

CEN

CWA 14921

WORKSHOP

February 2004

AGREEMENT

ICS 35.080; 35.240.99

English version

Web Services: Technology and Standardization Aspects

This CEN Workshop Agreement has been drafted and approved by a Workshop of representatives of interested parties, the constitution of which is indicated in the foreword of this Workshop Agreement.

The formal process followed by the Workshop in the development of this Workshop Agreement has been endorsed by the National Members of CEN but neither the National Members of CEN nor the CEN Management Centre can be held accountable for the technical content of this CEN Workshop Agreement or possible conflicts with standards or legislation.

This CEN Workshop Agreement can in no way be held as being an official standard developed by CEN and its Members.

This CEN Workshop Agreement is publicly available as a reference document from the CEN Members National Standard Bodies.

CEN members are the national standards bodies of Austria, Belgium, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.



EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

Management Centre: rue de Stassart, 36 B-1050 Brussels

© 2004 CEN All rights of exploitation in any form and by any means reserved worldwide for CEN national Members.

Ref. No. CWA 14921:2004 E

Contents

Contents.....	2
Foreword.....	4
Introduction	5
1 Scope.....	6
2 Abbreviations	7
3 Requirements Overview for Web Services	8
3.1 Business requirements.....	8
3.1.1 Business process automation	8
3.1.2 Business-to-business integration	9
3.1.3 Enterprise application integration	9
3.1.4 Software engineering and application development.....	9
3.2 IT requirements	10
3.2.1 Directory and service description requirements	10
3.2.2 Communication and protocol requirements	10
3.2.3 Security requirements	10
3.2.4 Process modelling requirements.....	10
4 Technological aspects.....	11
4.1 The Web Services stack	11
4.1.1 Universal Directory, Discovery and Integration (UDDI)	11
4.1.2 Web Services Definition Language (WSDL)	12
4.1.3 SOAP.....	12
4.2 Business process related activities.....	13
4.2.1 Introduction.....	13
4.2.2 Business Process Markup Language (BPML)	14
4.2.3 Business Process Execution Language for Web Services	15
4.2.4 Business process related activities in ebXML	17
4.3 Semantics for Web Services	17
4.3.1 Introduction.....	17
4.3.2 The Semantic Web.....	18
4.3.3 WSDL and the Semantic Web	18
4.3.4 UDDI and ontologies	19
4.3.5 Implementation of the Semantic Web	19
4.4 Security related activities	19
4.4.1 General XML security.....	19
4.4.1.1 XML Encryption (XML-Enc).....	20
4.4.1.2 XML Digital Signature (XML-DSig).....	20
4.4.1.3 Extensible Access Control Markup Language (XACML).....	21
4.4.1.4 Security Assertion Markup Language (SAML)	21
4.4.1.5 XML Key Management Specification (XKMS).....	21
4.4.1.6 Platform for Privacy Preferences (P3P)	22
4.4.2 WS-Security (WSS).....	22
4.5 Relevant standardisation bodies in the field of Web Services.....	22
5 Analysis criteria for Web services	24
5.1 Security of Web services.....	24
5.1.1 Impact.....	24
5.1.2 Dimensions of security	24
5.1.3 Security risks	25
5.1.4 The future of Web services security.....	26
5.1.5 Usage scenario – Complex online purchase with credit check and PKI infrastructure	27
5.1.6 Conclusion.....	28
5.2 Quality of Web services.....	29
5.2.1 Availability.....	30
5.2.2 Accessibility	30

- 5.2.3 Integrity30
- 5.2.4 Performance30
- 5.2.5 Reliability30
- 5.2.6 Regulatory30
- 5.2.7 Security31
- 5.3 Business process support31
 - 5.3.1 Target Sector31
 - 5.3.2 Consortium Composition31
 - 5.3.3 Functional completeness31
 - 5.3.4 Registry and repository32
 - 5.3.5 Relation to document specifications32
 - 5.3.6 Modelling support32
- 6 Building and Using Web Services33
 - 6.1 Overview33
 - 6.2 Software Architectures33
 - 6.3 Development Platforms34
 - 6.4 Applications34
- 7 Current shortcomings of Web services35
- 8 Conclusion37
- Annex38

Foreword

This document represents the results of the Web Services Project Group of the Electronic Commerce Workshop in CEN/ISSS.

The Web Services project run from May 2002 until November 2003 and was coordinated by Dr. Boris Otto, Fraunhofer IAO, Stuttgart (Germany).

The work was initially planned to result in two CEN Workshop Agreements of which the first one dealt with technology and standardisation aspects and the second one with utilisation guidelines.

However, due to the closure of the CEN/ISSS E-Commerce Workshop that was announced in the 19th Workshop plenary meeting (Brussels, 15 October 2003) the project group reached consensus to finalise the current work status by combining the two documents into one CWA. Based on that decision this document represents the CWA as the result of the project group intending to outline technology and standardisation aspects, identifying current gaps and future needs for standardisation and giving an application scenario for the use of Web Services.

The final version of the CWA was approved in November 2003.

Introduction

Web Services technologies are considered to be a promising effort to facilitate efficient electronic business solutions. Using standard Internet protocols and specifications they aim at providing easy-to-use applications that enable companies to integrate and automate the business processes within their company, but especially with their business partners. Web Services are therefore intended to improve the interoperability of business processes between different organisational units and the underlying application systems. Web Services are also identified as one of the key building blocks of a trusted framework electronic business (CEN/ISSS 2003).

With regard to this, the present document represents the result of the Web Services Project Group of the CEN/ISSS Electronic Commerce Workshop. The project group aims at providing the user community with a sound analysis of the state of the art in the field of Web services and also with application scenarios that show how to efficiently use Web services technologies within industrial environments.

1 Scope

The present document gives guidance on technological and standardization aspects in the area of Web services. Within the overall scope, three main parts can be identified:

- An analysis of existing standardization work and technological foundations for Web services.
- A brief overview of areas of applications for Web services.
- Derived from the first two parts, the identification of current short comings of Web services.

The analytical part comprises the following aspects:

- Standardization initiatives that specify basic technologies, also business process related activities and those that deal with the security of Web services.
- Analysis criteria for Web services, especially from the security and the quality point of view.
- Development platforms and frameworks that enable and facilitate the use of Web services in professional environments.
- Software tools that are available on the market that support Web services technologies.
- The relationship between Web services technologies and the Semantic Web.
- The selected fields of applications are supposed to identify the main scenarios under which Web services are currently used. Considering the requirements of the latter and the capabilities of the current technological state of the art, shortcomings are outlined in the subsequent part.

The present document is applicable to all users of Web services technologies. It should provide them with a guidance to gain a clear understanding of the technological and business opportunities of Web services. The application of the document is not restricted to any type of organization.

2 Abbreviations

ACID	<i>Atomicity, Consistency, Isolation, Durability</i>
ANSI	<i>American National Standards Institute</i>
API	<i>Application Programming Interface</i>
BPEL4WS	<i>Business Process Execution Language for Web Services</i>
BPMI	<i>Business Process Management Initiative</i>
BPML	<i>Business Process Modeling Language</i>
CEN	<i>Committee Européen de Normalisation</i>
CORBA	<i>Common Object Request Broker Architecture</i>
CWA	<i>CEN Workshop Agreement</i>
EAI	<i>Enterprise Application Integration</i>
EDI	<i>Electronic Data Interchange</i>
EDIFACT	<i>Electronic Data Interchange for the Administration, Commerce and Transport</i>
EJB	<i>Enterprise Java Beans</i>
ERP	<i>Enterprise Resource Planning</i>
ETSI	<i>European Telecommunications Standards Institute</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hypertext Transfer Protocol</i>
IDL	<i>Interface Definition Language</i>
IETF	<i>Internet Engineering Task Force</i>
ISO	<i>International Organization for Standardization</i>
MDA	<i>Model Driven Architecture</i>
NAICS	<i>North American Industry Classification System</i>
OASIS	<i>Organization for the Advancement of Structured Information Standards</i>
OEM	<i>Original Equipment Manufacturer</i>
OMG	<i>Object Management Group</i>
P3P	<i>Platform for Privacy Preferences</i>
RDF	<i>Resource Description Framework</i>
RPC	<i>Remote Procedure Call</i>
SME	<i>Small and Medium Sized Enterprise</i>
TCP/IP	<i>Transfer Control Protocol/Internet Protocol</i>
SAML	<i>Security Assertion Markup Language</i>
SMTP	<i>Simple Mail Transfer Protocol</i>
UDDI	<i>Universal Description, Discovery and Integration</i>
UNSPSC	<i>United Nations Standards Products and Services Code</i>
URI	<i>Uniform Resource Identifier</i>
UUID	<i>Universally Unique Identifier</i>
W3C	<i>World Wide Web Consortium</i>
WSDL	<i>Web Service Description Language / Web Service Definition Language</i>
XACML	<i>XML Access Control Markup Language</i>
XML	<i>Extensible Markup Language</i>
XTM	<i>XML Topic Maps</i>

3 Requirements Overview for Web Services

Before starting to analyse Web Services technologies it is essential to identify relevant requirements that Web Services are supposed to fulfil. Web Services technologies do not just exist per se, but have been developed to solve distinct business or technical problems that currently available technologies do not solve to a sufficient extent.

