption of Web Services, currently there are no commonly accepted products and quantifiable measurement systems available for rating the quality of Web Services.

The core questions which arise in conjunction with that issue are:

- Is a rating system for Web Services feasible at all?
- What criteria should be rated and in what way should they be rated?
- Who should/could do the rating?
- Who rates the rating agency?
- Can the rating agency be held responsible for faults or misbehaviour of Web Services?
- Must ratings be available within the UDDI registry?
- What about the price readiness of the market? Will achievable payments be enough to sustain objective rating processes?
- Who will be charged for rating services (the Web Services provider or the Web Services user or both)?
- Should a rating agency have a monopoly in the industry (if not, what can be done about it)?

In this context, the question arises as to which quality aspects should be taken for benchmarking Web Services. Also, one has to decide on the indicators to be used for the measuring. In the evaluation process of Web Services the following quality aspects regarding non-functional properties should be considered (following IBM »Understanding quality of service for Web Services«).

As far as multilinguality of Web Services is concerned, the authors of the study would like to point out that multilinguality on the level of Web Services technology, which per definition is used to enable machine-to-machine communication and must not be mixed up with web-based services, is not considered a quality aspect that has to be taken into consideration here.

The following quality aspects and quality criteria regarding Web Services can be applied analogously to similar technologies and services.

4.1 Availability

Availability is the quality aspect of whether the Web service is present or ready for immediate use. Availability represents the probability that a service is available. Larger values represent that the service is always ready to use while smaller values indicate unpredictability of whether the service will be available at a particular time. Also associated with availability is time-to-repair, which represents the time it takes to repair a service that has failed. An indicator for measuring availability could be the percentage of daily/monthly/yearly availability.

4.2 Accessibility

Accessibility is the quality aspect that represents the degree a Web service is capable of serving a Web service request. It may be expressed as a probability measure denoting the success rate or chance of a successful service instantiation at a point in time. There could be situations when a Web service is available but not accessible. High accessibility of Web services can be achieved by building highly scaleable systems. Scalability refers to the ability to consistently serve the requests despite variations in the volume of requests.

4.3 Integrity

Integrity is the quality aspect of how the Web service maintains the correctness of the interaction in respect to the source. Proper execution of Web service transactions will provide the correctness of interaction. A transaction refers to a sequence of activities to be treated as a single unit of work. All the activities have to be completed to make the transaction successful. If a transaction is not completed, all the changes made are rolled back. A possible indicator for measuring integrity could be the percentage of correctly processed transactions.

4.4 Performance

Performance is the quality aspect which is measured in terms of throughput and latency. Higher throughput and lower latency values represent a good performance of a Web service. Throughput represents the number of Web service requests served at a given time period. Latency is the round-trip time between sending a request and receiving the response.

4.5 Reliability

Reliability is the quality aspect that represents the degree of being capable of maintaining the service and the service quality. The number of failures per month or year represents a measure of reliability of a Web service. In another sense, reliability refers to the assured and ordered delivery for messages being sent and received by service requesters and service providers.

4.6 Regulatory

Regulatory is the quality aspect in conformance with the rules, the law, compliance with standards, and the established service level agreement. Web services use a lot of standards such as SOAP, UDDI, and WSDL. Strict adherence to correct versions of standards (for example, SOAP version 1.2) by service providers is necessary for proper invocation of Web services by service requesters.

4.7 Security

Security is the quality aspect of the Web service of providing confidentiality and non-repudiation by authenticating the parties involved, encrypting messages, and providing access control. Security has added importance because Web service invocation occurs over the public Internet. The service provider can have different approaches and levels of providing security depending on the service requester. Please see the next chapter for detailed information on security of Web services.

The degree of fulfilment of both these quality aspects and the security aspects (see chapter 4) will have a crucial impact on Web Services based »high value / high risk« business scenarios. If cross-company processes involving high financial transactions are to be executed by means of Web Services, compliance with strict quality and security requirements is an indispensable prerequisite. Not meeting these requirements would lead to Web Services being used only in uncritical, strategically irrelevant domains.

4.8 Usage Scenario 1: Quality aspects

The relevance and importance of the quality aspects outlined above will now be illustrated by means of an example business scenario.

1 »Pharma&Health«, a producer of pharmaceuticals, is confronted with problems regarding the procurement of raw materials. Despite the implementation of an electronic procurement system, the execution of procurement processes is characterised by high inefficiencies, resulting in high transaction costs. One major problem is the fact that the materials are procured over a number of different electronic trading platforms. Due to the high volatility of prices, the procurers always have to compare up-to-date prices and find the most attractive offers on the various platforms before they

can make a purchase order. This sub-process, which can be executed only with a high degree of manual activities (e.g. by phone or e-mail), is very extensive and cost-intensive.

2 The situation is equally difficult for the trading partners of Pharma&Health. »Premium Materi als« is a large raw-materials wholesaler that provides one of the electronic platforms Pharma&Health is using for procurement. Premium Materials complain about the fact that – despite having implemented the electronic platform – the desired cost savings could not be accomplished. One reason for that is that the company has to deal with a high number of price inquiries made by phone or e-mail, which must be manually processed by the sales staff.

3 »Web Services Development & Consulting«, an IT company, gets to know about the problems of the two companies and decides to implement a Web Service for optimising sourcing processes, allowing procurers to easily and flexibly make price inquiries to the co-operating platforms. This requires that the Web Service developed by Web Services Development & Consulting must be made available on the platforms. Web Services Development & Consulting has the Web Service registered at an UDDI registry operator. Raw-materials wholesalers using the Web Service are charged on the basis of a license model. The advantage for the wholesalers: by using the Web Service they benefit from increased customer satisfaction and reduced process costs, mainly achieved through a significant reduction of manual activities.

4 Pharma&Health mandates that the Web Service satisfies all of the following quality criteria:

Criteria	Indicator	Value
Availability	Yearly availability	99%
Accessibility	Probability measure denoting the success rate	99%
Integrity	Correctly processed transactions	100%
Performance	Response time	< 40 sec
Reliability	Number of failures	2 per month
Regulatory	Certification	Yes
Security	Certification	Yes

If the desired values cannot be fulfilled, Pharma&Health and Premium Materials will not use the Web Service. The business model of Web Services Development & Consulting would then end in failure. The initial situation at Pharma&Health and Premium Materials, characterised by massive inefficiencies, would remain unchanged, and the investments for licensing the Web Service would have been made in vain.

5 Security aspects

Where data processing is concerned, security issues always need to be addressed. Particularly with regard to e-business, meeting security requirements for privacy, integrity and confidentiality is essential. »No security« effectively means »No trust« and »No trust« means »No business«. With Web Services, it is just the same. If this technology should be adoptable for business purposes, security issues must be addressed as with any other technological approach. While Web Services might differ from competitive technologies, the security requirements do not.

According to a survey of 400 enterprise development managers done by Evans Data Corp., the single biggest obstacle to Web Service implementation are security issues (45,5% of all answers).

While IT security often is equaled to access control (in the form of mechanisms to restrict access to information or systems to authorized users only), there are actually much more aspects to consider. For instance, other dimensions of IT security are confidentiality, data integrity, virus protection, intrusion detection or the controversially discussed intellectual property protection (in its technological incarnation: »Digital Rights Management«). Not all these dimensions are equally relevant for all applications of information technology. When identifying the relevant dimensions of security for Web Services, the result is a list that is basically identical with the one that also applies to the world of traditional client/server applications. This is not surprising – after all, Web Services still can be regarded as a form of client/server technology.

In conjunction with Web Services, these are the most relevant dimensions of security which need to be addressed:

– **Authentication / Authorisation** (sometimes also referred to as **access control**) is basically the demand to restrict system access to authorised users only. Authentication means that the system must be able to identify any user. Intruders should not be able to masquerade as someone else. Authorisation regulates what an authenticated user can do (e.g. read some specific data).
The access control mechanism must be able to deal with both external or internal attacks on the system. External attacks are conducted from outside the system by persons who should not gain access to the system at all. Internal attacks are made by users who are in principle authorised to access the system but intend to impersonate someone else).
Access control for Web Services can be implemented in different ways that are comparable to those mechanisms used for securing conventional Web sites. The most widely spread are the authentication via user name/password or to restrict access from authorised systems only (e.g. via IP addresses). However, the latter is mainly used to secure private networks.

– **Data/Message integrity**: Data integrity describes the need to be able to verify data (or messages) regarding their genuineness. The system should include mechanisms to identify and reject data that has been tampered with or data that has been transferred incorrectly. One possible implementation to ascertain data integrity is the usage of digital signatures.

– **Confidentiality:** Confidentiality is the demand that data may only be viewed by legitimate users, even if other access control mechanisms are bypassed. Confidentiality first of all means to secure data from eavesdroppers. It should be impossible for attackers to intercept and read any data during transfer. This is especially important for any service relying on the transfer of confidential data over insecure networks (like the Internet). Electronic business regularly makes the transfer of sensitive data necessary.
Confidentiality is usually implemented by the use of cryptographic algorithms like AES. However, there are different options to secure data against eavesdropping. For example it is possible to encrypt the entire session (e.g. via SSL) or just the payload (for example, in scenarios where the overhead of the SSL protocol is undesirable).

– **Transaction integrity**: The demand for transaction integrity has its origins in the world of database management systems but is valid for web based services, too (especially when a Web service conducts write-accesses on a database). It has be ascertained that data is changed in a consistent manner, e.g. not simultaneously changed by two different users or read by one user while changed by another. If transaction integrity can not be ascertained through suitable mechanisms, the corruption of data will become rather likely. For Web services it means, that function calls via Web services that can not correctly executed with a proper transaction, must be handled in some way (e.g. error handling, user feedback etc.). It is desirable to include this features directly to the protocol. Transaction integrity is usually handled by workflow protocols and will not be sfurther discussed here.

– **Privacy**: Privacy is a human right. It describes a person's right to limit access and use of individually identifiable information. Personally identifiable information is often required by individuals and companies in order to perform services for the individual (e.g. a doctor requires medical information about his patients). Privacy relates to control over what is done with this information, especially whether it is redistributed to third parties without the individual's knowledge or consent. Privacy may be managed by a combination of technical and legal means. Confidentiality technology may be used to protect privacy, but cannot prevent inappropriate sharing of information.

5.1 General XML security

Web Services are an XML-based technology, and as such the general issues of XML security also apply to Web Services. Moreover, the general XML security specifcations are incorporated in Web Service specific security specifications. Therefore, a keen understanding of the general XML security is an absolute must in order to apply security to Web Services.

