



SEVENTH FRAMEWORK PROGRAMME



GREEK INTEROPERABILITY CENTER

Deliverable D6.1 Interoperability Guide (1st version)

Deliverable Form	
Project Reference No.	204999
Deliverable No.	D6.1
Relevant workpackage:	WP6: Integrating and communicating knowledge
Nature:	R=Report
Dissemination Level:	PU = Public
Document version:	Version 1.00
Date:	29/7/2008
Editor(s):	NTUA

TABLE OF CONTENTS

TABLE OF CONTENTS.....	2
UPDATES TABLE.....	3
LIST OF TABLES	4
LIST OF FIGURES.....	5
EXECUTIVE SUMMARY.....	9
1. INTRODUCTION.....	10
1.1 PURPOSE	10
1.2 STRUCTURE OF THE DOCUMENT.....	10
2. THE INTEROPERABILITY DOMAIN	12
2.1 WHAT IS INTEROPERABILITY.....	12
2.2 HISTORICAL OVERVIEW AND POLITICAL CONTEXT OF INTEROPERABILITY	14
2.3 WHY INTEROPERABILITY MATTERS.....	16
2.4 FOCUS OF THE INTEROPERABILITY GUIDE.....	19
3. GUIDANCE TO ENTERPRISES.....	21
3.1 MIDDLEWARE DEFINITION	21
3.2 MICROSOFT BIZTALK SERVER	22
3.3 IBM WEBSHERE.....	24
3.4 ORACLE APPLICATION SERVER.....	26
3.5 SAP NETWEAVER	28
3.6 ESB ARCHITECTURE OVERVIEW.....	30
3.7 MULE	34
3.8 APACHE SERVICEMIX	36
3.9 FUSE ESB	37
4. GUIDANCE TO GOVERNMENTS.....	40
4.1 EUROPEAN INTEROPERABILITY FRAMEWORKS (EIFs)	40
4.2 EUROPEAN INTEROPERABILITY FRAMEWORK v1.0	40
4.3 EUROPEAN INTEROPERABILITY FRAMEWORK v2.0	43
5. INTEROPERABILITY STANDARDS	45
5.1 WEB SERVICE STANDARDS	45
6. RECOMMENDATIONS.....	55
6.1 FOR DECISION MAKERS	55
6.2 FOR POLICY MAKERS.....	56
REFERENCES.....	57
APPENDIX- INTEROPERABILITY KNOWLEDGE MAP	59
HORIZONTAL ISSUES.....	59
VERTICAL ISSUES	61
COUNTRY – RELATED ISSUES	62

UPDATES TABLE

Date	Version	Changes Made
29/07/2008	1.00	1 st Version Submitted to EC

LIST OF TABLES

Table 1: Horizontal Issues of the I-KMap	60
Table 2: Vertical Issues of the I-KMap	62
Table 3: Regional Dimensions of the I-KMap	64

LIST OF FIGURES

Figure 1: Focus of the Interoperability Guide	20
Figure 2: BizTalk Server architectural overview	23
Figure 3: SAP NetWeaver Integration Framework	29
Figure 4: Enterprise Service Bus Architecture (Connecting Different Applications and Technologies)	31
Figure 5: Apache ServiceMix integrating components and services	36
Figure 6: FUSE ESB Runtime Environment	39
Figure 7: G2C and G2G Interactions through a Governmental eService Portal	41
Figure 8: Interactions among Middleware Components	42
Figure 9: Web Services Standards Overview	46
Figure 10: Dependencies of Business Process Specification Standards to other Categories of Web Service Standards	47
Figure 11: Dependencies of Metadata Specification Standards to other Categories of Web Service Standards	48
Figure 12: Dependencies of Reliability Specification Standards to other Categories of Web Service Standards	49
Figure 13: Dependencies of Messaging Specification Standards to other Categories of Web Service Standards	50
Figure 14: Dependencies of Management Specification Standards to other Categories of Web Service Standards	51
Figure 15: Dependencies of Presentation Specification Standards to other Categories of Web Service Standards	51
Figure 16: Dependencies of Security Specification Standards to other Categories of Web Service Standards	52
Figure 17: Dependencies of Transaction Specification Standards to other Categories of Web Service Standards	53
Figure 18: Dependencies of Resource Specification Standards to other Categories of Web Service Standards	54