Considering this, the following chapter aims at identifying those business and technical needs that turned out to be essential for the impetus of the development of Web Services technologies. The chapter is divided into two parts of which the first one deals with the business requirements as the driving and economically justifying factor for the development of new technologies. The second parts covers mainly technical problems that Web Services address without emphasising an explicit business need. Thereby, the requirements identification takes into account that the development of Web Services is mainly a technology driven approach.

Comprising, the identification of the requirements for Web Services aims at answering the following questions:

- What problems do Web Services promise to solve?
- What is the differentiating factor of Web Services compared to similar technologies?
- What are the main fields of application Web Services should be utilised for?
- Why is standardisation of Web Services a critical success factor?

3.1 Business requirements

Web Services technologies are often considered a technical approach. The main development initiatives are dominated by software vendors and technical expertise. However, the idea of Web Services will only be able to succeed if it becomes clear to users which problems the technology aims to solve and which business needs they help to fulfil.

Web Services technologies are based on the concept of offering certain business functionalities over standard Internet protocols to a variety of users. Services can be invoked independently of the underlying network architecture, the application interfaces and the communication protocols. Also, Web Services can be equally used in internal and external environments.

Considering especially the service idea in providing business functionalities to 'services consumers', a couple of application fields seem to be particularly suitable for the use of Web Services. These are outlined in greater detail in the following.

3.1.1 Business process automation

In many supply chains, companies tend to reduce the vertical range of manufacture so that an increasing part of value-added activities are carried out outside the company. This leads to an improved and closer synchronization of the business processes with the involved partners. Moreover, supply chains tend to become more complex and more dynamic due to fast-changing requirements of end customers and due to global markets.

Both the trend for outsourcing and for more dynamic co-operations between companies require efficient execution of the business processes. Since they are a means to increase the competitiveness of supply chains, business processes must be automated to reduce the cost that occur for their execution, to increase the quality of the execution and to reduce the cycle times.

Business process automation across company boundaries means to synchronise processes that are supported by different application systems with different configurations and different interface formats. Web Services technologies are a promising approach to solve the problem of automated business processes. Thanks to the use of standardized communication protocols and standardized interface descriptions Web Services technologies allow business processes to be inter-connected without manual activities. Applications expose certain functionality to the web in a format that can be understood by the application that supports the 'outside' part of the same business process.

In a final consequence of this idea, 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 entire parts of the organisation to specialised providers, reducing the functional scope of the organisation and enabling it to concentrate on its core competencies.

3.1.2 Business-to-business integration

Closely related to business process automation is the idea of business process integration as the technical foundation for automated business processes. Especially in the field of business-to-business integration, i.e. between companies within a supply chain, efficient integration becomes more and more critical because

- One-dimensional supply chains become complex networks,
- Business relationships are not static any longer, but instead become increasingly dynamic,
- The vertical range of manufacture in many industries decreases continuously (as outlined above),
- Time constraints become more and more dominant.

This leads to the need of a flexible, fast and easy way to integrate – and also to decouple – business processes across company boundaries. Cost and time savings must therefore not only be realized during transaction time (see process automation), but also during the time of integration itself.

Considering this, Web Services are a promising approach for minimising integration costs and efforts. They offer standardized interfaces, protocols and registry technology that allows the automation of business process integration.

3.1.3 Enterprise application integration

Besides the integration of cross-company business processes, Web Services technologies can also be employed in the area of enterprise application integration (EAI). A lot of similarities of EAI and business-to-business integration can be identified: Both deal with the coupling of business processes and business functionality that runs on different platforms and/or on different applications. Also, applications are connected using several types of middleware technology.

However, there are some important differences between the concept of EAI and business-to-business integration:

- EAI usually is behind firewall so that a different level of security applies in contrast to business-to-business integration.
- All process supported by internal applications are under the control of the company whereas the control for a business-to-business process is not within the company.
- Internal business processes are deterministic in the sense that the process design is not influenced by a third party.
- The system landscape and the system infrastructure is also under control of the company. Platforms, programming languages, interface and middleware technology can be harmonized within a company.

Web Services are often employed internally to achieve that very objective, namely to harmonise the internal system infrastructure. Their employment follows an evolutionary approach where harmonization is first intended internally, thus creating the prerequisite for a harmonization of processes and systems between business partners in a second step.

3.1.4 Software engineering and application development

Web Services technologies also provide a promising approach to improve the development of information systems and applications. The concept of Web Services can be considered the consequent advancement of the idea of component based software engineering. Taking this into account, Web Services offer the following benefits in the area of software engineering:

- Fast application development and application provisioning
- Reduced costs through component re-use
- Improved quality through proven components
- Reduced complexity through standardized interfaces
- Increased flexibility

3.2 IT requirements

The IT requirements for the provision of effective Web Services to the user community pose an interesting challenge to the developers of Networking services and software components alike. Important factors for the effective implementation of such services are on the network side 'real-time' communications. The effective band-width not only between the services but also from user to service should be adequate enough for the service user to believe that he is working on his system. In fact the key to all the requirements are the effective invisibility of the network and systems working in the background.

3.2.1 Directory and service description requirements

Requirements in this sector can be divided into 2 relatively autonomous sections. One requirement is the ability to find data and services irrespective of their location or as to how they are being searched for. The key here is the 'Semantic Web' as envisaged by Tim Berner-Lee in 2010 in which the keywords irrespective of context or language lead one to the respective service or data store location. It may well be that the requirements can't be satisfied by one single location or service provider, this should however be irrelevant to the user whether he be a person or a system.

The second requirement is the ability of service providers to change location without effecting a previously built chain of processing or data searches. Data and Service providers require the ability to change the systems upon which their systems rely without affecting the complete process chain from the Users or calling processes view.

3.2.2 Communication and protocol requirements

For Web Services to function over the existing internet connections the question of process control, process responsibility and timing needs very clear definitions and implementations. Although in critical situations, which are not infrequent data packets are lost or misrouted, a user controlling the process always has the possibility of breaking the waiting loop and re-initiating the request. Should the process or information request be initiated by an automatic process, much more clearly defined rules and controls are required in order to ensure that no automatic processes are left indefinitely in limbo.

3.2.3 Security requirements

With systems being permanently on-line firewalls are now indispensable, as was earlier the case for virus scanners. More importantly both have to be kept up to date with the latest know-how. The firewalls have also to be professionally configured based on user and system requirements. The less the firewall lets through the safer the local environment, but also the more restricted one is in what can be performed locally.

The second security area is that of securing the transmissions with signatures and various combinations of public and private key depending upon both the level of security required as well as the sensitivity of the information being transferred.

The third area is where separate networks are installed and used for a community of users in order to avoid data and traffic being compromised. This is not only an area for military transmissions, also the automobile industry has chosen this direction.