XML-derived languages, such as SOAP, are text-based and extensible. So, it should be possible to provide security measures such as confidentiality, integrity or access control to entire XML documents or portions of these documents in a way that doesn't hamper the advantages of XML,

most important of all the possibility to process XML documents with standard XML tools. This has the consequence that most »classical« security technology cannot be applied to XML unchanged. For instance, it would be inappropriate in most circumstances to simply encrypt the entire XML document with a cryptographic algorithm (as it would be done with binary data), for the resulting data would not even be recognizable as XML anymore and thus it would be impossible to use a standard XML parser to process this document.

As a result, a specific architecture for XML security has been developed. The core XML security specifications are:

- **XML Encryption (XML-Enc)** (which addresses confidentiality)
- **XML Digital Signature (XML-DSig)** (which addresses integrity and verifiability)
- **XML Access Control Markup Language (XACML)** (which addresses authorization rules)
- **Security Assertion Markup Language (SAML)** (which addresses authentication and authorization, with a particular focus on »single sign-on« services)
- **XML Key Management (XKMS)** (which addresses key management)

These specifications transfer existing knowledge and technologies into the XML world to meet the new requirements of XML. It is possible to use XML security technology in conjunction with existing transport security technologies such as Secure Socket Layer (SSL) / Transport Layer Security (TLS). All these listed XML security specifications can be applied to Web Services as well and shall be addressed in more depth.

5.1.1 XML Encryption (XML-Enc)

The XML Encryption specification enables the application of confidentiality to XML documents. While transport security technology such as SSL also could be used, these technologies secure content only while in transit, not when the content is stored. XML Encryption can be used to ascertain confidentiality to XML documents both while in transit and when stored on a server. Besides, there are scenarios where SSL simply is not an option – for SSL operations take tremendous processing time. Also SSL can't be used when different portions of the document need different treatment. For instance, sometimes a part of the document should be readable by one person while a different part should be accessible only to another person.

Also it would be possible to encrypt XML documents with common cryptography tools. For instance, the standard cryptographic algorithm AES (Advanced Encryption Standard) could be used to encrypt an XML document before sending it to the recipient. However, this operation would convert an XML text document to binary data. It would be no longer possible to use standard XML tools to process this document since it no longer can be parsed. In many situations, this is highly undesirable. XML-Enc does address and eliminate this problem.

Like conventional security technology, XML Encryption makes data confidential using cryptographic algorithms which garble data into a non-readable and non-understandable form by using a key. Only a person (or system) in possession of the proper key will be able to reconvert the encrypted data into a readable form. In general, encryption can be done in two different ways:

– Symmetric Encryption uses the same key for both encryption and decryption. This means, that both sender and receiver of the data must have the same key.

– Asymmetric Encryption is sometimes also referred to as »public key encryption« (because the encryption key can be made public without hampering security). Asymmetric Encryption uses different keys for encryption and decryption: a public key and a private key.

XML Encryption supports both encrypting technologies. It also is possible to apply confidentiality to entire XML documents, portions of XML documents, single XML elements and even single element content. Content also can be secured using different keys for different portions of the document, so parts of the document can be made accessible for different recipients. This is especially valuable when the document has to be processed by an intermediary before passed on to the final recipient. In this case, the intermediary only has access to those parts of the document that do concern him.
When XML Encryption is used on XML documents, the result is still an XML document that can be processed with standard XML tools (such as a parser). But XML Encryption is in no way limited to XML content. It also can be used to encrypt arbitrary content, including binary data. This means that binary attachments (e.g. pictures or sounds) can also be processed using the XML Encryption specification.

5.1.2 XML Digital Signature (XML-DSig)

Digital signatures are the electronic equivalent of handwritten signatures. They allow data to be verified as authentically created or sent by a certain person or system. If a person sends digitally signed data to a recipient, the latter can positively verify that the data hasn't been altered during transport. This is called **»Message authentication«**. Any alteration of digitally signed data would render the attached signature immediately invalid. The recipient can also use a sender's digital signature for **»nonrepudiation«**. This means, the digital signature can be used as an evidence that the sender really did create the data. After attaching his digital signature, the sender can no longer reasonably deny not to have signed the data. However, it should be not remain unsaid that nonrepudiation usually takes some instance of trust along with the digital signature itself. This is due to the fact, that despite it is easy to show that a certain digital signature has been used to sign a message, it is much harder to show that this digital signature really belongs to a certain individual. This only can be ascertained when the digital signature has been issued by a trustworthy instance (such as a government body), that has positively verified the individual's identity in the process of issuing the digital signature.

A digital signature is made by computing a digital »fingerprint« (also called a »hash« or »message digest«) of the message to be signed. This fingerprint is then encrypted with the signer's private key and attached to the message. To verify the signature, the recipient computes the fingerprint of the message again and decrypts the one in the signature with the sender's public key. If the two fingerprints are identical, the message is authentic. If the message was altered, the fingerprint of the message will differ from the one in the signature.

Just like XML Encryption, XML-DSig can be used to sign entire XML documents or portions thereof. Usage of this specification is also not limited to XML documents, for binary data also be signed using XML-DSig. Nested signatures are supported as well. This makes it possible to sign a digital signature with another signature which can be used to ascertain trust. For instance, a trustworthy instance could sign an individual's digital signature with their own to ascertain a recipient that this digital signature has been verified by them as authentic.

5.1.3 Extensible Access Control Markup Language (XACML)

The Extensible Access Control Language is a policy language that can be used to enable access control to XML documents. Usually, access control models involve a user making some access request and the system either authorises this access request or denies it. This is called a subject-privilege-object model.

With XACML it is possible to regulate access to complete XML documents or portions thereof (such as single elements). Also it can be specified which actions on the document or its portions are allowed and which are not. XACML is not limited to XML documents, however. XACML policies can regulate any type of resource via an XPath expression.

When XACML is applied, a processor is needed that evaluates each user request to a target XML document and makes an access decision based on the associated policies written in XACML. If the access is to be granted, the request will be executed and the results delivered to the requester.

5.1.4 Security Assertion Markup Language (SAML)

A general requirement of modern distributed systems is »single sign-on« authentication. This means that a user should only need to authenticate once to the system and the authentication information is shared throughout the entire system to avoid repeated authentication procedures.

The XML Security Assertion Markup Language enables »single sign-on« functionality by providing an XML vocabulary for sharing security assertions. SAML also includes a request/response protocol and a SOAP protocol binding. SAML does create assertions that a specific subject was authenticated by a specific mechanism at a specific time. Mechanisms that can be used in conjunction with SAML range from simple password verification to biometric attributes.

SAML assertions are compounds of one or more of three kinds of »statement« about a »subject« (which can be either a human user or a system):
– Authentication
– Attribute
– Authorisation decision

5.1.5
XML Key Management Specification (XKMS)

The XML Key Management Specification (XKMS) provides protocols for management of public keys. Public key technology is an essential part of digital signatures. It also plays an important role for providing confidentiality in distributed systems.

Important steps in applying public key cryptography include the creation of a public/private key pair and associate the public key with identity information of its owner. In case the private key gets compromised (e.g. by theft), a mechanism to revoke the thereby also compromised public key is needed, too.

XKMS defines XML messages for request, response, key registration, key revocation and key updates. The primary objective is to allow a user of a public key application to easily locate required keys and underlying key information. XMKS consists of two parts, the XML Key Information Service Specification (X-KISS) and the XML Key Registration Service Specification (X-KRSS). X-KISS defines protocols for processing key information (such as locating keys or key holder information) while X-KRSS provides support for key registration services. The registration process mainly consists of sending a public key along with the key user information to a trusted key registration server. After registration, key information may be obtained using X-KISS messages to send queries to the key server.

5.1.6
Platform for Privacy Preferences (P3P)

The Platform for Privacy Preferences (P3P) is a protocol developed by the World Wide Web Consortium (W3C). P3P is an XML format by which Web sites can describe their privacy policies in a machine readable format, e.g. whether personally identifiable data is being collected, the purpose for this collection and which third parties will have access to the collected data. A client (such as a Web Browser or a Web Service) can parse the P3P policy file and then make the decision of whether or not to use this service.

5.2
Web Services Security

The point of Web Services technology is to let any system communicate with any other system by the use of easy-to-implement and widely spread protocols. However, with these new possibilities comes a whole range of security risks that need to be addressed. Web Services technology will be applied in an environment that is highly decentralised in both architecture and administration. Also it is a extremely heterogeneous environment in regard to the applied implementation technologies. Last, the Web Services environment is one where services are regularly open to the public Internet. Enforcing

security policies across a highly decentralised and heterogeneous environment is no piece of cake.

A few of the major security risks that apply to Web Services are:

Denial of Service Attacks:
A Denial of Service (DoS) attack aims to disable a system by flooding it with more traffic than it can handle. With conventional web sites, DoS attacks are relatively easy to detect. However, in a Web Service environment, detection of DoS attacks might be more difficult. Especially when the Web Service takes a lot of computing time, it is entirely possible that only a few dozen requests might disable such a system. However, to safely detect a DoS attack by automated intrusion detection systems, many more requests are necessary.

Malicious Attacks:
Web Services are all about calling functions on remote systems. They do this by sending SOAP requests over the standard HTTP protocol (or other Internet protocols). The standard HTTP port is open at most firewall systems - which means that the SOAP traffic can pass unhindered to the internal network (or at least to the demilitarised zone). It must be clearly stated that the possibility to execute function calls on remote systems opens this system to hacker attacks who could use the Web Services interface to try to gain entry to the system. It is highly advisable to carefully design the Web Service (and its back-end applications) with regard to prevent hackers to exploit weaknesses and thus compromising the entire system. Also, all incoming SOAP requests should be parsed and evaluated before passed on.

Replay Attacks:
A replay attack is to copy a valid request and sending it to the recipient repeatedly. An example for an application of a replay attack would be to copy a valid bank order and sending it to the bank a couple of times to make the bank transfer many times the original sum to you.

Replay attacks can be pretty much prevented by including a timestamp and a digital signature to all requests.

Man-in-the-Middle Attacks:
A man-in-the-middle attack is when a malicious attacker intercepts communication traffic (such as SOAP messages) between two parties and modifies, deletes or totally replaces the messages. By doing so, the attacker can forge messages to make communication partner A think that he is talking to partner B while in fact he is talking to the attacker and vice versa.

A public key infrastructure with a trusted key issuer can help to prevent such attacks.

Buffer Overflow Attacks:
Buffer overflow attacks try to alter an application's function call stack to either make the system crash or execute code. This is done by calling a function and passing longer parameter values than the function can handle (e.g. a password with 200,000 characters). It is sometimes possible for a hacker to include code of his own to the function call by passing it as a parameter value and then make the application execute this code by using a buffer overflow attack.