LIST OF TERMS AND ABBREVIATIONS

AIX	Advanced Interactive eXecutive
API	Application Program Interface
ASN	Advance Shipping Notice
B2B	Business To Business
BI	Business Intelligence
BPA	Business Process Automation
BPEL	Business Process Execution Language for Web Services
BPM	Business process Management
BPSS	Business Process Specification Schema
CIDX	Chemical Industry Data Exchange
COM	Component Object Model
COTS	Commercial-Of-The-Shelf
CPG	Consumer Packaged Goods
EAI	Enterprise Application Integration
eBIF	e-Business Interoperability Framework
ebXML	Electronic Business Xtensible Markup Language
EDI	Electronic Data Interchange
EIF	European Interoperability Framework
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
ESQL	Extended Structured Query Language
EU	European Union
FIPA	Foundation for Intelligent Physical Agents
FP7	Seventh Framework Programme
FTP	File Transfer Protocol
GIC	Greek Interoperability Center

GS1/UCC	Global Standards One/ Uniform Code Council
HTTP	HyperText Transfer Protocol
IBM	International Business Machines
ICT	Information and Communication Technology
IDABC	Interoperable Delivery of European eGovernment Services to public Administrations, Business and Citizens
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IIS	Internet Information Services
ISV	Independent Software Vendor
IT	Information Technology
J2EE	Java 2 Platform Enterprise Edition
JAX-RPC	Java API (Application Program Interface) for XML-based RPC (Remote Procedure Call)
JB	Java Business Integration
JBoss	Application server written in Java that can host business components developed in Java. Essentially, JBOSS is an open source implementation of J2EE that relies on the Enterprise JavaBeans specification for functionality.
JCA	J2EE Connector Architecture
JonAS	Java Open Application Server
JMS	Java Message Service
KW	Knowledge Warehouse
MDM	Master Data Management
MI	Mobile Infrastructure
MOM	Message-Oriented Middleware
MQ	Message Queuing
MSMQ	MicroSoft Message Queuing
NIF	National Interoperability Framework
OAG	Open Applications Group
OASIS	Organization for the Advancement of Structured Information Standards
ODETTE	Organisation for Data Exchange by Tele Transmission in Europe

OEM	Original Equipment Manufacturer
OEMS	Oracle Enterprise Messaging Service
PEGS	PanEuropean eGovernment Services
PO	Purchase Order
POJO	Plain Old Java Object
POSIX	Portable Operating System Interface
QoS	Quality of Service
SAAJ	SOAP with Attachments API for Java
SAP	Systems, Applications and Products in Data Processing
SME	Small and Medium Enterprises
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Socket Layer
TREX	Text Retrieval and information EXtraction
UDDI	Universal Description, Discovery and Integration
UN/CEFACT	United Nations Centre for Trade Facilitation and Electronic Business
VAN	Value- Added Networks
VSE	Virtual Stock Exchange
WS	Web Services
WS-CDL	Web Services Choreography Description Language
WSDL	Web Services Description Language
WS-I	Web Services Interoperability
XI	eXchange Infrastructure
XLANG	eXtensible Language
XML	eXtensible Markup Language
XSL	eXtensible Style Language

EXECUTIVE SUMMARY

Interoperability is defined as the ability of two or more systems or components to exchange information and to use the information that has been exchanged. The advances in ICT, the cross-border eGovernment service provision as a result of EU integration, the competitiveness in a globalized world and the tendency for better administration services to business and citizens contribute in huge interest and advances in the Interoperability domain. Variations of interoperability are applied to engineering, health, telecommunications and software. Main focus domains of this guide are Enterprise Interoperability, eGovernment Interoperability, technical interoperability, semantic interoperability, organisational interoperability, business process interoperability and legal interoperability.

The tendency for coherent distributed architectures in interoperability leads to focus and development of middleware platforms. Middleware is computer software that connects software components or applications. Two different prominent types of middleware software are commercial Message Oriented Middleware (MOM) Platforms and Enterprise Service Bus (ESB) applications. Some important middleware platforms are Microsoft BizTalk Server, IBM Websphere, Oracle Application Server, SAP Netweaver.