3.2.4 Process modelling requirements

The future of IT processing via the internet is that Processes no longer need to be closely coupled or completely known and understood. The use of internet services means that processes physically residing in geographically distant or business competitors premises are being used to provide a more efficient and cost effective IT environment. Also the tendency is to migrate from the simple transfer of information towards a cooperation at the business function level which implies that not only is it important to be able to transfer data between cooperating companies it's also imperative that one is aware of the business process being used to process that data and provide the initiator with a response without any manual process having to be initiated in between. On the other hand the real processing of such requests for competitive reasons has to be hidden from the partner.

4 Technological aspects

4.1 The Web Services stack

4.1.1 Universal Directory, 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 internet 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. However, it should not remain unstated that UDDI is not limited to internet usage but can also be applied and customized for an intranet environment, too. For example, a company could build up and run its own customized UDDI registry.

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 , NAICS and ISO 3166¹) are used, which offer requestors 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 storing 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 compatible 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.²

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.

¹ A classification for geographical regions and countries

² A full presentation of all API functions would exceed this paper. For further information please see (BELLWOOD ET AL. 2002).

4.1.2 Web Services Definition 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 formats 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 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 Figure 1.

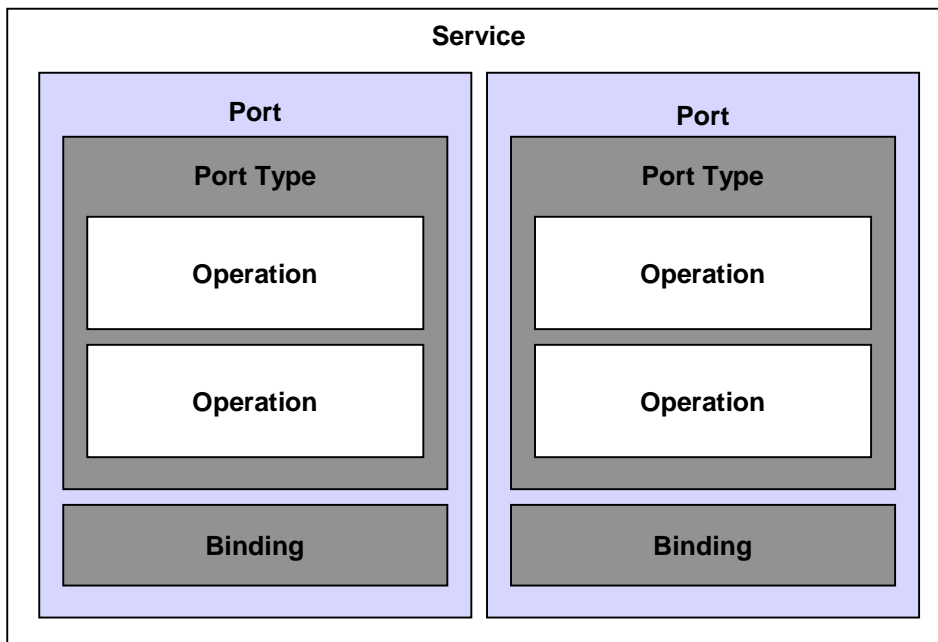


Figure 1: Structure of WSDL

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.

4.1.3 SOAP

SOAP is probably the most important component of Web Services technology. SOAP represents an abstract layer for transferring actual data. In particular, it specifies a neutral representation of exchanged data by hiding specific adoptions to communication protocols like HTTP or SMTP.

Like WSDL, SOAP is based on XML and grown up from 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 organizations have taken part in the collaboration meanwhile. The latest version, SOAP v1.2, has now received a high status ('candidate recommendation') by the W3C, implying that this specification is before its final approval for recommendation and later on to be used as a standard.

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.

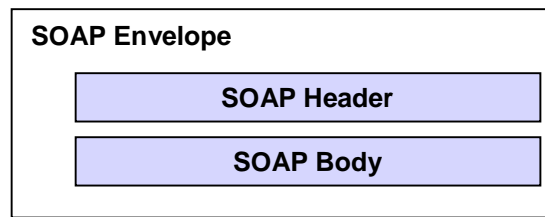


Figure 2: Structure of a SOAP document

Today, SOAP tools exist for nearly all important application platforms, e.g. for Java 2 Enterprise Edition, Microsoft .NET, and for all major programming languages. They provide support in establishing the required serialization and deserialisation of the SOAP messages, as well as in using SOAP with different bindings to communication protocols like HTTP and SMTP. These tools are supporting the application programmer in setting up the required communication infrastructure. In case of an HTTP binding of SOAP, this includes also extensions to web and application servers such as Apache, IIS, or Web Sphere.

Thus, SOAP provides a technology for exchanging any kind of data independent of a particular communication protocol. Crucial advantages of SOAP are its simplicity and extensibility. Another characteristic of SOAP is that it can be applied to synchronous protocols like HTTP as well as to asynchronous ones like SMTP. Of course, this advantage may also be turned into a disadvantage, because application independent communication protocols are used and the corresponding security systems (like Firewalls in case of a HTTP communication protocol) are not able to differentiate SOAP remote procedure calls from simple web page access. Thus, system administrators have the duty to very carefully inspect the services offered to avoid security gaps. Finally, it must be said that SOAP does not support security mechanisms like encryption, authentication, or digital signatures, which might be of interest in business scenarios, although, there exist such specifications for XML documents. Further, it does not provide process support and transaction management, which are provided by higher levels of the web service stack.

4.2 Business process related activities

4.2.1 Introduction

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 ORB, 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 inter-dependent. 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 3,

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.

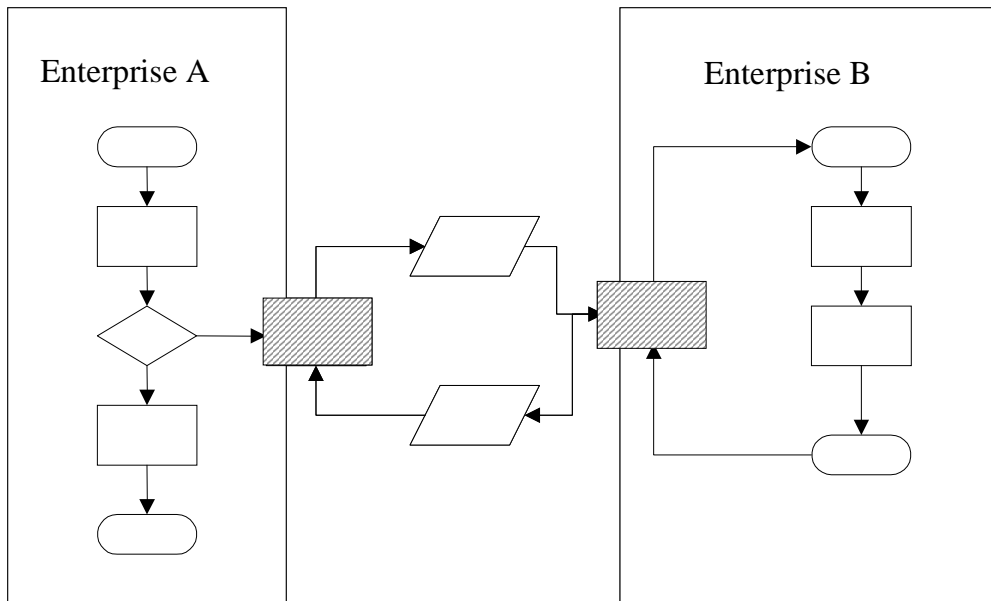


Figure 3: Cross-company business processes

Since development activities in the field of process-related specification are in a state of flux, new specifications are issued while others are not developed further. Early specifications were XLANG by Microsoft (THATTE 2001), WSFL by IBM (LEYMAN 2001) and BPML by BPMI. XLANG and WSFL will not be further developed, because a new specification, the BPEL4WS, was introduced in August 2002 by Microsoft, IBM and BEA, representing a convergence of XLANG and WSFL. Since business processes and workflow management are two strong relating areas, the work of the Workflow Management Coalition³ is also interesting from a web service point of view, especially the XML Process Definition Language (XPDL).

After the publication of the Web Service Conversation Language (WSCL) (BANERJI ET AL 2002) and the Web Service Choreography Interface (WSCI) (ARKIN ET AL. 2002) the W3C has started in January 2003 the Web Services Choreography Working Group⁴ to develop a vendor independent specification for “describing a choreography, as well as the rules for composition of, and interaction among, such choreographed Web services. The language(s) should build upon the foundation of the Web Service Description Language 1.2 (WSDL 1.2).” (CHINNICI ET AL. 2003).