Password Attacks:
As with any other system that can be protected

by password authentication, Web Services are also vulnerable to the same attacks known from past experience. A password attack is the try to access password protected services by guessing a valid user/password combination. This could be done either by brute force (trying out every possible password) or an »educated guess« such as a dictionary attack that tries a list of popular used passwords.

5.2.1 WS-Security (WSSL)

SOAP, the core Web Services specification, does not include any security mechanism. However, the listed XML security specifications can of course also applied in conjunction with SOAP. Moreover, a set of SOAP extensions has been created that helps implementing confidentiality and message authentication for SOAP documents. This specification goes by the name »WS-Security«, sometimes it is also referred to as the »Web Services Security Language« (WSSL). WS-Security supports encryption, digital signatures, trust domains and security tokens (such as identity credentials like a user name). It is largely based on both the XML-Enc and XML-DSig specifications. WS-Security provides a mechanism to attach a security header block to a SOAP header which can include security relevant information such as digital signatures or security tokens. Because multiple security headers can be included, WS-Security also supports intermediary security processing.

The WS-Security specification has recently been updated to be able to support message timestamps. These can be used to effectively prevent replay attacks. With a timestamp, a system can easily determine if a message has been sent more than once.

5.2.2 The future of Web Services security

With WS-Security, a powerful mechanism to ensure message integrity and confidentiality has been created. Currently, research focus on creating a more complete security architecture for Web Services. The creators of WS-Security, IBM and Microsoft, plan to integrate WS-Security in a broader range of security specifications.

The following figure shows the proposed roadmap by IBM and Microsoft (see: Security in Web Services World: A proposed Architecture and Roadmap. IBM Corporation and Microsoft Corporation; 2002):

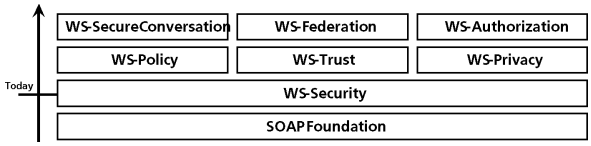


Figure 10:
Future of WS-Security (Source: IBM and Microsoft)

- WS-Policy will describe the capabilities and constraints of the security (and other business) policies on intermediaries and endpoints (e.g. required security tokens, supported encryption algorithms, privacy rules).

- WS-Trust will describe a framework for trust models that enables Web services to securely interoperate.

– WS-Privacy will describe a model for how Web services and requesters state subject privacy preferences and organisational privacy practice statements.

– WS-SecureConversation: will describe how to manage and authenticate message exchanges between parties including security context exchange and establishing and deriving session keys.

– WS-Federation will describe how to manage and broker the trust relationships in a heterogeneous federated environment including support for federated identities.

– WS-Authorisation will describe how to manage authorisation data and authorisation policies.

5.3 Usage Scenario 2: Security aspects

In the following example a Web Services scenario with considerable security needs will be presented. It will be illustrated how the different specifications can be implemented to address the arising security needs.

»Smith Company« plans to purchase several new machines at »Machine Company«, a vendor that sells expensive machines over the Internet. Therefore, »Machine Company« requires a credit check before a transaction can be done. Of course, »Smith Company«'s credit rating is sensitive data, so that it has to be treated confidential. »Smith Company« has its accounts at »The Bank«. The forth player in this scenario is the »Trust Company« which provides a public key infrastructure. »Trust Company« also operates an XKMS key registration server.

That is how it can be done:

1 The IT departments at »Machine Company«, »Smith Company« and »The Bank« each create a public/private key pair that can be used for digital signatures or public key cryptography.

2 All three register their public keys with »Trust Company«'s XKMS server via X-KRSS.

3 »Smith Company« retrieves »Machine Company«'s public key at »Trust Company« via X-KISS.

4 »Smith Company« creates a SOAP document containing the purchase order and digitally signs it with »Machine Company«'s public key. The signature information is embedded in the SOAP Header according to the WS-Security / XML-DSig specifications. Thereafter, the purchase order is sent to »Machine Company«.

5 »Machine Company«'s order processing system retrieves »Smith Company«'s public key at »Trust Company« via X-KISS and validates the digital signature on the purchase order to ascertain that it is authentic.

6 »Machine Company«'s order processing system sends an access request to »The Bank«'s credit rating database and signs it with its digital signature.

7 »The Bank«'s credit rating system retrieves »Machine Company«'s public key at »Trust Company« via X-KISS and verifies the digital signature on the access request to ascertain that it is authentic.

8 »The Bank« charges »Machine Company« for accessing its database and sends a SOAP document containing a SAML assertion to »Machine Company«.

9 »Machine Company« retrieves »The Bank« public key at »Trust Company« via X-KISS.

10 »Machine Company«'s order system creates a SOAP request to »The Bank«'s credit rating database, encrypts the payload containing »Smith Company«'s data via WS-Security / XML-Enc with »The Bank«'s public key. Then the order system attaches the SAML assertion to the SOAP header and digitally signs the document with »Machine Company«'s digital signature.

11 »The Bank«'s credit rating database verifies »Machine Company«'s digital signature and the SAML security assertion.

12 »The Bank«'s credit rating database decrypts the SOAP payload using its own private key.

13 Satisfied that both the signature and the SAML assertion are valid, the credit rating system at »The Bank« creates a SOAP document containing »Smith Company«'s credit rating, which of course gets encrypted via WS-Security / XML-Enc using »Machine Company«'s public key. The digitally signed SOAP document is sent to »Machine Company«.

14 »Machine Company«'s order processing system verifies the digital signature and decrypts the SOAP payload containing the requested information.

15 Satisfied that »Smith Company«'s credit rating is »AAA«, the order processing system sends a SOAP document containing a digitally signed order confirmation to »Smith Company«'s procurement system.

The described process is illustrated in the following graphic:

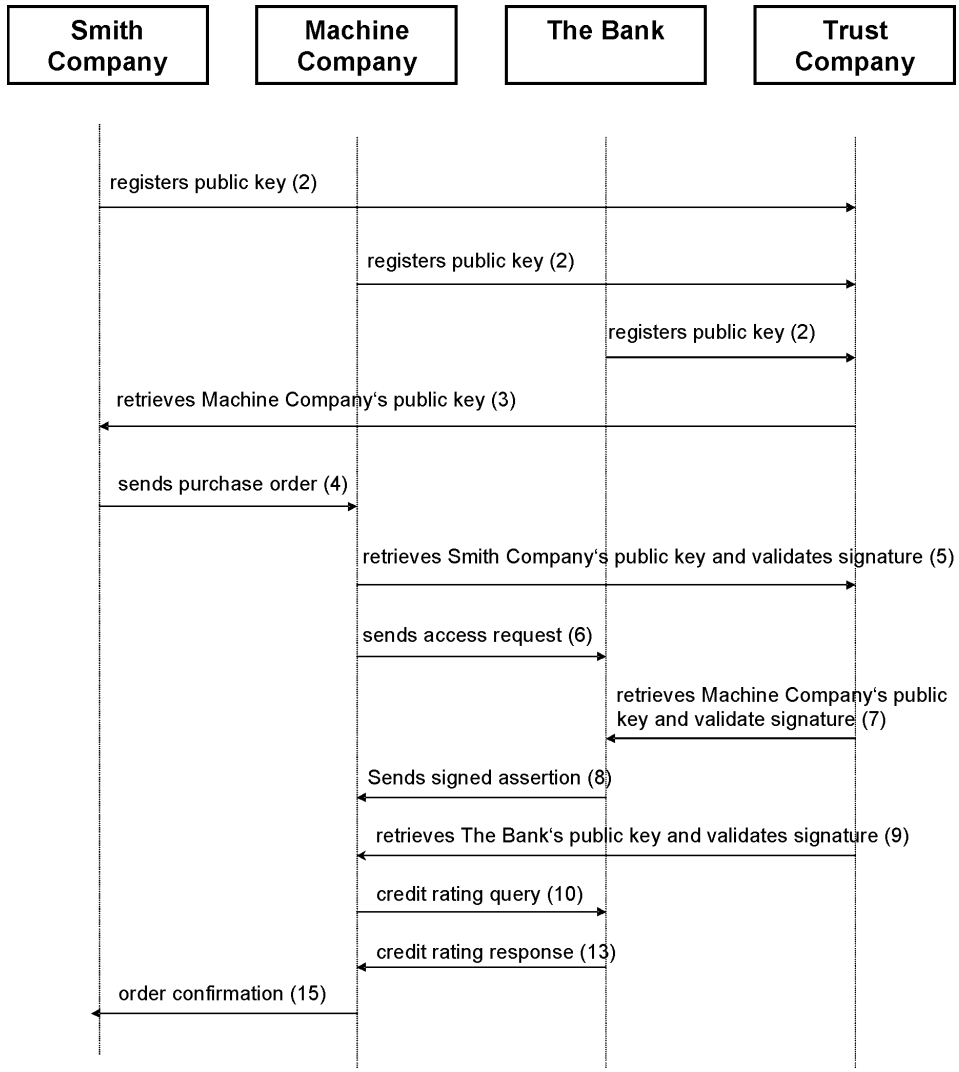


Figure 11:
Process flow of the usage scenario

5.4 Conclusion

Security is no quality characteristic that can be optimised and improved over time. Security is nothing less than a hard prerequisite for implementing Web Services technology in a business environment. While it is true that some business scenarios do not have equally strong security needs as others do, this does not change the fact that every scenario's security requirements have to be covered, or Web Services cannot be applied to this scenario at all. There is no such thing as »half-good« security. Either your application is sufficiently secure or it is not. There is no in-between.

The security need for Web Services has been addressed by a large number of XML and WS security specifications.

Authentication / Authorisation is covered by SAML and XACML. These two specifications are fully compatible and can probably solve any possible use case for authentication / authorisation.

Data Integrity is covered by XML-DSig and XKMS, which provides mechanisms for digital signatures and public key management.

Confidentiality is covered by XML-Enc which allows a flexible application of proven cryptographic methods to Web Services.

WS-Security combines some of these general XML security approaches into a security architecture specifically for Web Services applications.

By the use of named specifications these dimensions of security can be covered – in theory. However, at this time, there are almost no products available on the market that implements these specifications in a complete manner. Some of these specifications are still too new to be considered stable. It is still possible that some of them will undergo significant changes in the future – what might be a reason that many companies are somewhat cautious not to implement a product based on these specifications too early.

It has to be clearly stated that before products are not available on the market Web Services security will not happen at a large scale. In conjunction with the fact that lack of security is one of the largest obstacles for implementing Web Services in a business environment, it should be clear that resources must be assigned to this matter with high priority.