For the successful establishment of eGovernment practices, the existence, acceptance and application of eGovernment Interoperability Framework constitutes a key factor. In this direction, Pan European eGovernment Services published in 2004 the 1st version of European Interoperability Framework (EIF), while the 2nd one is under public consultation until the end of September. The targets for EIF v.2 span across presentation layer, data representation/ encoding, middleware, platforms, database and data models, networks and programming languages.

Standardisation constitutes one of the most crucial domains in interoperability. In the area of Web Services, standards are categorized in various fields like Business Process Specifications, Metadata Specifications, Reliability Specifications, Messaging Specifications, Management Specifications, Presentation Specifications, Security Specifications, Transaction Specifications and Resource Specifications.

Decision makers on interoperability issues should bear in mind that MOM platforms are easily deployed and can cover the interoperability needs of many categories of enterprises. They should be sceptical towards ESB and consider open source solutions as an alternative. Also, their grade of adoption for Web Services should be greater as those services have already widely acceptable and established standards. Concerning policy makers, they should focus on back-office and front-office technical interoperability, security, and multilingualism. Policy makers should also direct towards common guidelines for the technical interoperability and recognized open standards. Finally, they should bear in mind that any eGovernment initiative should be assessed by professionals in public/private sector before being designed and implemented.

1. Introduction

1.1 Purpose

The objective of the G.I.C. Interoperability Guide is to provide a volume of concise and comprehensive information on Interoperability that can help its targeted audience, mainly decision makers of enterprises and policy makers of governments, to design their strategies and find solution tactics on specific interoperability problems. For this purpose the Interoperability Guide will comprise, during its six versions, material on Interoperability from a multitude of sources:

- Research results from ongoing projects and initiatives in EU and international level, focusing on the specific sub-domains and applications of Interoperability.
- Deliverables and Reports from projects, initiatives, standardisation bodies and institutions on various issues that impact the Interoperability domain.
- National, EU and global Best Practices and Strategies on specific areas of the Interoperability domain that describe how key Interoperability issues/problems have been tackled effectively by enterprises and governments.
- Legal and Statutory Framework documents, reports and white papers that affect the Interoperability domain or shape specific areas of business/governmental practice where Interoperability has a major impact.

In order to make the previous material comprehensive also to readers with limited knowledge on the Interoperability domain, the Guide incorporates declarative definitions in plain terms on the subjects it engages in its various versions and concludes by providing a set of straightforward recommendations to enterprise decision makers and government policy makers.

1.2 Structure of the Document

The Interoperability Guide is structured as following

- Section 2 describes the Interoperability domain and its various aspects, it gives insights on the various definitions on Interoperability, presents its historical overview and political context and justifies its added value for the enterprise and governmental perspective.
- Section 3, entitled Guidance to Enterprises, presents material and information on a specific aspect of the Interoperability domain that is of particular interest to Enterprises. For the purpose of this version of the Interoperability Guide the focus is in Middleware Platforms and Architectures.
- Section 4, entitled Guidance to Governments, is similar to section 3 but from the Governments perspective regarding the Interoperability aspect that is in focus. For the purpose of this version of the Interoperability Guide the focus is in the Technical Dimension of the European Interoperability Framework.

- Section 5 presents a view on the standardisation landscape on Interoperability. The objective is not to provide a thorough analysis of standards to inform the interested reader on the existing standards in a specific area that relates to the Interoperability domain. For the purpose of this version of the Interoperability Guide the area of Web Service Standards is selected.
- Section 6, concludes the Guide by presenting a set of concrete recommendations to enterprise decision makers and government policy makers based on the material and information reported in the previous sections.

2. The Interoperability Domain

2.1 What is Interoperability

IEEE broadly defines Interoperability as “*the ability of two or more systems or components to exchange information and to use the information that has been exchanged*”¹.

Although the above definition clearly views interoperability from a technical, systems engineering, sense, however this does not preclude it from being examined also in a broader sense, taking into account specific scientific or technical domains or particular social, political, and organizational factors that impact system to system performance and therefore interoperability.

Having said that, the following variations of the interoperability definition can be found in literature, in respect to the specific scientific domain they apply to:

Engineering. Apart from the former one IEEE provides also two more complimentary definition for interoperability that apply specifically in the engineering domain:

1. The capability for units of equipment to work together to do useful functions.
2. The capability, promoted but not guaranteed by joint conformance with a given set of standards, that enables heterogeneous equipment, generally built by various vendors, to work together in a network environment.