In the following sections two exemplary specifications will be described a little more in detail: the BPML and the BPEL4WS.

4.2.2 Business Process Markup Language (BPML)

The 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. Companies that have substantially contributed to BPMI are CSC, Intalio, SAP, SeeBeyond, Sun and Versata.

BPMI defines the focus of BPML as follows (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

³ See <http://www.wfmc.org>.

⁴ See <http://www.w3.org/2002/ws/chor/>.

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 (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 4 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.

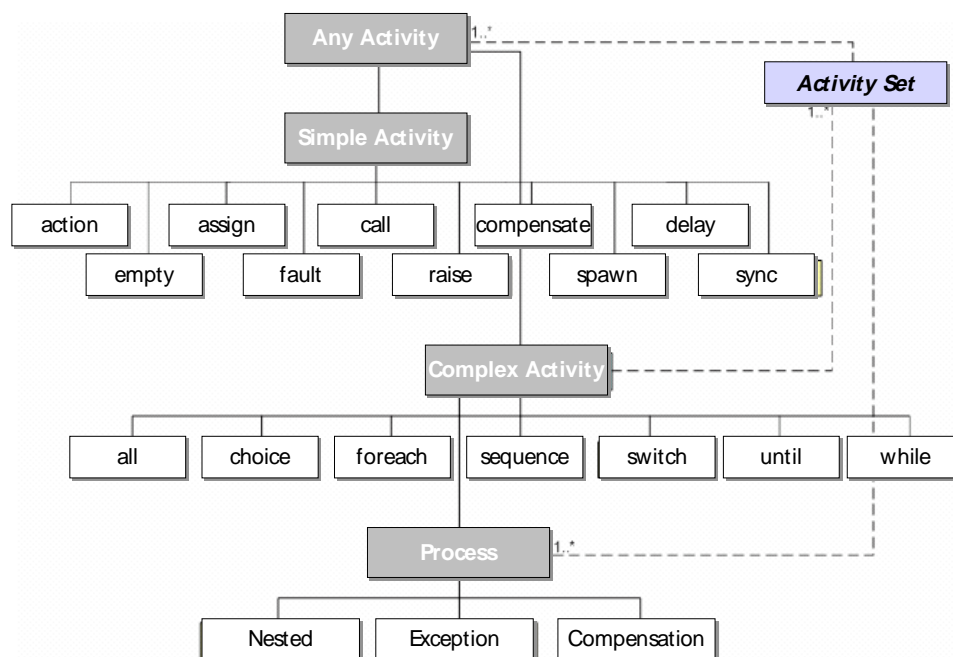


Figure 4: Different activities of the BPML specification

On the publication of BPEL4WS (see below) the BPMI consortium recognises the initiative as a major contribution to the area of business process management in general and envisages strongly a convergence of different competing standardisation efforts in the thematic area. It also reckons BPML a superset of BPEL4WS (BPMI 2002).

4.2.3 Business Process Execution Language for Web Services

Business Process Execution Language for Web Services (BPEL4WS) is the most up-to-date specification. Its latest version 1.1 was published in May 2003 by a group of large vendors (ANDREWS ET AL. 2003). As already mentioned, BPEL4WS is the official successor of XLANG and WSFL.

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 are compliant with 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.

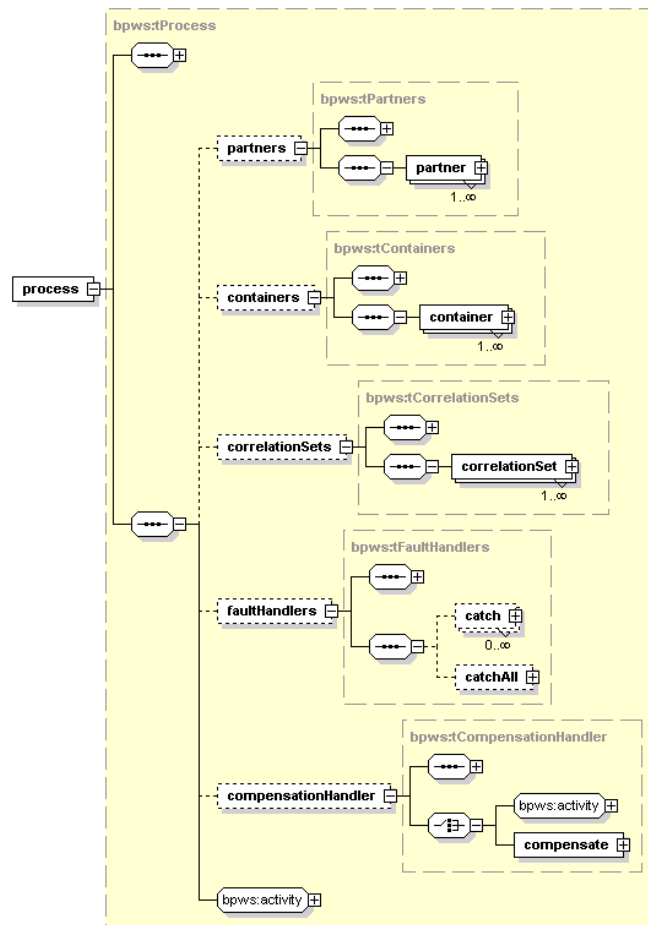


Figure 5: Schema of a BPEL4WS process

The schema of a process definition in BPEL4WS is shown in Figure 5. 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 modelling 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. 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.

4.2.4 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 (OPENXCHANGE 2002).

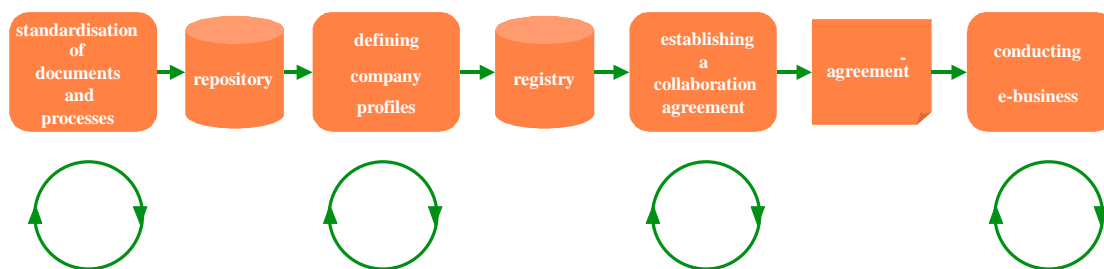


Figure 6 – ebXML Infrastructure Components

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.

4.3 Semantics for Web Services

4.3.1 Introduction

With UDDI and WSDL powerful mechanism for the description of the metadata of Web Services are available. As explained in this document this allows a formal specification of the Web Service providing information about vendor identifier, prose description, Internet address for messages and format of request and response messages etc. To use a Web Services it is necessary to understand the interface description. Both communications partners have to agree on the formal specification (this is done by WSDL) and have to have a common understanding of the elements defined by WSDL. The communication partner has to appreciate the semantic of a message like 'StockrateRequest'. A prose description provided with the message definition helps the programmer of the client to understand the semantic of the interface description. This prose description is not understandable by machines and therefore a formal verification of the interface interoperability and an automatic interface mapping is not possible.

The same dilemma other communication standards like CORBA, COM or EJB have. With IDL a powerful interface definition language is available to describe the interface of Objects communication via a CORBA Object Request Broker. The semantic of the interface is not defined. The OMG was elaborating the Model Driven Architecture (MDA) that expresses also the business logic that helps to understand the semantic of the interfaces.

A technology getting popular in parallel with Web Services is the Semantic Web. The Semantic Web is a vision of creating a navigation map of the Internet. Derived from the linguistic technology of Semantic Networks the Semantic Web describes the content of information resources in the Internet and helps to make

the links between information more intelligent. This technology can be used to extend the Web Service Specification by semantic descriptions.