- The security specifications relevant to Web Services have to reach a sufficient degree of **stability** to enable developers to apply these specifications in their products. However, it can be safely assumed that this will happen in the near future, because most of these specifications have in the meantime been proposed to standardisation organisations like OASIS or the W3C.

- **Security awareness** has to be created. This means that Web Services vendors have to be made aware of the security risks their services are vulnerable to and the measure they can implement to address these risks. Past experience has showed us that some vendors are strong at security while others are not. In the

Web Services world where applications and systems become more tightly integrated than ever, such weak spots will become more dangerous than ever.

– **Tools** will have to be created that address all these security risks that the XML security specifications (and products based on them) alone can not. For instance, we need security tools that can parse SOAP documents for malicious content or tools that can be used for intrusion detection in a Web Services environment.

A lot of attention will also be required for **carefully designed back-end applications.** If a back-end application is vulnerable to buffer overflow attacks or if passwords are badly chosen, the system still can easily be compromised, even if XML security is implemented.

6 Potentials of Web Services

With regard to an assessment of the degree of maturity of the Web Services market, it must be stated that there still is no highly developed Web Services market. After comprehensive research it was not possible to find real-existing cross-company business applications that could be identified as Web Services according to the definition of W3C. A respectable amount of companies seem to be experimenting with Web Services technology, for example in behind-firewall EAI scenarios, but are still far from operating a mature solution that could be categorized or classified in one way or the other. The bottom line is: At this moment, the focus of Web Services related activities lies in the development and provision of tools and platforms that support Web Services technology. For example, a lot of software developing companies extend their product suites to support Web Services technology (such as IBM and SAP) or develop entirely new products (for example IONA). However, a real market, as has developed for other e-business models such as ASP (Application Service Providing), has not emerged yet for real-world applications of Web Services technology in cross-company scenarios.

6.1 Application fields

By means of Web Services technology companies may provide internal and external users with access to electronic business functions. This may take place on a commercial basis or to foster better business-to-business and business-to-consumer relationships. Most certainly, Web services will support this goal by providing new business opportunities and new business models. The introduction of software/business functions-as-a-service will present an opportunity for businesses to use new ways of delivering and gaining value.

There are a couple of application fields which seem to be particularly suitable for the use of Web Services. These application fields already pose strategic challenges on enterprises aiming at extending business domains by means of the Internet and designing and establishing new (Internet-based) business models by means of combined technological approaches.

Business process automation

Web services reduce the need to carry out separate integration projects with each partner, customer, or supplier an organisation wishes to conduct automated business with, thus minimising integration costs and efforts. This objective is accomplished by a high degree of standardisation as to the definition of interfaces, making externally available business logic that was previously restricted to be used in backend systems.

Business-to-business integration

Web services make it simpler for an organisation to interact with other businesses and syndicate the functions it provides or aggregate the functions provided by others. Organisations will find greater freedom to outsource not only IT functions but also entire parts of the organisation to specialised providers, reducing the functional scope of the organisation and enabling it to concentrate on its core competencies.

Enterprise application integration

Today's enterprise application integration products are typically point-to-point solutions built to integrate specific products with each other. Web Services will make integration easier and force an emphasis on the subject of interoperability, thus making it easier to work with partners and to integrate systems.

Software engineering / Application development

With Web Services businesses will have more choice as to how they engineer applications. Instead of being faced with the alternatives of either keeping all IT in-house or using the services of an Application Service Provider (ASP) to deliver all (or a substantial part) of its IT services, an organisation can select a mix of in-house and outsourcing with complete freedom to mix and match.

6.2 Implications for demanders of Web Services

There are two main business implications of Web Services: **outsourcing** (by the company that uses Web Services based functions of another company) and **externalisation of business functions** (by the company that offers the Web Services based functions), both on a much finer granularity level than possible today.

6.2.1 Deployment of Web Services along the entire value chain

Web Services can be implemented along the entire value-adding chain – not only on certain stages. On every stage where the externalisation of transactions and (sub-) processes may lead to cost savings and an increase of efficiency, Web Services could be applied in principle with positive impacts.

Web Services, which are integrated as external services in the process structures of an organisation and which perform clearly defined tasks and process steps, can yield a significant increase of efficiency. Organisations using Web Services can concentrate more on their core competencies while at the same time being able to offer customers additional services which have not been developed in-house and which do not necessarily belong to the organisation's core business field.

The following sections contain usage scenarios that shall illustrate this aspect by showing business cases from different areas of the value chain.

Optimisation of distribution channels

In order to optimise distribution channels, a carrent company connects business partners (e.g. travel agencies) to its electronic reservation system by means of a Web Service. In the past, electronic bookings were made over EDI, a very extensive and cost-intensive method. The configuration of the interfaces was quite time-consuming and had to be adapted each time the system of one of the partners was modified. Now the company has the possibility to offer partners a single Web-based interface, resulting

in significant technical and organisational improvements. Due to much lower costs for implementation and maintenance, it is now possible to connect much more business partners with the reservation system.

Opening up new business potentials
A car rental agency plans to open up new business potentials and offers its core competence, which is in the area of vehicle fleet management, as a new offer on the Internet. Among other things, the offer consists of several services running on different back-end systems that have to be integrated into the newly developed customer oriented enterprise portal by the help of Web Services technology. This portal can be used by customers, e.g. for the conclusion of new contracts. Via Web Services an integration of a customer's ERP application with the vehicle fleet management system of the car rental agency can be implemented.

Qualification of suppliers
In the qualification process of suppliers there often is the necessity to assess suppliers in order to optimise or enlarge the company's supplier portfolio. An individual assessment of every single supplier, if done by the company itself, would be costly as well as elaborate. In this context, external, specialised service providers promise an increase in process efficiency and thus reduction of costs. These services could of course also be offered on the Internet. By the help of Web Services, the necessary integration of external service providers with internal systems can be realised easily and quickly.

6.2.2
Usage Scenario 3: Business application

A credit rating company offers a Web Services based application for credit assessment of persons and companies. Third parties (e.g. Web Shops, banks, mobile communication companies and so on), can use this service for assessing new customers or other business partners. This service could be directly called from the ERP system and thus tightly integrated with the process of recording the new customer's data or the order processing system. Depending on the result of the credit assessment, the order can be automatically processed or rejected.

Here is an example how it works today without Web Services:

1 John Black has decided to purchase a mobile phone. He surfs through the Internet in order to search for suitable offers.

2 After some time he decides on an offer of »Communications Unlimited«, a mobile-phone service provider. He is a bit annoyed that Communications Unlimited offers no possibility to complete the contract online. But since the offer seems to be very suited to his needs, he accepts that he has to go to the nearest branch office of the company.

3 The sales assistant in the branch office asks John to fill in and sign the order form. After that the sales assistant has to enter John's data into the customer registration mask of the ERP system of Communications Unlimited.

4 Before John can be accepted as a new customer of Communications Unlimited, the general business conditions of Communications Unlimited require that a solvency check has to be made. Therefore an inquiry is made to »Prime Financials«, a finance service provider with which Communications Unlimited holds a steady partnership.

5 The sales assistant enters John's data into a form and faxes it to Prime Financials.

6 Thirty minutes later the sales assistant gets a phone call from a staff member of Prime Financials who asks for additional information about John's credit-worthiness. John is still waiting.

7 A few minutes later, Communications Unlimited receives a response-fax containing all the information necessary for completing the contract. The sales assistant concludes the customer registration process and John is ready to take his new mobile phone and leave the store.

The example is characterised by the extensive and inefficient execution of the customer registration process, with media discontinuity leading to high transaction costs. A major problem lies in the low degree or complete lack of interconnectivity of the systems involved. A real-time execution of the processes is thereby impossible.

Such difficulties provide for a decrease of customer satisfaction. The described scenario is by far not uncommon in reality, since efficient electronic integration of branch offices of mobile-phone service providers with information systems of external suppliers is still a problem. Here, Web Services offer cost-efficient and technically feasible possibilities to improve the customer registration process, with a number of actors to benefit from these potentials.

That's how it could work with Web Services (security and quality aspects are neglected in this example; please refer to respective chapters for details):

1 Prime Financials decides to establish the start-up »ABC Check«, which is supposed to offer its core competence of conducting solvency checks of customers on the Internet. A Web-based business model, built upon Web Services technology, is supposed to serve as the basis for the operative business of ABC Check.

2 ABC Check has its Web Service registered in the UDDI directory of a renowned organisation called »World Wide Registry«.

3 Communications Unlimited plans the automation of customer solvency checks in order to save transaction costs and accomplish an increase of efficiency with regard to the process of customer registration. The company decides to integrate that process as an external service by means of a Web Service. In order to find an adequate Web Service, Communications Unlimited contacts World Wide Registry. After the descriptions of various Web Services have been examined, Communications Unlimited decides on the Web Service of ABC Check.

4 By means of Web Services technology, Communications Unlimited is able to realise the technical integration of the Web Service with the company's ERP system.

5 Back to John Black: In the branch office of Communications Unlimited, the sales assistant captures John's data and activates the solvency check by a simple click on a button directly in the customer registration mask of the ERP system.

6 Immediately, a SOAP document is created and sent to ABC Check over the Web Services platform of Communications Unlimited. ABC Check receives the SOAP document and examines John's credit-worthiness in real-time over a database inquiry. The result (again as a SOAP document) is sent back to Communications Unlimited.

7 The ERP system of Communications Unlimited receives and interprets the result of the solvency check. Depending on this result, the customer registration process is continued accordingly.

The process requires no manual activities anymore! Although illustrated in a simplified fashion, it becomes obvious that the customer registration process is much more efficient with than without the use of Web Services:

– Prime Financials makes available its core competence on new level, thereby establishing new business fields.

– By means of Web Services, business relationships can be established and dissolved ad-hoc. Prime Financials gets the possibility to connect any business partner with its systems flexibly and cost-efficiently, since the use of Web Services does not cause dedicated implementation and maintenance costs (business process automation).

– New business opportunities arise for both new and established (technology) companies. The need for a global registry calls for new business models allowing to offer users of Web Services an integrated services spectrum. For example, World Wide Registry may offer an added value for their customers by letting registered Web Services undergo a standardised and accepted certification process in order to be able to rate Web Services (for example as outlined in chapter 3).

– By automating and outsourcing of sub--processes, Communications Unlimited is able to concentrate on their core competencies. With Web Services technology, the service offered by ABC Check is easily and cost-efficiently integrated into Communication Unlimited's internal process structures. Further savings can be accomplished by the fact that no development and maintenance efforts are necessary.