Health. In the domain of electronic health (eHealth) the European Commission proclaims interoperability should enable the integration of heterogeneous systems, allow secure and fast access to comparable public health data & patient information located in different places over a wide variety of wired and wireless services². Proportionally, the US Department of Health defines Interoperability as the ability to exchange patient health information among disparate clinicians and other authorized entities in real time and under stringent security, privacy and other protections.

Telecommunications. In the telecommunications domain interoperability is defined as the ability of systems, units, or forces to provide services to and accept services from other systems, units or forces and to use the services exchanged to enable them to operate effectively together. The condition achieved among communications-electronics systems or items of communications-electronics equipment when information or services can be exchanged directly and satisfactorily between them and/or their users. The degree of interoperability should be defined when referring to specific cases³.

Software. According to ISO/IEC 2382-01, interoperability is defined as follows: "The capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units"⁴. In

¹ Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY: 1990.

² European Commission, Communication on a European eHealth Area Com (2004) 356

³ Federal Standard 1037C, entitled Telecommunications: Glossary of Telecommunication Terms and

⁴ ISO/IEC 2382-01, Information Technology Vocabulary, Fundamental Terms

software, the term interoperability is used to describe the capability of different programs to exchange data via a common set of exchange formats, to read and write the same file formats, and to use the same protocols. In the preceding definition of interoperability in the software domain the phrase “user of a program” can also be another program.

All the foregoing definitions approach interoperability within the scope of a specific scientific domain. However taking also into consideration social, political, organizational and business parameters we come across the following definitions of interoperability that apply to the entrepreneurial and governmental domains, which are the main focus of this Interoperability Guide.

Enterprise Interoperability. According to the Enterprise Interoperability Research Roadmap Enterprise Interoperability is a relatively recent term that describes a field of activity with the aim to improve the manner in which enterprises, by means of Information and Communications Technologies (ICT), interoperate with other enterprises, organisations, or with other business units of the same enterprise, in order to conduct their business. This enables enterprises to, for instance, build partnerships, deliver new products and services, and/or become more cost efficient⁵.

In a proportionate way to the IEEE definition of interoperability, Enterprise Interoperability is the ability of an enterprise to interact with other organisations, to exchange information and to use the information that has been exchanged. It should be noted therefore that in this context interoperability is not only a property of ICT systems, but also concerns the business processes and the business context of an enterprise.

eGovernment Interoperability. European Commission defines Electronic Government (eGovernment) as the use of ICT in public administrations combined with organisational change and new skills in order to improve public services and democratic processes, and strengthen support to public policies⁶. In this scope eGovernment Interoperability is defined as the ability of Information and Communications Technologies (ICT) systems and of the business processes they support to exchange data and to enable sharing of information and knowledge⁷.

The latter two definitions in the Enterprise and eGovernment domains place interoperability as an issue of the underlying Information and Communication Technology (ICT) infrastructures of the engaged stakeholders, may these be enterprises or administrations. In this context more apt sub-definitions of interoperability can be found in the terms “technical interoperability”, “semantic interoperability”, “organisational interoperability”, “business process interoperability” and “legal interoperability”.

Technical Interoperability. Technical interoperability covers the technical issues of linking computer systems and services. Key aspects include open interfaces, interconnection services, data integration and middleware, data presentation and exchange, accessibility and security services.

⁵ European Commission Enterprise Interoperability Research Roadmap, http://cordis.europa.eu/ist/ict-ent-net/ei-roadmap_en.htm

⁶ European Commission, 2003, ‘The role of eGovernment for Europe’s future’ Communication from the Commission to the Council, the European Parliament, the European Economic and Social Committee and the Committee of the Regions, Brussels, 26.9.2003, COM(2003) 567 Final.

⁷ IDABC 2004. European Interoperability Framework for pan-European eGovernment Services. Luxembourg, European Communities.

Semantic Interoperability. Semantic interoperability ensures that the precise meaning of exchanged information is understandable by any other application that was not initially developed for this purpose. It thus enables systems to combine received information with other information resources.