4.3.2 The Semantic Web

The Semantic Web is used to define the semantic of information resources in the network and to describe their relations. The technology of the Semantic Web is Semantic Networks. Derived from ISO 13250 Topic Maps two major activities are emerged: XML Topic Maps (XTM) and the Resource Description Framework (RDF) (BRICKLEY & GUHA 2003).

RDF provides meta data to define meta information to describe the content and the links between information resources in the Internet. The content provider extends the information e.g. web pages by RDF. The RDF specification can be analysed and used to find and to navigate to the content and relates information resources.

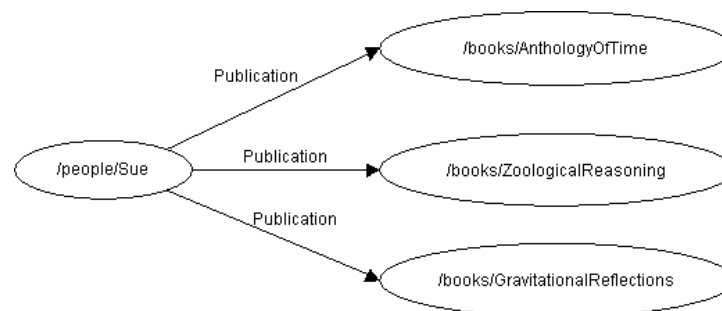


Figure 7: Simple RDF example

RDF defines 'Resources' by connecting them to web resources by providing an URI. "Properties" link Resources (s. fig.) or describe them by "Literals". Based on this very simple and generic mechanism RDF Schema is developed. RDF Schema is an open and extendable way to enhance RDF by typed elements to model more complex semantic information. With RDF and RDF Schema the content developer is able to express ontologies (and define their semantic by referencing standard resources e.g. by linking the term 'Tomato' to <http://www.usda.gov/doc/tomatogr.htm#gradeA>) and to use this ontology to describe Internet information resources and their links.

XTM uses a top down approach to set up ontology by defining 'Topics'. 'Associations' link these Topics. Topics and Associations are defined by unique references to web information resources. It is shown that information described with XTM can be translated in RDF and vice versa.

4.3.3 WSDL and the Semantic Web

As depicted WSDL is a meta language to describe transactional Internet resources: Services. The interface definition exactly defines the interactions a Web Service is offering by combining messages to operations, describes the data schema of the message body and binds it to a transport protocol. The semantic of these elements can only be expressed in prose. The client has to understand the semantic for the schema elements and their relations in the Type definition and the names used for Messages and Operations and if used the name of the Port Type. Port Type and Binding is sufficiently technically described by the specifications of SOAP etc.

By using RDF the Semantic Web can help to provide additional meta information to specify the semantic (e.g. the terminology used for the names) of these elements. One of the requirements of the Web Service Architecture Specification of the W3C is to be RDF compliant⁵. RDF can be used to supercharge WSDL with semantic information. The client uses this information to understand the Web Service in a better way or use (semi-)automatic tools to map interfaces (here is still a lot of work to do!).

⁵ See 'Web Services Architecture Requirements' working draft <http://www.w3.org/TR/2002/WD-wsa-reqs-20021114>: 'AR009.2 new Web services technologies, developed by W3C Web Services WGs, *should* be capable of being mapped to RDF/XML'.

The Semantic Web Technology can also be used to describe the quality of a Web Service e.g. performance, availability, costs. This has to be incorporated in to the standard by the driving organizations (s. chapter 4.5). The selection of the technology is still open.

4.3.4 UDDI and ontologies

As depicted in chapter 4.1.1 yellow pages are used to classify the UDDI provider. For this classification systems like UN/SPSC, NAICS and ISO 3166 are used. Because of the broad scope of these classification systems the differentiation between similar Web Services is pretty rough. Additionally UDDI classifies the Web Service as a whole. A classification of special Port Type or Operations is not possible.

Ontologies⁶ provide richer semantic than classifications. Using ontologies the specification of Web Services can be more detailed and domain specific. With the Semantic Web powerful technologies for the definition and referencing of ontologies are available. This allows defining a much more detailed and domain specific description of Web Services managed by UDDI. By enriching WSDL with Semantic Web technologies the semantic description of the Web Service elements is possible. That can help to search Web Services on operation level.

Using Topic Maps for the definition of navigational Maps of Internet resources, these Maps also can reference transactional resources by identifying Web Services. Therefore Topic Maps can provide an alternative mechanism to the UDDI concept.

4.3.5 Implementation of the Semantic Web

Until now it is discussed in this document how the Semantic Web technologies are used to enrich the Web Service specification and to identify and find Services. But Web Services can also provide tools to enable Semantic Maps. There are several approaches like document server (access by XLINK references), catalogue server for ontologies and many more.

4.4 Security related activities

4.4.1 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 specifications 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)

⁶ An ontology is an explicit specification of a conceptualization and a shared understanding of some domain of interest.

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.

4.4.1.1 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.

4.4.1.2 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.

4.4.1.3 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.

4.4.1.4 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

4.4.1.5 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. XKMS 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.

4.4.1.6 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.

4.4.2 WS-Security (WSS)

SOAP, the core Web Services specification, does not include any security mechanism. However, the listed XML security specifications can of course also be 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' (WSS). 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.

4.5 Relevant standardisation bodies in the field of Web Services

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 standards 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 authorities 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.

5 Analysis criteria for Web services

5.1 Security of Web services

5.1.1 Impact

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 per cent 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.

5.1.2 Dimensions of security

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 further 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.

5.1.3 Security risks

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.

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’ (WSS). 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.

5.1.4 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):

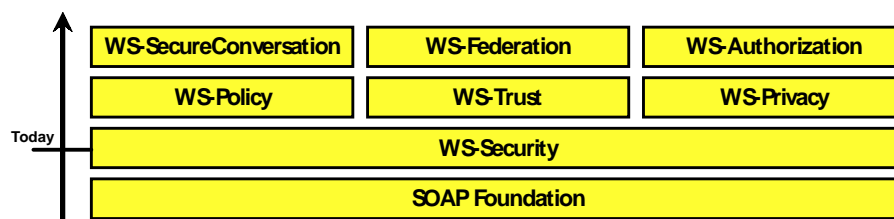


Figure 8: Future of WS-Security (Source: IBM and Microsoft)

- 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.

5.1.5 Usage scenario – Complex online purchase with credit check and PKI infrastructure

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 fourth 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:

- 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.
- All three register their public keys with 'Trust Company's' XKMS server via X-KRSS.
- 'Smith Company' retrieves 'Machine Company's' public key at 'Trust Company' via X-KISS.
- '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'.
- '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.
- 'Machine Company's' order processing system sends an access request to 'The Bank's' credit rating database and signs it with its digital signature.
- '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.
- 'The Bank' charges 'Machine Company' for accessing its database and sends a SOAP document containing a SAML assertion to 'Machine Company'.
- 'Machine Company' retrieves 'The Bank' public key at 'Trust Company' via X-KISS.
- '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.
- 'The Bank's' credit rating database verifies 'Machine Company's' digital signature and the SAML security assertion.
- 'The Bank's' credit rating database decrypts the SOAP payload using its own private key.
- 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'.
- 'Machine Company's' order processing system verifies the digital signature and decrypts the SOAP payload containing the requested information.
- 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.