– The seamless integration of ABC Check's services with Communication Unlimited's ERP system allows the customer registration process to be handled without media discontinuity and time delays, making it possible for the mobile-phone service provider to accomplish successful customer relationship management.

– John Black gets a transparent impression of the cross-company process. He is satisfied by a customer-oriented service in which he needs not bother about long waiting-times. His loyalty with the supplier is thereby enhanced.

The scenario about solvency checks of customers has been chosen to offer a closer look at one possible application field where Web Services could lead to considerable improvements. We briefly outlined other scenarios before, but it must become clear that Web Services are not restricted to certain application fields, respectively to certain value added stages. Rather, we should conceive of Web Services implemented along the entire value-adding chain. Where externalisation of clearly defined transactions and processes may lead to cost savings and an increase of efficiency, the use of Web Services appears to be reasonable.

The John Black scenario also makes obvious, however, that certain requirements and problems of handling business processes cannot be optimised and solved, respectively, by Web Services. Additional problematic aspects must be addressed, if Web Services are to become a broadly accepted technological solution for conducting business:

– new legal aspects arise that must be dealt with by legislators (e.g. digital signature),
– reconfiguration of process structures,
– new quality of security requirements,
– quality assurance /certification of services,
– technologically and conceptually optimised billing mechanisms based on new business models (e.g. pay per use),
– psychological barriers caused by a new technological approach,
– Web Services do not solve logistics problems (e.g. John Black still has to go to the branch office).

So, a number of questions arise, which are still largely unanswered. If these aspects are neglected, however, the Web Services approach is in danger of ending in failure before even having the chance to achieve a critical level of dissemination and acceptance. The scenario above has deliberately not taken into account legal aspects and other framework conditions regarding the commercial use of Web Services. In this context, it is an indispensable requirement that binding framework conditions for a broad use of Web Services are established, in order to foster Web Services both by politically motivated (e.g. the EU) and market-driven, broadly accepted instruments. What is needed is a close cooperation of different actors, which are directly or indirectly concerned by this new technology. Let us first have a look at the actors that will use and provide Web Services on a commercial basis.

6.2.3 Possible benefits for organisations using Web Services

Both business-to-consumer and business-to-business relationships are likely to change substantially as a result of the use of Web Services. By publishing services in a standardised form, demanders get the opportunity to quickly and efficiently find suitable business opportunities. On the hand, this may lead to situations where organisations move away from one business partner to another more easily and quickly, which may result in a decrease of customer loyalty and – as a consequence – in an increase of competition.

However, the use of Web Services may also yield positive changes with regard to existing partnerships. Here are some examples:

- The mere electronic transfer of data does not necessarily mean that organisations may achieve a significant improvement with regard to process efficiency. In order to benefit from shorter processing times, shorter delivery times, lower warehousing costs etc., automatic processing of incoming data by the addressee's systems and automatic generation of outgoing messages by the sender's systems is required. Web Services offer cooperating businesses a possibility to directly get access to certain areas of the partner's systems.

- Distribution cooperations of companies of the same industry and of the same value-adding stage, respectively, may benefit from scale effects and specialisation effects by implementing a Web Service in order to make available a common offering of all product variants. Orders received by the Web Services based application would then be passed on directly to the company most adequate to deliver the demanded products. If one company faces capacity overload, the Web Service allows to pass on the inquiry to a partner organisation that offers identical products and services.

- Companies that participate in a so-called Virtual Enterprise have the advantage to bundle single products and services into customised packages of products and services, thereby providing customers with a considerable added-value. Here, a network of Web Services could make it possible for individual companies to contribute special services which are then put together in an overall Web Service and offered to specific customers. Particularly, SMEs, which frequently are not able to provide complete solutions, would benefit from such a possibility.

6.2.4 Necessary requirements to effectively realise the benefits

In order to support complex business processes by means of Web Services, products, services and competencies are required by which an integration of single Web Services into an overall, process-oriented application is possible. The following tasks are relevant:

- Orchestration of Web Services: Single Web Services are combined to form complex Web Services based applications. Ideally, such orchestration tasks can be done by tools that have a graphical user interface, thereby allowing also persons with no or little programming skills to combine Web Services in order to get the desired functionality.

- Location of Web Services: Products and services are required to facilitate the process of finding and selecting Web Services. Such offerings could also include quality checks of Web Services. Possible providers of such services can be operators of electronic marketplaces or other electronic platforms.

- Reengineering of business processes: Existing business processes must be redefined and – if needed – reconfigured in order to make sure that they are suitable of being used

in connection with Web Services. Such modifications can be extremely difficult and usually demand extensive consulting.

– Need to establish new technological skills: The adoption and operation of Web Services will in most cases demand that new competencies need to be developed by an organisation's staff. Particularly IT departments will be confronted with considerable changes.

All these necessary requirements for establishing the Web Services concept on a broad basis will bring about new opportunities for basically all kinds of actors in the IT market.

6.3 Implications for suppliers of Web Services

The development and dissemination of Web Services is currently observed by market participants of various domains with the hope to be able to take advantage of new business opportunities. Particularly, technology suppliers and consulting companies, both well-established ones and new ones, see the chance to enter new business fields and follow new business models. Some of the possible implications that are likely to occur with the dissemination and adoption of Web Services technology are listed below.

Shift of roles

One consequence of the emergence of Web Services is that market participants are beginning to scrutinise their previous roles and trying to gain a foothold in new business domains. Several providers of electronic platforms have already begun to occupy additional business fields as system integrators or platform operators. Furthermore, what can be expected is a closer collaboration between system integrators and platform suppliers. As far as the role of software developers is concerned, it can be assumed that employees who until now have implemented individual interfaces may take over more complex integration tasks.

New business fields and business models

It is expected that new service-providing models will be established which allow Web Services suppliers to let certain administrative or operative tasks perform by an external provider. Those services could be offered by system integrators, management consultants, product suppliers, or integration service providers. Such services may be:

– the management of the Web Services life cycle (from development to operation to introduction of a new version; this comprises also dependence control and version control issues),
– security, transaction integrity and availability,
– risk management
– control of service-level warranty,
– quality evaluation,
– billing,
– consulting,
– service hosting,
– workflow management,
– logistics.

Particularly, services concerning the billing process around Web Services may turn out to be a real differentiator in the market. Web Services suppliers may offer their services either on a time-based or usage-based calculation. One can also think of agreements which guarantee the

use of a certain Web Service over a defined time period and at a defined price. Another possibility is that Web Services are financed by advertising revenues, as is the case with conventional Internet offerings today.

Development of new skills – Impact on job market

As integration projects become easier to execute due to the use of Web Services, the services level in such projects may play a lesser role in the future. However, this does not mean that the e-business-integration services sector will experience a decrease in importance, since lower costs for executing integration projects will lead to a stronger overall demand for integration expertise.

Companies that perform Web Services based integration projects need to develop new technological skills in their IT departments. From this necessity, new job profiles in the field of interface programming are likely to emerge.

New consulting services will particularly be needed for the efficient orchestration of Web Services as well as for the communication of single Web Services components in complex business processes.

Finally, the technical development of Web Services is another field that requires adequate know-how, thereby providing new opportunities on the job market.

General market impact

Provided that Web Services will experience sound acceptance and broad dissemination over a mid-term to long-term perspective,

1 EAI suppliers will be confronted with an increased market pressure, particularly with regard of the elements which serve the mere technical integration (e.g. adapters), because of the now possible combination of enterprise software and Web Services technology,

2 application-server suppliers may lose momentum in the end-customer market, mainly because application server are increasingly be offered free of charge, but also because of the growing importance of open-source initiatives in the integration market; however, application server may become a part of other integration products,

3 mainframe specialists will experience a further reduction of their market share,

4 e-business platform suppliers will gain importance, since integration projects will focus more on the mapping and management of business processes and not so much on purely technical integration issues anymore; those suppliers will also benefit from the fact that technical integration will become easier through the existence of Web Services technology.

7 Conclusions and Recommendations

7.1 Potentials and limits of Web Services

A main advantage of Web Services arises from the fact that this technology is based on well-known, proven Internet standards (HTTP, XML). That means that there are no insurmountable technical barriers that impede an adoption of Web Services. In order to implement a simple Web Service, only a standard Web server and a development environment are required that support XML and SOAP. Companies get the chance to collect experiences with Web Services without having to invest a lot of money, which is an important aspect in favour of Web Services particularly for SMEs. More comprehensive software architectures, such as CORBA, require much higher initial investments.

Web Services technology allows to flexibly and loosely couple single Web Services to compose a powerful distributed system. It can therefore be expected that Web Services help lower the development effort of distributed systems, especially of systems with low complexity. Due to the high degree of standardisation with regard to the design of interfaces, individual Web Services functionality may be substituted by other Web Services based components (of other business partners), without substantially penetrating running processes. By being able to design modular business processes, Web Services support the need to flexibly integrate or separate the systems of business partners.

However, it must clearly be said that Web Services technology, despite the high potentials which they undoubtedly provide, is not able to fully solve certain problems that occur when distributed systems are interconnected. For example, there are still no market-ready methods that ensure the security of transactions. Also, methods and products for controlling, measuring and billing of Web Services based transactions are far from offering satisfying solutions.

A prerequisite for a successful use of Web Services in integration projects is that the applications to be integrated must support Web Services technology. This means that functionality must be made available from operative systems (business logic) as Web Services and that receiving systems must be able to process the results of a Web Service. But especially at the beginning of any Web Services engagement, suppliers of enterprise software are expected to make available only part of the complex functionality over a Web Service.

With regard to converting and mapping issues, Web Services offer only partial solutions. By the spread of XML as the universal format for electronic data exchange, these requirements are partially addressed, but simplification can only be accomplished on the syntactical level. The difficult part of integration projects, the actual definition of the semantic data transformation, is not facilitated by Web Services.

Further difficulties can be found in the missing legal regulations regarding Web Services. In order to be used for commercial purposes, Web Services need accepted and established models for contract completion and contract execution.

7.2 Cost and efficiency potentials

Whether Web Services may yield cost savings depends on the concrete integration task. A decisive factor is to what extent Web Services can actually reduce the proportion of the work of service providers around integration projects. Due to the reluctance of potential users and the low degree of dissemination of Web Services at the moment, cost savings as a result of the use of Web services are not likely to occur in the near future.

Generally, it can be said that companies using Web Services will rather benefit from long-term advantages that can be realised by the flexibility of a homogeneous Web Services based IT infrastructure. However, those companies should always take into account that previously relevant business questions remain relevant, that those questions have to be dealt with now in a new context, and that other questions emerge. Companies that decide to employ Web Services must consider the following aspects:

- the definition of a business model,
- the establishing confidence among business partners,
- the agreement on skeleton agreements,
- the agreement on business conditions,
- the definition of target groups,
- the definition of price model and service degree,
- the balancing of the need to provide customers with customised solutions on the one hand, and the need for the supplier to achieve scale effects on the other hand.