Organisational Interoperability. Organisational interoperability is concerned with defining business processes and bringing about the collaboration of organisations that wish to exchange information and may have different internal structures and processes, as well as aspects related to requirements of the user community.

Business Process Interoperability. Business process interoperability is a state that exists when a business process can meet a specific objective automatically utilizing essential human labor only. Typically, business process interoperability is present when a process conforms to standards that enable it to achieve its objective regardless of ownership, location, make, version or design of the computer systems used. The main attraction of business process interoperability is that a business process can start and finish at any point worldwide regardless of the types of hardware and software required to automate it. Because of its capacity to offload human "mind" labour, business process interoperability is considered by many as the final stage in the evolution of business computing.

Legal Interoperability. In addition to the above definitions, the European Interoperability Framework in its 2nd (draft version) introduces also the dimension of legal interoperability. Legal interoperability involves the appropriate synchronization of the legislation in the cooperating EU Member States and countries so that electronic data originating in any given country is accorded the proper legal weight and recognition wherever it needs to be used in other countries.

In fact, interoperability is often confused with other, related concepts. It can be therefore a useful exercise to observe explicitly what interoperability is NOT:

- Interoperability is not Integration, which is a means of changing loosely coupled systems to make them into more tightly coupled systems.
- Interoperability is not Compatibility, which is more about the interchangeability of tools in a particular context
- Interoperability is not Adaptability, which is a means of changing a tool, adding additional capabilities as needed even on an ad-hoc basis, whereas interoperability refers to inherent capabilities

It is also worth noting that interoperability is neither ad-hoc, nor unilateral (nor even bilateral) in nature. Rather, it is best understood as a *shared value of a community*.

The final point to be made about interoperability from the definition standpoint, is that it is also a quality that could be broken down into a series of quantifiable characteristics (metrics) which could be assessed (measured) separately, as the need arises.

2.2 Historical Overview and Political Context of Interoperability

Since the 1960s, many companies developed in-house computer systems and internal networks to streamline business functions. Still, the speed in which a business could respond was determined by the communication link between the company and its customers. That communication link consists

of the postal service and the telephone, and remains a slow and costly process even today. Some business executives were working on methods to shortcut the conventional communication link. Electronic communications was a prime consideration in circumventing the paperwork/telephone problems. It soon became clear that linking up to other business electronically had one major initial problem, information format.

The Electronic Data Interchange (EDI) emerged in the 1960s when the railroad industry sought a way to speed up and automate business communications between remote computer systems and to eliminate the high cost of sending paper documents by snail mail. However, the concept did not fully take hold industry wide until the 1980s when standards were introduced to define data exchange and when it became apparent that business-to-business (B2B) computer-mediated communications would require all parties to adopt a common protocol for purchase orders, advance shipping notices (ASN), and other pertinent documents. As a result, several technical committees have developed protocols governing such exchanges, which channeled through value-added networks (VANs) carrying these exchanges became collectively known as EDI.

Transportation, finance, insurance, and other industries have heavily leveraged EDI and proprietary communications lines to conduct business. Also, major manufacturers, such as automotive original equipment manufacturers (OEMs) and consumer packaged goods (CPG) companies have embraced EDI and mandated that their suppliers do the same. However, implementation of the technology has initially caused many a headache particularly in smaller companies.

The emergence of various XML based data standards during the last decade promised to solve the main drawbacks of traditional EDI-based technologies like very specific expert knowledge, non flexibility, expensiveness and complexity. Instead, the numerous XML based languages that have emerged during the last years even increased the complexity inherent to Enterprise Application Integration (EAI) and data mapping.

Since 2000, there have been massive changes in the overall direction of Application Integration in general and EAI in particular. A vast number of interoperability standards and frameworks have arisen in the light of fulfilling the need of automated collaboration. Some frameworks provide architectural guidelines that support the interoperability of inter-enterprise systems, such as the RosettaNet Framework, the OAG Integration Specification, and FIPA's architecture and associated set of specifications for distributed, communicating software agents. Among the most well-known is ebXML, a set of specifications supported by OASIS and UN/CEFACT, and proposed by a large group of businesses, government standards committees and academics to enable a well structured electronic business framework. Almost every established industry sector has set up organisations that have developed sector specific specifications for business-to-business (B2B) transactions within the industry, such as ODETTE in automotive, CIDX in chemical, GS1/UCC in retail and PIDX in petroleum. On the technical side, the Service Oriented Computing paradigm and Service Oriented Architectures (SOA) have emerged as an evolutionary step from Object and Component based approaches, with the promise to overcome the deficiencies of past solutions, real or perceived.