This process is described in the figure below:

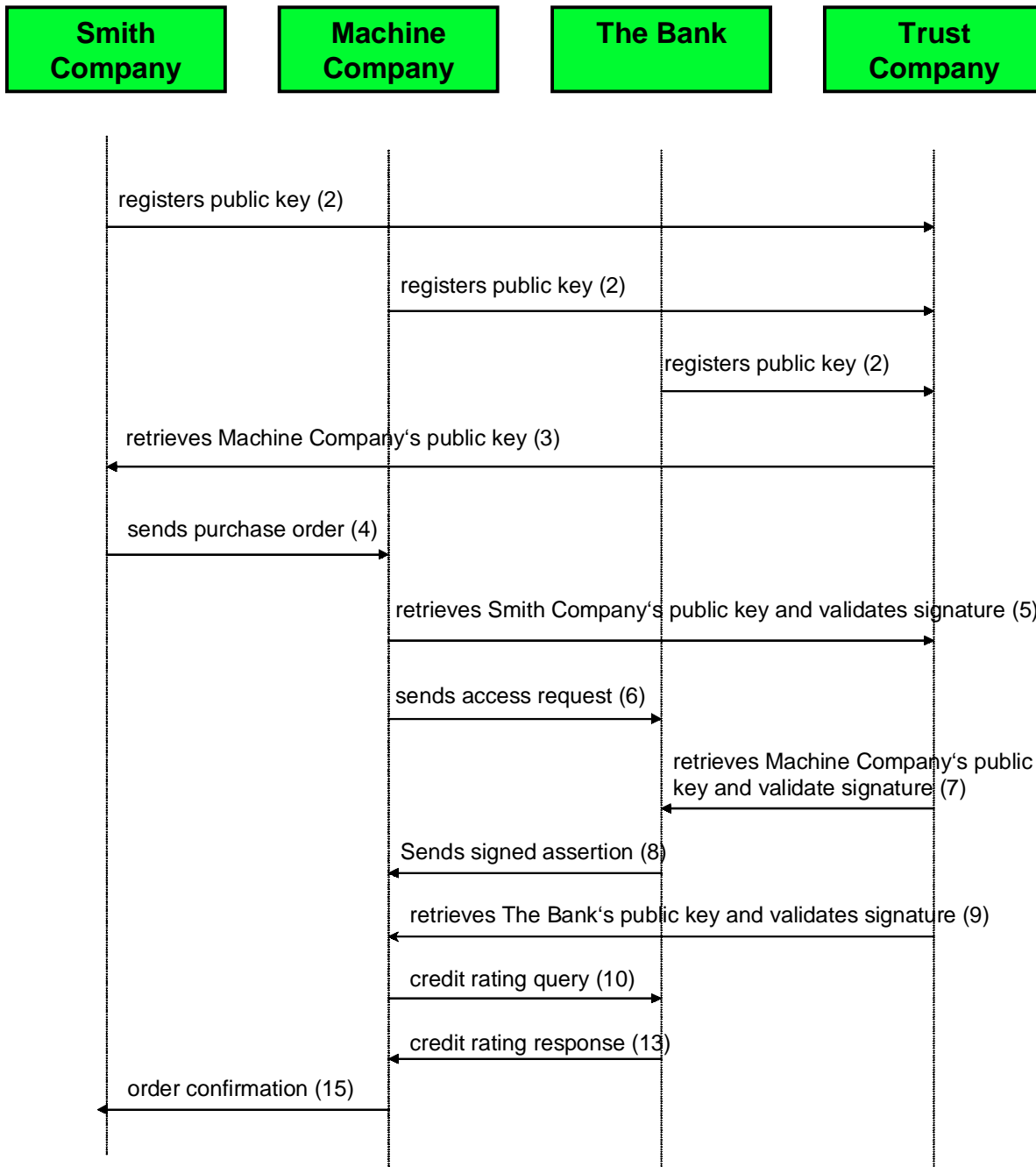


Figure 9: Process flow of the usage scenario

5.1.6 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.

5.2 Quality of Web services

With the widespread dissemination of Web services, basic requirements such as quality will become a crucial factor in distinguishing the success of service providers and therefore an 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.

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?
- 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").

It should not remain unstated that the following quality aspects and quality criteria drafted and pictured for the Web Services field, can be principally adopted for similar technologies and services with relative ease.

5.2.1 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.

5.2.2 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.

5.2.3 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.

5.2.4 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.

5.2.5 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.

5.2.6 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.

5.2.7 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.

5.3 Business process support

Analysing Web Services technologies from a process scope point of view means to identify to which extent the currently available specifications support business processes in user companies. Due to the fact that this is a multidimensional criterion, a set of analysis parameters can be identified in order to describe the current ability of Web Services technologies.

The following chapter introduces the main parameters to determine the extent of business process support of Web Services.

5.3.1 Target Sector

It can be distinguished between specifications that are developed for certain industries and those that can be used in a broader sense. The first have a vertical dimension, the latter a horizontal one (ESSWEIN & ZUMPE 2002). In the field of business process frameworks e.g. RosettaNet has a clear focus on the High Tech Industry whereas ebXML is rather independent of specific industry application.

5.3.2 Consortium Composition

The composition of the consortia that develop Web Services specifications is an important criterion for user companies. It affects the following aspects:

- Independence of certain vendors
- Openness for users' feedback
- Acceptance due to the fact that the specification is supported by products of the vendors that are involved in the specification
- Stability and sustainability of the approach
- Potential of standardization

Different types of consortia can be identified. Some consist of industry representatives, some are driven by software vendors, and others are developed under the umbrella of an independent standardization organization, e.g. CEN, OASIS or UN/CEFACT.

5.3.3 Functional completeness

This analysis parameter has two dimensions. On one hand, functional completeness refers to the ability of the specification to support certain functional business processes, e.g. from the field of purchasing or accounting. On the other hand it refers to the ability to support complex business process structures. The most important process structures are:

- Sequential processes
- Iterative processes
- Alternative processes
- Joins and forks

5.3.4 Registry and repository

Another evaluation parameter derives from the provisioning of mechanisms for registries to store company and service information as well as for repositories to store process schemas. This derives from the requirement not just to support the execution of established processes, but to also cover the needs of the business process design phase and of highly dynamic process environments.

5.3.5 Relation to document specifications

Business processes form the framework for the exchange of business information. Business information is mostly packaged in business documents such as purchase orders, delivery notifications, and invoices. To judge the ability of Web Services technologies to support business processes it is therefore important to analyse their compatibility with prominent business document specifications. Among those are XML based ones such as xCBL, openTRANS, cXML etc. but also EDI based initiatives such as EDIFACT and ANSI X.12.

5.3.6 Modelling support

Since Web Services technologies aim to facilitate the business-to-business integration and the process automation a close relation of Web Services to modelling methodologies is required. In an ideal situation, Web Services could be modelled via a graphical user interface and be automatically be instantiated. The modelling support of Web Services technologies relates therefore to the following aspects:

- Modelling methodologies such as UML
- Tools support for the modelling and implementation of Web Services

6 Building and Using Web Services

6.1 Overview

The following chapters give a short overview over the different aspects arising during the development of web service based applications. First we will discuss some architectural aspects for web service based applications followed by a short section about development platforms. The last section describes some possible fields of application of web services, which will be discussed more in detail in CWA 2 'Selected Areas of Application and Utilisation Guidelines'.

6.2 Software Architectures

From a technical, software-development oriented point of view web services are a (another) technology to realize distributed software systems. The main challenge in the area of distributed software systems is to enable different applications, build on different operation systems and different infrastructures, to cooperate both within an enterprise and between enterprises. Because Web Services are using only XML (e.g. in form of WSDL and SOAP) and common Internet protocols they can be used as middleware between software applications based on different technologies like J2EE or CORBA and so facilitate the interoperability between these systems. Figure 1 shows the basic architecture for systems using Web Services. The service provider wants to publish some functionality of his software systems (e.g. a J2EE based proprietary application and an ERP system) for other enterprises. Therefore he has to provide different Web Services connectors (whichever protocol should be used), which are receiving the SOAP Messages, mapping these SOAP calls to the relevant services, calling the service, i.e. background applications and sending the result (if any) to the caller. The caller (Enterprise A) does not have to know, which technologies the service provider is using and vice versa.