Strategic and operative impediments do not just vanish only because innovative technology lowers the costs for providing or demanding services. The technology, however, allows the supplier or demander to stronger focus the business challenges of such activities. For example, it can be expected that the programming of source codes will become less important in Web Services using companies. Instead, it will become a crucial requirement for system developers to integrate system components of various origin along a process chain. This will lead to an increased demand for business-oriented, analytical skills on the part of system developers and other IT experts.

7.3 Standardisation

Why is standardisation so important? Standardisation provides the ground for interoperability between e-business partners and eliminates the need for bilateral agreements. Ad-hoc e-business relationships become much more efficient when built upon standardised conditions. Standardisation also makes possible a simplification of products. Products that can be handled more easily reduce the need for developers and users to acquire sophisticated skills for developing and utilising those products. Product prices and implementation costs shrink, which leads to further dissemination of the standards.

The behaviour of market participants is always a decisive factor for the success of standardisation activities. Also with Web Services, the big players in the market could modify recommended stan-

dards or use own standards in order to gain a dominant market position. Another crucial aspect will be whether relevant market participants turn away from the technical focus on standards and concentrate more on the business process requirements.

It must be considered that none of the specifications around Web Services (SOAP, WSDL, UDDI) are formal standards in terms of being published by a governmental standardisation authority like ISO or DIN. However, some specifications, such as SOAP, have been recommended by non-governmental standardisation organisations like W3C or OASIS.

The following standardisation groups and industry consortia specifically address Web Services:

- OASIS Web Services for Interactive Applications (WSIA) Technical Committee, Web Services for Remote Portals (WSRP) Technical Committee, Web Service Security (WSS) Technical Committee,
- Web Services Interoperability Organization (WS-I),
- W3C's Web Service activities,
- CEN/ISSS Electronic Commerce Workshop,
- ETSI: Specialist Task Force,
- IETF: Working group with W3C on digital signatures,
- Liberty Alliance: activity on federated network identity specifications,
- OMG: Business Enterprise Integration Domain Task Force.

Also, specifications such as Electronic Business XML (ebXML) or RosettaNet can be mentioned in this context, which are partially built on Web Services specifications or are compatible with these specifications, respectively. RosettaNet's activities in the field of electronic components and semi-conductor products (High Tech) as well as the activities of ebXML aim at defining a uniform vocabulary, descriptions and specifications, which are valid for all industries and cover also the semantic and business-process management level. Some standardisation initiatives recommend WSDL for describing interfaces and SOAP as the basic communication protocol.

However, at the moment these activities appear to be quite uncoordinated, which causes several problems. For example, for the description of a partner company three different sets of master data exist: UDDI Schema, RosettaNet Directory and ebXML Core Components. Furthermore, BPSS in ebXML and BPEL4WS (Business Process Execution Language for Web Services) offer different, partially incompatible approaches for modelling business processes in coupled Web Services.

7.4 Perspectives

The basic level of Web Services functionality (concerning transportation standards and data standards, i.e. SOAP, WSDL and UDDI) appears to yield a success story of Web Services. Other very important aspects of integration (security, scalability, licensing, warranting, authentication, reliable messaging standards etc.) have not yet achieved a sufficient degree of maturity so that at the moment Web Services cannot be recommended to be used for more complex purposes. Because

of that it is expected that – for the short term – Web Services will predominantly be implemented for simple solutions within organisations or for similarly simple, clearly defined business-to-business relationships, such as content syndication. Such projects will serve to develop and establish Web Services related know-how and pave the way for more complex tasks which take into account strategic aspects as well.

Provided that quality and security requirements will be met to a satisfying degree, Web Services are likely to be used soon in cross-company projects conducted by companies that have fostered long-standing relationships before. Regardless of such positive prerequisites, it must be considered that presently installed ERP systems usually are not Web Services capable, and that many enterprises do not have the resources to implement the basic functionality needed for the operation of Web Services themselves. In all probability, it will take some time until Web Services functionality will be made available to a significant extent.

As already illustrated in the present paper, a multitude of cooperation forms may emerge on the basis of Web Services. In order to provide for broad adoption of Web Services, proper business models must be defined and established which take into consideration the specific requirements and framework conditions related with Web Services (i.e. quality control, billing, orchestration etc.). This requires the development and distribution of applications and tools that allow to fulfil these tasks. Particularly, the field of Web Services orchestration seems to be a promising market segment for IT companies and consultants.

Compared to previous applications, the possibility to orchestrate Web Services allows a much better integration of the business process level, on the one hand, and software engineering, on the other hand. Systems reconfiguration in case of changing process can be handled quickly and flexibly. On the long run, the availability of performant tools for the orchestration of Web Services could force suppliers of standard enterprise applications to reconsider their business models. One possibility to combine different approaches could be to integrate best-of-breed Web Services with existing legacy systems.

In order to develop sound strategies with regard to a sustained use of Web Services, respective know-how and experience both on the part of decision-makers and employees has to be established. Small pilot projects, initially designed for inner-company purposes, could help build up competencies, and comprehensive project evaluation could yield valuable findings about technical interrelations and profitability aspects for future, large-scale cross-company projects.

As far as standardisation issues are concerned, it can be said that SOAP, WSDL, and UDDI can be considered as widely accepted specifications. What other Web Services related specifications will finally achieve broad acceptance cannot be predicted at the moment. Although there are organisations, such as WS-I or Liberty Alliance, which pursue the objective of achieving compatibility among various standardisation initiatives, it is not certain that such endeavours will be successful, since there is always one or more important market actors that do not participate in either of the standardisation initiatives.

On the user side, many companies are currently experimenting with SOAP, WSDL, and even UDDI. Early applications of ebXML can be found in the insurance sector. RosettaNet is used in high-tech companies like Cisco, HP and Texas Instruments. Other Web Services related standards have as yet not achieved considerable dissemination.

7.5 Recommendations

We have shown that – despite the fact that Web Services technology is a very promising approach – there are still many things left to do in order to help this new concept to succeed. Some aspects should be regulated by the market, but there are also several fields of action that should be addressed by public or governmental institutions.

As for the core technical specifications themselves (SOAP, WSDL, UDDI), meanwhile they can be regarded as quite stable. However, as with many other similar standards and specifications, it can be expected that they will further grow and change – at least to a certain degree. The standardisation process is already organised in the hand of well-known and respected organisations, like the W3C or OASIS. As for the question whether this standards should be regulated by a governmental body, the answer probably has to be »influence is desired – regulation is not«. It can be safely assumed that the market will regulate the technological aspects of Web Services quite well, especially since several important market players actively participate in the standardisation process. This does not mean that the **standardisation process** should not be **fostered and encouraged by political instruments** – on the contrary!

What definitely needs to be regulated by governmental bodies is a suitable **legal framework** that incorporates the additional legal needs that result from the new (or changed) business models and business procedures made possible by Web Services. Especially, regulations are needed that make international ad-hoc business relations with no underlying written agreements (that will become typical in a Web Services world) legal proof.

Another area of possible action by public institutions could be the **establishment and operation of Web Services registry services**. Currently, UDDI servers are operated by major market players for free. It cannot be said with certainty if and how this will change in the future. It also is not certain whether it is desirable that the operation of registry servers remains with companies that also offer Web Services themselves. And last, it cannot be ascertained that registry server operators will have a fair pricing model that offers non-discriminatory access to all market participants. In this case, it could become necessary to have neutral organisations operate registry servers. So, it could be guaranteed that no market player gains unfair competition advantages by operating – and thus possibly influencing – registries and registry entries.

It is also desirable to develop a **framework of commonly accepted quality** criteria and measurement indicators for these criteria that could be used to rate Web Services regarding their quality. A standardised certification process based

on this »quality framework« could also be helpful. Also, a »rating organisation« could be established that does the rating process according to the quality standards framework and issues certifications for complying Web Services. That could be a critical aspect for raising trust and acceptance of Web Services technology.

Regarding security aspects, it is considered reasonable to take measures that **raise security awareness** of all providers and users of Web Services technology. Adoption and application of high security standards are crucial for the success of Web Services in the business environment. Small and medium enterprises probably need special attention, since it can be safely assumed that often they neither have the budget nor the necessary expertise to apply state-of-the-art security techniques all by themselves.

Also, the wide-spread acceptance of Web Services will depend on the users' trust in the new technology. Trust can be achieved by minimised investment risks and sound sustainability. To help Web Services achieve this level of acceptance, a number of measures might be useful:

- online services for the user community to get informed on latest developments in the related area and to learn how to apply the technology (cp. the DIFFUSE project under http://www.diffuse.org for the area of electronic data exchange standards),
- combination of global and European fostering activities by governmental and non-profit organisations with local initiatives to bring the technology closer to potential user companies and make them leverage the benefits of the technology; the collaboration between the Munich Web Services Circle (http://mwsc.net) on a local level with the CEN/ISSS Web Services project group is a first step in the right direction,
- synchronisation of global and European policies on how to support emerging technologies with national strategies,
- preparation of usability guidelines that facilitate Web Services implementation and show how to integrate with similar approaches,
- synchronisation of related standardisation initiatives, such as RosettaNet, ebXML and Web Services initiatives.

Selected bibliography

Arkin, A. (2002): Business Process Modeling Language, http://www.bpmi.org

Berlecon Research, (2002): E-Business-Integration und Web Services -- Chancen und Perspektiven im deutschen Markt, http://www.berlecon.de/

BPMI, (2000): http://www.bpmi.org

Castro, E. (2002): The Web Services Community Portal – A perspective on Web Services, http://www.webservices.org/

CAP Gemini Ernst & Young, (2002): Der Markt für Web Services: Erwartungen, Treiber, Investitionsabsichten, http://www.de.cgey.com/servlet/PB/menu/1004619/index.html

Cape Clear, (2002):Web Service-Oriented Architecture: The Best Solution to Business Integration, http://www.capeclear.com/clear_thinking/Web_Service_Oriented_Architecture2.pdf

Cerami, E.: Web services essentials. Beijing: O'Reilly, 2002.

Clark, M.; Fletcher, P.; Hanson, J.; Irani, R.; Waterhouse, M.; Thelin, J.: Web Services Business Strategies and Architectures. San Francisco: Expert Press, 2002.