From a policy point of view, the i2010 Strategic Framework recognises the importance of Interoperability. This Framework is the logical link between the high-level goals of the Lisbon Strategy and more operational ICT-related actions. The i2010 Framework recognises that "businesses are getting productivity gains from ICT but still face a lack of interoperability, reliability and security, difficulties to reorganise and integrate ICT into the workplace and high cost

of support”. Interoperability is explicitly identified as one of the key bottlenecks that should be tackled by i2010 in order to create a single European information space and make the European Union more competitive⁸.

Interoperability has also been recognised as a key research area by the FP7 – ICT Work programme (Area 1.3), the European Commission Enterprise Interoperability Research Roadmap⁹, as well as the IDABC eGOV Research Roadmap¹⁰ and the FP6 eGovRTD2020 Project e-Government Research Roadmap¹¹.

Furthermore, the role of interoperability of organisations and systems has been recognised by the European Commission through the creation of the European Interoperability Framework¹² (EIF) and the respective e-Business Interoperability Framework¹³ (eBIF) as evolving tools for guiding administrations and industries.

From a national viewpoint, the Greek Digital Strategy¹⁴ recognizes interoperability as a core pillar towards the Information Society of 2013, while latest IDABC reports underpin this thesis stating that interoperability is a vital issue in Greece.

2.3 Why Interoperability Matters

Today, escalating economic and societal demands, along with the continuous advancements in Information and Communication Technologies (ICT), set a growing agenda for enterprises and governments and challenge the capabilities of their underlying technical infrastructures to support them effectively in their effort to adopt new models of operation – such as the paradigms of eBusiness and eGovernment – that will enhance their competitiveness and efficiency.

In the business domain, novel business practices such as the paradigm of electronic transactions (eTransactions) are constantly taking up momentum. Worldwide the number of stakeholders, from 10person Small-Medium Enterprises (SMEs) to global 2000 companies or local and regional administrations to national and federal governments, who are currently in the process of implementing eTransaction infrastructures, is ever growing. Proportionally to Porter’s statement in 2001 that it was not a question for enterprises whether or not to move to the Internet but when and how to do it in order to create new value, today the “silver bullet” decision for businesses and organisations does not lie any more in the dilemma of adopting or not ICT enabled eTransaction and collaboration practices but how to integrate them as quickly and effectively as possible to their operation to achieve the biggest possible benefits.

⁸ The i2010 Strategy Framework, http://ec.europa.eu/information_society/eeurope/i2010/index_en.htm

⁹ European Commission Enterprise Interoperability Research Roadmap, http://cordis.europa.eu/ist/ict-ent-net/ei-roadmap_en.htm

¹⁰ IDABC, <http://ec.europa.eu/idabc/>

¹¹ eGovRTD2020 Project,

http://www.egovrtd2020.org/EGOVRTD2020/navigation/work_packages/wp4_roadmapping/D41

¹² European Interoperability Framework, <http://ec.europa.eu/idabc/servlets/Doc?id=19528>

¹³ eBusiness Roadmap: addressing key eBusiness standards issues 2006-2008,

<http://www.cen.eu/cenorm/businessdomains/businessdomains/iss/activity/ebusfinal.pdf>

¹⁴ Greek Digital Strategy 2006-2013, <http://www.infosoc.gr/infosoc/en-UK/sthnellada/committee/default1/top.htm>

Despite though the willingness of stakeholders to proceed to such networked enterprise paradigms and although technical solutions to enable e-transactions have been thoroughly justified during the last years their adoption and application into the everyday business practice by enterprises still remains limited. Technical and business characteristics of the proposed solutions, such as inflexible workflows, predefined formats for the exchanged data, predefined business and legal rules, exchange of business information through third party systems, use of proprietary technologies and inability to be readily set up and deployed act as the main inhibitors for the potential users. This renders current solutions costly, rigid and difficult to adapt to meet the requirements of evolving enterprise.