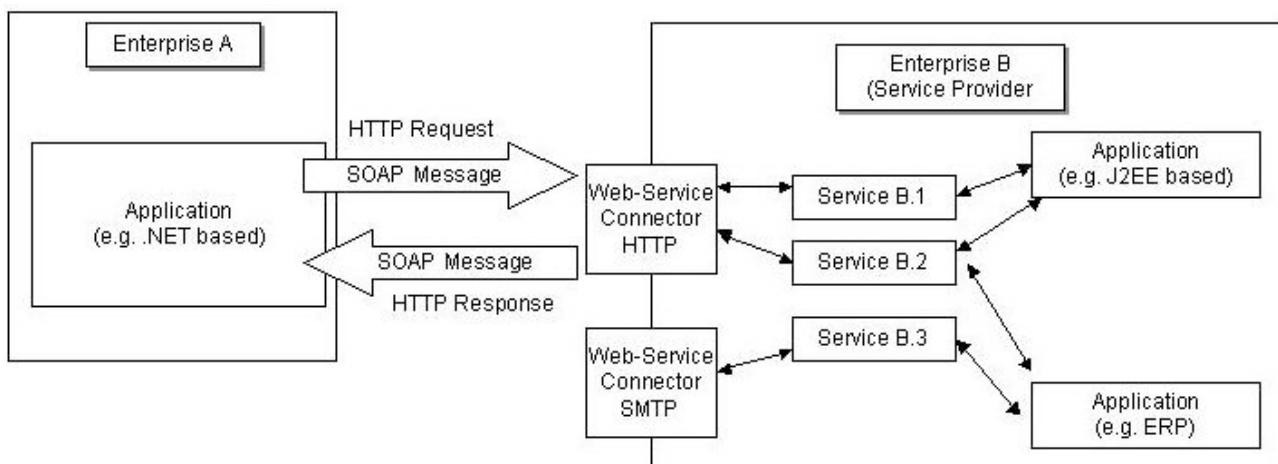


Figure 10: Basic Architecture for systems using Web Services

In the case of a http-based Web-Service connector the actual instantiation of the general architecture shown in Figure 10 the Web-Service connector is piece of software, which is listening on a TCP/IP-Port for http requests containing SOAP messages and processing these SOAP messages. In practice these Web-Service connectors are normal web servers (like Microsoft IIS or Apache) extended by additional functionality.

As an example let us have a look onto the J2EE specification. The J2EE specification defines a mechanism to extend a web server, so that web based applications (e.g. a HTTP based Web Services connector) can be implemented. This is the Java Servlet Specification. Servlets are Java classes, which are integrated in a Java Servlet Engine. Multiple servlets can be packaged into so called web applications (including all other needed resources like HTML pages). Via the servlet engine a servlet can access the information of the incoming HTTP request and can generate resp. manipulate the content of the http response. This operation mode is shown in Figure 11. This mechanism can now be used to implement a set of servlets, which are analyzing the via http incoming SOAP messages and calling the according application logic implemented as further Java classes.

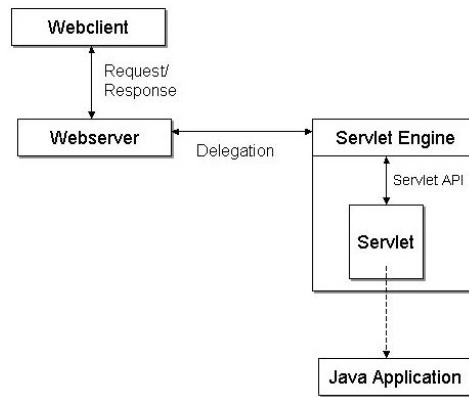


Figure 11: The Operation Mode of a Java based Web-Service Connector

6.3 Development Platforms

Principally every programming language and every development platform, which can handle XML data and use Internet protocols like http, can be used to implement Web Services. Surely there are platforms that are more suitable than others. In practice there are currently two main development platforms: the Microsoft .NET environment and the J2EE platform. In the context of Web Services (and distributed systems in general) a development platform is not only an IDE for editing and compiling the code but also an application server, where the code is deployed. Please notice, that the term J2EE platform is used here as a placeholder for any J2EE conformant development platform (e.g. BEA WebLogic, Sun ONE, JBoss, etc.)

For both platforms (.NET and J2EE) there are a number of additional tools (built-in and third party), which ease the implementation and the consumption of web services. Typical tasks of this additional tools are:

- Handling of XML data;
- Support for the generating and analysing of SOAP messages;
- The generation of WSDL documents from the service implementation;
- The generation of programming code for the use of a Web Service with a given WSDL description;
- The deployment of a Web Service;
- The generation of code, to test the service.

With the support of these tools the development of web service based applications is not different to the development of other distributed software systems.

6.4 Applications

The annex to this document contains an example for an application based on web services. In this section we will just shortly introduce the application of web services in the area of software development itself.

With web services the concept of object-oriented software development respectively component based software development will be extended to the service based software development with an underlying service oriented architecture. In this model, a software system is developed of building blocks implemented as web services. Such building blocks are for example an user- and right-management service including an authentication service or a payment service. There will be a set of common services used by a broad variety of applications and there will be special services used only for special applications, for example format conversions in the area of CAD. Another technical application (in conjunction with the above mentioned service oriented architecture) of web services is the use as application programming interface as already done for UDDI registries.

7 Current shortcomings of Web services

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.

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 currently 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.

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.

A major issue for the dissemination and adoption of web services will be the success of current standardization activities. 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 standards 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 until now none of the specifications around Web Services (SOAP, WSDL, UDDI) are formal standards in terms of being published by a governmental standardisation authorities like ISO or DIN. However, some specifications, such as SOAP, have been recommended by non-governmental standardisation organisations like W3C or OASIS.

At the moment the current standardization 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.

Moreover 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 must be defined and regulated. 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.

A further requirement is 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. Institutions like the European Commission (e.g. through CEN/ISSS) could be first choice for developing such quality standards. 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.

Another critical issue about Web Services is that they are totally dependent on the availability of the network. Consumers of services which rely on Web Services must be able to execute their processes and business tasks while working outside their offices or in low bandwidth conditions. Therefore software designers need to make sure their applications that use Web services function correctly even if the service is temporarily unavailable. On the other hand because mobile devices are constantly losing network connections, providing good and reliable services will be extremely difficult.

8 Conclusion

The present document represents the work of the Web Service Project Group of the CEN/ISSS E-Commerce Workshop. The project group aimed at analysing of the state of the art in terms of technology and standardisation aspects in the area of Web Services and at identifying current gaps and future needs for standardisation.

The document starts with a brief summary of the rationale for the development of Web Services technologies. After that main fields of standardisation are examined and evaluation criteria are introduced. Based on that, major gaps and future needs for action are identified. Among the most relevant are:

- Security of interoperable applications based on Web Services is not fully reached yet. Technical specifications are currently under development and under testing, but no trusted framework exist to allow e.g. reliable billing of the use of a Web Service.
- A legal and regulatory framework has yet to be developed that provides comprehensive guidelines on how to run and operate registries for Web Services.
- The mapping of different document specifications remains unsolved. Web Services can form the foundation for advanced integration solutions in the future that combine the advantages of Web Services and Semantic technologies to overcome current hurdles.
- At present the standardisation landscape in the area of Web Services is heterogeneous and not sufficiently accepted by user communities at the same time. Convergence of different efforts is as needed as more intensive dissemination of existing work.

The document closes with an exemplary application scenario that can be found in the annex.

Some of the recommendations of the project group should be considered as a baseline of future work, e.g. in the eBusiness Interoperability Forum proposed under the umbrella of the CEN/ISSS.

Annex

Bibliography

ANDREWS ET AL. 2003

Andrews, T. et al.: Business Process Execution Language for Web Services Version 1.1. © 2003. <http://www-106.ibm.com/developerworks/library/ws-bpel/>, requested on 2003-11-05.

ARKIN ET AL. 2002

Arkin, A. et al.: Web Service Choreography Interface (WSCI) 1.0. © 2002. <http://www.w3.org/TR/wsci/>, requested on 2003-05-28.

BANERJI ET AL. 2002

Banerji, A. et al.: Web Services Conversation Language (WSCL) 1.0. © 2002. <http://www.w3.org/TR/wscl10/>, requested on 2003-05-28.

BELLWOOD ET AL. 2002

Bellwood, T. et al.: UDDI Version 3.0. UDDI Spec Technical Committee Specification. © 2002. <http://uddi.org/pubs/uddi-v3.00-published-20020719.htm>, requested on 2003-05-28.

BPMI 2000

BPMI.org: The Business Process Management Initiative. © 2000. <http://www.bpmi.org>, requested on 2003-05-28.

BPMI 2002

BPMI.org: BPML | BPEL4WS. A Convergence Path toward a Standard BPM Stack. © 2002. <http://www.bpmi.org/downloads/BPML-BPEL4WS.pdf>, requested on 2003-11-05.