Curbera F.; Goland Y.; Klein J.; Leymann F.; Roller D. (2002): Business Process Execution Language for Web Services, Version 1.0, http://www.ibm.com/developerworks/library/ws-bpel/

Friedrichs, J.: Methoden empirischer Sozialforschung. Opladen: Westdeutscher Verlag, 1990.

Hada, S.: SOAP security extensions: digital signature; IBM; 2001; http://www-106.ibm.com/developerworks/library/ws-soapsec/

IBM: Web Services Security (WS-Security), 2002.
ftp://www6.software.ibm.com/software/developer/library/ws-secure.pdf. 2002-06-11.

IBM: Understanding quality of service for Web services, 2002.
ftp://www6.software.ibm.com/software/developer/library/ws-quality.pdf. 2002-06-11.

IBM: Web Services Flow Language (WSFL 1.0), 2001.
http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf. 2002-06-11.

Kotok, A.; Weber, D.R.R.: ebXML – The New Global Standard for Doing Business over The Internet. Indianapolis: New Riders, 2002.

Kromrey, H.: Empirische Sozialforschung. Opladen: UTB Leske&Budrich, 1998.

Leymann F. (2001): Web Services Flow Language (WSFL 1.0), http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf

Mactaggert, M.: Enabling XML Security. An introduction to XML encryption and XML signature. IBM; 2001; http://www-106.ibm.com/developerworks/library/s-xmlsec.html

McKinsey & Company (2002): The strategic value of Web Services, http://download.mckinseyquarterly.com/web_services.pdf

Menasce D.A.; Almeida V.A.F.: Capacity planning for Web services. Upper Saddle River: Prentice Hall PTR, 2002.

Microsoft (2002): XLANG Overview, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/biztalks/htm/lat_sched_concept_aind.asp

Newcomer, E.: Understanding Web Services. XML, WSDL, SOAP and UDDI. Pearson Education; Boston; 2002

OASIS: eXtensible Access Control Markup Language (XACML), 2003. http://www.oasis-open.org/committees/download.php/2406/oasis-xacml-1.0.pdf

OASIS: Security Assertion Markup Language (SAML), 2002. http://www.oasis-open.org/committees/download.php/2290/oasis-sstc-saml-1.0.zip

Oellermann, W.L.: Architecting Web Services. Berkeley: Apress, 2001.

OPENXCHANGE (2002): The openXchange Consortium: IST-2000-28548 Project Deliverable 2.1 – State-of-the-Art Analysis, 2002.

Schmelzer, R.: XML and web services unleashed. Indianapolis: Sams, 2002.

SUN Microsystems: Web Services Made Easier, 2001. http://java.sun.com/xml/webservices.pdf. 2002-06-11.

SUN Microsystems: Developing Web Services with SUN OpenNet Environment, 2002. http://wwws.sun.com/software/sunone/wp-spine/spine.pdf. 2002-06-11.

Sunsted, T. (2001): Building Security into Web Services, http://sunonedev.sun.com/building/tech_articles/webserv_security.html

Thatte S. (2001): XLANG Web-Services for Business Process Design, http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm

The Stencil Group: Defining Web Services, 2001. http://www.stencilgroup.com/ideas_scope_200106wsdefined.pdf

Thiagarajan R.; Srivastava A.; Pujari A.; Bulusu V. (2002): BPML: A Process Modelling Language for Dynamic Business Models, in : Forth IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, Newport Beach

World Wide Web Consortium (W3C): SOAP 1.2 Primer, 2003. http://www.w3.org/TR/2003/REC-soap12-part0-20030624/ 2003-07-09.

World Wide Web Consortium (W3C): Web Services Description Language (WSDL) 1.1, 2001. http://www.w3.org/TR/2001/NOTE-wsdl-20010315 2002-12-09.

World Wide Web Consortium (W3C): XML Encryption Syntax and Processing, 2002. http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/ 2003-06-10.

World Wide Web Consortium (W3C): XML-Signature Syntax and Processing, 2002. http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/ 2003-06-10.

World Wide Web Consortium (W3C): XML Key Management Specification (XKMS), 2001. http://www.w3.org/TR/2001/NOTE-xkms-20010330/

UDDI.org: UDDI Specification; 2002; http://www.uddi.org/specification.html

Appendix – Example Source Code

SOAP

An exemplary query on a Web Service that delivers stock quotes:

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml;
charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <m:GetLastTradePrice
      xmlns:m="Some-URI">
        <symbol>MOT</symbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The reponse from the Web Service could look like this:

```
HTTP/1.1 200 OK
Content-Type: text/xml;
charset="utf-8"
Content-Length: nnnn
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
  <SOAP-ENV:Body>
  <m:GetLastTradePriceResponse
    xmlns:m="Some-URI">
    <Price>14.5</Price>
  </m:GetLastTradePriceResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

WSDL

A WSDL description for the "stock quote" Web Service (taken from: Vasudevan, Venu; A Web Services Primer; 2001):

```
<?xml version="1.0"?>
<definitions name="StockQuote"
  targetNamespace="http://example.com/stockquote.wsdl"
  xmlns:tns="http://example.com/stockquote.wsdl"
  xmlns:xsd1="http://example.com/stockquote.xsd"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="http://example.com/stockquote.xsd"
      xmlns="http://www.w3.org/1999/XMLSchema">
        <element name="TradePriceRequest">
          <complexType>
            <all>
              <element name="tickerSymbol" type="string"/>
            </all>
          </complexType>
        </element>
        <element name="TradePrice">
          <complexType>
            <all>
              <element name="price" type="float"/>
            </all>
          </complexType>
        </element>
      </schema>
    </types>
<message name="GetLastTradePriceInput">
  <part name="body" element="xsd1:TradePriceRequest"/>
</message>
<message name="GetLastTradePriceOutput">
  <part name="body" element="xsd1:TradePrice"/>
</message>
<portType name="StockQuotePortType">
  <operation name="GetLastTradePrice">
```

```
    <input message="tns:GetLastTradePriceInput"/>
      <output message="tns:GetLastTradePriceOutput"/>
  </operation>
</portType>
<binding name="StockQuoteSoapBinding"
  type="tns:StockQuotePortType">
<soap:binding style="document"
  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="GetLastTradePrice">
      <soap:operation
soapAction="http://example.com/GetLastTradePrice"/>
        <input>
          <soap:body use="literal"
            namespace="http://example.com/stockquote.xsd"

encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
        </input>
        <output>
          <soap:body use="literal"
            namespace="http://example.com/stockquote.xsd"

encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
        </output>
    </operation>
</binding>
<service name="StockQuoteService">
  <documentation>My first service</documentation>
    <port name="StockQuotePort" binding="tns:StockQuoteBinding">
      <soap:address location="http://example.com/stockquote"/>
    </port>
</service>
</definitions>
  <binding name="StockQuoteServiceBinding"
    type="StockQuoteServiceType">
      <soap:binding style="rpc"
        transport="http://schemas.xmlsoap.org/soap/http"/>
          <operation name="getQuote">
            <soap:operation
soapAction="http://www.getquote.com/GetQuote"/>
```

```
                <input>
                  <soap:body type="InMessageRequest"
                    namespace="urn:live-stock-quotes"

encoding="http://schemas.xmlsoap.org/soap/encoding/"/>
                </input>
                <output>
                  <soap:body type="OutMessageResponse"

encoding="http://schemas.xmlsoap.org/soap/encoding/"/>
                </output>
            </operation>
  </binding>
  <service name="StockQuoteService">
    <documentation>My first service </documentation>
    <port name="StockQuotePort" binding="tns:StockQuoteBinding">
      <soap:address location="http://example.com/stockquote"/>
    </port>
  </service>
</definitions>
```

UDDI

The following example shows how to insert organisational information into a UDDI registry database via a API call (in this case the "save_business" function of the UDDI publisher API).

```
POST /save_business HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "save_business"
<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3c.org/2001/12/soap-
envelope>
  <SOAP-ENV:Body>
    <save_business generic="2.0" xmlns="urn:uddi-org:api_v2">
      <business_key="">
      </business_key>
      <name>
        Stockquote Inc.
      </name>
      <description>
      We know any stock quote
      </description>
     <save_business>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

A call of the API function "get_businessDetail" could look like this:

```
POST /get_businessDetail HTTP/1.1
Host: www.stockquoteserver.com
Content-Type text/xml; charset="utf-8"
SOAPAction: "get_businessDetail"
<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3c.org/2001/12/soap-
Envelope>
  <SOAP-ENV:Body>
    <get_businessDetail generic="2.0" xmlns="urn:uddi-org:api_v2">
      <businessKey ="100D0815-0815-D7E0-C3E0-53033007CD0E">
```

```
      </businessKey>
    </get_businessDetail>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In the example above, a "business key" is used to get detailed information about an organisation. The business key could have been retrieved, for example, by a prior call of the function "find_business".

XML-Enc

The following example shows a simple XML document and its encrypted counterpart:

```
<?xml version='1.0'?>
<CreditCardInfo xmlns='http://www.creditcardcompany.org/creditinfo'>
  <Name>John Doe</Name>
  <CreditCard>
    <Number>4000 6000 0815</Number>
    <Issuer>Bank of Pluto</Issuer>
    <Expiration>08/2199</Expiration>
    <Limit>5,000</Limit>
  </CreditCard>
</CreditCardInfo>
```

The completely encrypted version of this document would look like this:

```
<?xml version='1.0'?>
<EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
   Type='http://www.isi.edu/in-notes/iana/assignments/media-
   types/text/xml'>
  <CipherData>
   <CipherValue>A402EF03223CD3F7A2DF77E801324AE623DF7</CipherValue>
  </CipherData>
</EncryptedData>
```

A version of the document with only the credit card limit hidden:

```
<?xml version='1.0'?>
<CreditCardInfo xmlns='http://www.creditcardcompany.org/creditinfo'>
  <Name>John Doe</Name>
  <CreditCard>
    <Number>4000 6000 0815</Number>
    <Issuer>Bank of Pluto</Issuer>
    <Expiration>08/2199</Expiration>
    <Limit>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
       Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <CipherData>
          <CipherValue>C6EAA402EF0</CipherValue>
        </CipherData>
      </EncryptedData>
    </Limit>
  </CreditCard>
</CreditCardInfo>
```

XML-DSIG

The following example shows a simple detached digital signature (taken from Mactaggert, Murdoch: Enabling XML security. An introduction to XML encryption and XML signature, IBM developer works, 2001):

```
<Signature Id="MyFirstSignature"
  xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/
      REC-xml-c14n-20010315"/>
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/
      xmldsig#dsa-sha1"/>
    <Reference URI="http://www.w3.org/TR/2000/REC-xhtml1-20000126/">
      <Transforms>
        <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
          20010315"/>
      </Transforms>
      <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue>MC0CFFrVLtRlk=...</SignatureValue>
  <KeyInfo>
    <KeyValue>
      <DSAKeyValue>
        <p>...</p><Q>...</Q><G>...</G><Y>...</Y>
      </DSAKeyValue>
    </KeyValue>
  </KeyInfo>
</Signature>
```

The information that is actually signed is that in the »SignedInfo« element. »SignatureValue« contains the actual digital signature. The »KeyInfo« element indicates the key that's used to validate the signature.

XACML

The following example from the XACML specification shows a sample policy where any user with an e-mail in the medico.com domain is authorized to execute any action on any resource.