BRICKLEY & GUHA 2003

Brickley, D.; Guha, R. V.: RDF Vocabulary Description Language 1.0: RDF Schema. © 2003, <http://www.w3.org/TR/rdf-schema/>, requested on 2003-02-14.

CEN/ISSS 2003

CEN/ISSS: Roadmap for addressing key eBusiness standards issues 2003-2005. CEN: Brussels, 2003.

CHINNICI ET AL. 2003

Chinnici, R. et al.: Web Services Description Language (WSDL) Version 1.2. © 2003. <http://www.w3.org/TR/2003/WD-wsdl12-20030303/>, requested on 2003-05-28.

ESSWEIN & ZUMPE 2002

Esswein, W.; Zumpe, S.: Realisierung des Datenaustauschs im elektronischen Handel. In: Informatik-Spektrum (2002) Vol. 25 No. 8, p. 251-261.

LEYMAN 2001

Leymann F.: Web Services Flow Language (WSFL 1.0). © 2001 <http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf>, requested on 2003-05-28.

OPENXCHANGE 2002

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

THATTE 2001

Thatte S.: XLANG Web-Services for Business Process Design. © 2001 http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm, requested on 2003-05-28.

THIAGARAJAN ET AL. 2002

Thiagarajan R. et al.: BPML: A Process Modeling Language for Dynamic Business Models, in : Forth IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, Newport Beach, 2002.

Exemplary application scenario

The application scenario describes the collaboration between two companies during the purchasing and delivery of complex products. The scenario is characterised by the following aspects:

Need for the integration of various communication channels.

Need for the visualisation of product information.

The reference process for the scenario is depicted in the following figure:

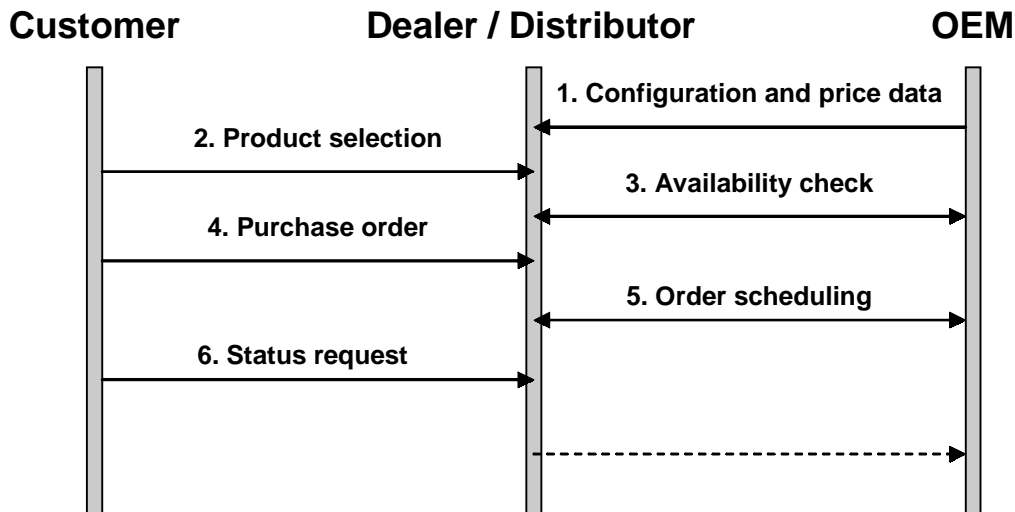


Figure 12: Reference for the application scenario

From the business requirements can be derived the following system requirements:

The overall system must provide an interface for user interaction as well as an interface for automated process integration.

- The system must combine synchronous and asynchronous communication between the dealer and the OEM.
- The system must be open for future enhancements of the overall scenario.

The system was realised by the integration of different classes of information systems. A portal application was chosen to support the human-machine communication between the customer and the dealer. For the integration of the portal and the ERP system of the OEM, a Web Services based integration approach was chosen in order to provide both systems with a maximum degree of openness for future adaptations.

The overall system architecture is depicted in Figure 13. The distributor provides a visual interface to the customer based on a portal application. Product information is offered to the customer in a central and personalised way. In doing so, the portal application combines three different systems, namely a content management system for product documentation (based on ZOPE), a shop system (based on Intershop infinity) that supports the order process, and a product configuration component that support the presentation of complex products. The integration of the portal application with the ERP system of the OEM was realised using Web Services technologies (WSDL and SOAP). The proprietary BAPI formats of the underlying SAP R/3 system were encapsulated as WSDL services.

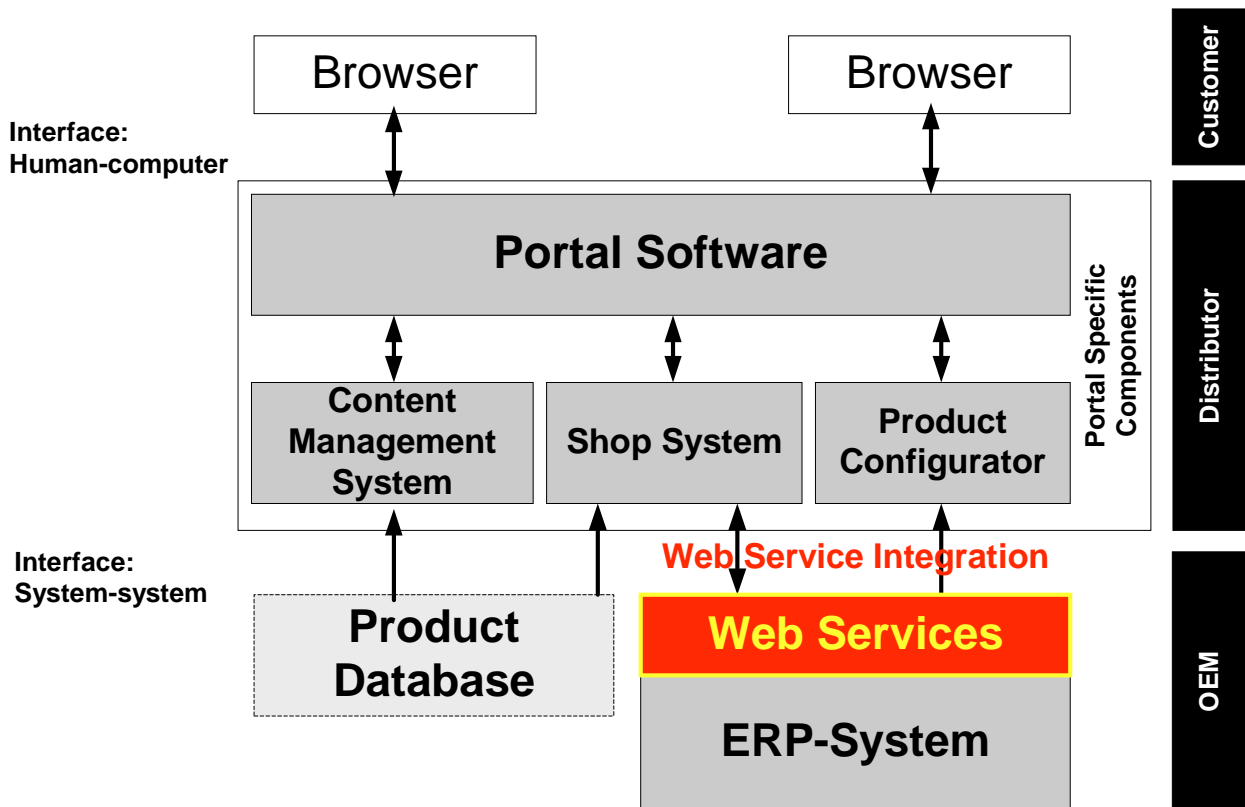


Figure 13: System architecture for the application scenario

Among the major experiences of the Web Services use in the present scenario were:

- Web Services provide a technology that facilitates efficient business process integration. Through the use of standardised interface descriptions the distributor in the application scenario is able to easily connect his system with the ERP system of a different OEM.
- Performance is good although the overall system is based on a very complex architecture.
- Problems in the use of Web Services occurred through the non-standardised use of SOAP in the products of the different vendors. The resulting problems could only be solved by agreements that went beyond the specification.