```
<?xml version=1.0" encoding="UTF-8"?>
<Policy xmlns="urn:oasis:names:tc:xacml:1.0:policy"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:policy
http://www.oasis-open.org/tc/xacml/1.0/cs-xacml-schema-policy-01.xsd"
PolicyId="identifier:example:SimplePolicy1"
RuleCombiningAlgId="identifier:rule-combining-algorithm:deny-overrides">
<Description>Medi Corp access control policy</Description>
<Target>
  <Subjects>
    <AnySubject/>
  </Subjects>
  <Resources>
    <AnyResource/>
  </Resources>
  <Actions>
    <AnyAction/>
  </Actions>
</Target>
<Rule
  RuleId= "urn:oasis:names:tc:xacml:1.0:example:SimpleRule1"
  Effect="Permit">
  <Description>
    Any subject with an e-mail name in the medico.com domain
    can perform any action on any resource.
  </Description>
<Target>
  <Subjects>
    <Subject>
      <SubjectMatch MatchId="
        urn:oasis:names:tc:xacml:1.0:function:rfc822Namematch">
      <SubjectAttributeDesignator
        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="urn:oasis:names:tc:xacml:1.0:data-
```

```
type:rfc822Name"/>
         <AttributeValue
           DataType="urn:oasis:names:tc:xacml:1.0:data-
           type:rfc822Name">medico.com
         </AttributeValue>
     </SubjectMatch>
    </Subject>
  </Subjects>
  <Resources>
    <AnyResource/>
  </Resources>
  <Actions>
    <AnyAction/>
  </Actions>
</Target>
</Rule>
</xacml:Policy>
```

SAML

The following example shows a sample SAML assertion that a subject named »John Doe« has logged on to »Smith Corporation« by using a password authentication method:

```
<saml:Assertion
  MajorVersion=”1”- MinorVersion=”0”
  AssertionID=”172.16.1.110.12345678”
  Issuer=”Smith- Corporation”
  IssueInstant=”2001-12-03T10:02:00Z”>
    <saml:Conditions
      NotBefore=”2001-12-03T10:00:00Z”
      NotOnOrAfter=”2001-12-03T10:05:00Z”>
      <saml:AudienceRestrictionCondition>
        <saml:Audience>…Some URI…</saml:Audience>
      </saml:AudienceRestrictionCondition>
    </saml:Conditions>
    <saml:AuthenticationStatement
      AuthenticationMethod=”urn:oasis:names:tc:SAML:1.0:am:password“
      AuthenticationInstant=”2001-12-03T10:01:00Z”>
      <saml:subject>John Doe</saml:subject>
    </saml:AuthenticationStatement>
</saml:Assertion>
```

WS-Security (WSSL)

This example illustrates how a username security token can be included to a SOAP message:

```
<SOAP:Envelope xmlns:SOAP="http://www.w3.org/2001/12/soap-envelope"
  xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/04/secext">
  <SOAP:Header>
    <wsse:Security>
      <wsse:UsernameToken>
        <wsse:Username>John Doe</wsse:Username>
        <wsse:Password>0815</wsse:Password>
      </wsse:UsernameToken>
    </wsse:Security>
  </SOAP:Header>
</SOAP:Envelope>
```

This more complete example (taken from the WS-Security specification) shows a SOAP document with a security token (in the form of a X.509 certificate), a digital signature and encryption of the contents in the SOAP Body.

```
<?xml version="1.0" encoding="utf-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2001/12/soap-envelope"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/04/secext"
  xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
  <S:Header>
    <m:path xmlns:m="http://schemas.xmlsoap.org/rp/">
      <m:action>http://fabrikam123.com/getQuote</m:action>
      <m:to>http://fabrikam123.com/stocks</m:to>
      <m:from>mailto:johnsmith@fabrikam123.com</m:from>
      <m:id>uuid:84b9f5d0-33fb-4a81-b02b-5b760641c1d6</m:id>
    </m:path>
    <wsse:Security>
      <wsse:BinarySecurityToken
        ValueType="wsse:X509v3"
        Id="X509Token"
        EncodingType="wsse:Base64Binary">
        MIIEZzCCA9CgAwIBAgIQEmtJZc0rqrKh5i...
      </wsse:BinarySecurityToken>
```

```
<xenc:EncryptedKey>
  <xenc:EncryptionMethod Algorithm=
    "http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
  <ds:KeyInfo>
    <ds:KeyName>CN=Hiroshi Maruyama, C=JP</ds:KeyName>
  </ds:KeyInfo>
  <xenc:CipherData>
    <xenc:CipherValue>d2FpbmdvbGRfE0lm4byV0...
    </xenc:CipherValue>
  </xenc:CipherData>
  <xenc:ReferenceList>
    <xenc:DataReference URI="#enc1"/>
  </xenc:ReferenceList>
</xenc:EncryptedKey>
<ds:Signature>
  <ds:SignedInfo>
    <ds:CanonicalizationMethod
      Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
    <ds:SignatureMethod
      Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
    <ds:Reference>
      <ds:Transforms>
        <ds:Transform Algorithm="http://...#RoutingTransform"/>
        <ds:Transform Algorithm=
          "http://www.w3.org/2001/10/xml-exc-c14n#"/>
      </ds:Transforms>
      <ds:DigestMethod
        Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <ds:DigestValue>LyLsF094hPi4wPU...</ds:DigestValue>
    </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>
    Hp1ZkmFZ/2kQLXDJbchm5gK...
  </ds:SignatureValue>
  <ds:KeyInfo>
    <wsse:SecurityTokenReference>
      <wsse:Reference URI="#X509Token"/>
    </wsse:SecurityTokenReference>
  </ds:KeyInfo>
```

```
      </ds:Signature>
    </wsse:Security>
  </S:Header>
  <S:Body>
    <xenc:EncryptedData
      Type="http://www.w3.org/2001/04/xmlenc#Element" Id="enc1">
      <xenc:EncryptionMethod
        Algorithm="http://www.w3.org/2001/04/xmlenc#3des-cbc"/>
      <xenc:CipherData>
        <xenc:CipherValue>d2FpbmdvbGRfE0lm4byV0...
        </xenc:CipherValue>
      </xenc:CipherData>
    </xenc:EncryptedData>
  </S:Body>
</S:Envelope>
```

BPEL4WS

The basic structure of BPEL4WS looks like as follows (? behind an element or attribute means that the use of an element or attribute is optional; + means that an element or attribute may occur several times, but at least once; * means that an element or attribute may occur several times, but may also occur not at all):

```
<process    name="ncname"                          targetNamespace="uri"
            queryLanguage="anyURI"?                expressionLanguage="anyURI"?
            suppressJoinFailure="yes|no"?          enableInstanceCompensation="yes|no"?
            abstractProcess="yes|no"?
             xmlns="http://schemas.xmlsoap.org/ws/2002/07/business-process/">
<partners>?
      <!-- Note: At least one role must be specified. -->
       <partner   name="ncname"           serviceLinkType="qname"
                  myRole="ncname"?        partnerRole="ncname"?>+
      </partner>
</partners>
<containers>?
      <!-- Note: The message type may be indicated with the
      messageType attribute or with an
      inlined <wsdl:message> element within. -->
      <container   name="ncname"    messageType="qname"?>
            <wsdl:message name="ncname">? ... </wsdl:message>
      </container>
</containers>
<correlationSets>?
      <correlationSet name="ncname" properties="qname-list"/>+
</correlationSets>
<faultHandlers>?
      <!-- Note: There must be at least one fault handler or default. -->
       <catch faultName="qname"? faultContainer="ncname"?>*
             activity
      </catch>
      <catchAll>? activity </catchAll>
</faultHandlers>
<compensationHandler>?
       activity
</compensationHandler>
activity

</process>
```

Appendix – Questionnaire

Introduction / General Comments

What is your notion of Web Services? (core components, functions, difference to other/previous concepts/terms)

Without going into detail: How do you assess the importance and relevance of Web Services, currently and in the future?

How do you assess the current discussion around Web Services? (suppliers, technology, standards, degree of dissemination, readiness and maturity of businesses with regard to providing/using Web Services ....)

Potentials (user side / supplier side)

Who will be the players around Web Services in the future? (e.g. ASPs, SMEs vs. large companies, new businesses vs. existing businesses with extended range of services ...)

What are the concrete expectations with regard to providing and using Web Services?

What sectors and industries will be particularly affected by Web Services?

What are the value-adding stages which may be particularly affected by Web Services?

What are the economic potentials an enterprise may develop through the provision or use of Web?

How will Web Services affect the profitability of integration projects?

What changes will Web Services evoke along the added-value chain?

What possibilities are there to promote Web Services?

What aspects ought to be taken into account by an enterprise before or during the provision or use of Web Services?

Problems (supplier/developer side and user side)

How do you assess the current situation on the side of the suppliers/developers of Web Services? (What are currently the difficulties with regard to developing and establishing Web Services?)

How should these difficulties be resolved in order to come to broadly accepted basic conditions concerning Web Services?

Should there be superior authorities to direct regulation and standardisation activities? (What organisations should be substantially involved in the decision process?)

(Critical aspects with regard to) Design of Web Services: quality, security, business models

What would be a good Web Service? (What quality criteria are particularly important?)

How could Web Services be compared with regard to quality aspects? (How could quality assessment be standardised?)

Who could administrate the quality management of Web Services?

What security criteria are particularly relevant concerning Web Services?

What standardisation activities in the field of XML-/Web security do you know?

Considering the state of the art – what must be improved with regard to Web Services security and XML security, respectively?

How important are security aspects of Web applications in general in relation to other criteria (e.g. performance, costs, usability ...)?

What business models could be pursued for operating Web Services?

What problems may occur with business models based on Web services with regard to quality and security?

Impacts on economy

Will Web Services create new jobs?

What impact will Web Services have concerning the division of labour? Will there be new forms of division of labour through the existence of Web Services?

Final statement
What are the most urgent challenges to face and steps to take in order to promote Web Services in general?