The past decade has seen significant advances to Enterprise Interoperability, particularly those related to ICT infrastructure aspects. For example, the Internet (and browser technology) has facilitated the search for information, and the exchange of information among enterprises. It has also contributed to the creation of new business models of Enterprise Interoperability. In addition, a number of software suppliers (e.g. IBM, Microsoft, Oracle and SAP) have gained a de facto ascendancy in the enterprise software market, contributing to the integration and efficiency gains of enterprise functions. However, questions remain about the impact and significance of these vendor-based solutions to Enterprise Interoperability. More than a decade after Enterprise Interoperability issues have been raised and discussed within various communities, interoperability is still a problem for enterprises. Islands of interoperability persist. Integration projects remain complex and expensive. The business case for interoperability is often not apparent to potential adopters of Enterprise Interoperability solutions, particularly for SMEs. Various technologies and tools resulting from research lack follow-up beyond (further) research. Large question marks remain as regards the “value” and “impact” of the myriad of initiatives undertaken within the research lab, promoted by technology providers, or organised around groupings of companies.

Today Enterprise Interoperability reaches all enterprises at national and international level and constitutes a thriving research domain from all aspects – technical, entrepreneurial, societal and political. Lack of interoperability appears as the most long lasting and challenging problem for enterprises and governmental organizations. It emerged from proprietary development or extensions, unavailability or oversupply of standards, and heterogeneous hardware and software platforms. To meet their business objectives, enterprises need to collaborate with other enterprises: for many enterprises, doing business globally has become critical to their survival, while others (mainly governmental organizations and SMEs) discover new opportunities by focusing their business in a local setting. The situation has become more critical and important through new business paradigms like extended enterprises and networked organisations that require organizations to work together to achieve further benefits. Therefore, today an organization’s competitiveness is to a large extent determined by its ability to seamlessly interoperate with others.

In trying to characterize the current problem space for Enterprise Interoperability, EU’s Enterprise Interoperability Research Roadmap identifies the following relevant dimensions where Interoperability arises as an issue of high importance for contemporary enterprises.

- Managing more rapid change/innovation
- Adapting to globalisation
- Large integration/interoperability costs

- Difficulties in decision making (e.g. when to interoperate with other enterprises)
- Lack of business case for Enterprise Interoperability
- A change in the model of collaboration towards open innovation.

In the governmental domain, whereas in the last decade the main task of eGovernment initiatives focused on just the vertical integration aspects of the public sector's service provision, nowadays, fully integrated, both vertically and horizontally, one-stop, electronic services are gradually becoming a reality, or they are considered, at least, a feasible goal. Such services are called Pan European eGovernment Services (PEGS), which means cross-border public sector information and interactive services, either sectoral or horizontal, i.e. of cross-sectoral nature, provided by European public administrations to European public administrations, businesses, including their associations, and citizens, including their associations, by means of interoperable trans-European telematic networks¹⁵.

The main issues that arise regarding the provision of such services by national public administrations have to do with the complexity, the multiplicity and the diversity of public sector's organisations. Considering the added complexity of procedures, information needs and systems, technologies used, legal frameworks, language and other special regional needs between the European Member States and the thousands of front and back-office systems which have to be taken into account in order to achieve this, the need for inclusive e-Government Interoperability Frameworks (eGIFs) becomes apparent. The interoperability challenge is becoming gradually even a more urgent matter, as the need for modernisation in the public sector has led to many small-scale projects that have not been based on any interoperability standards, which means that the information and the services they provide are not easily accessible by, or compatible to, information systems, technologies or business processes in other public-sector organisations, or even other offices within the same organisation.

An Interoperability Framework describes the way in which organisations have agreed, or should agree, to interact with each other, and how standards should be used. In other words, it provides policies and guidelines that form the basis for selection of standards⁶. It may be contextualised (i.e., adapted) according to the socio-economic, political, cultural, linguistic, historical and geographical situation of its scope of applicability in a specific circumstance/situation (a constituency, a country, a set of countries, etc).

In an effort to promote the provision of Pan European eGovernment Services, EU published in 2004 the 1st version of the European Interoperability Framework¹⁶ – for which recently put on consultation a 2nd draft version¹⁷. In this context, many Member States already have or are in the process of developing their own National IF's, (NIF's) addressing interoperability issues arising within their own country, across internal borders, between national agencies, departments, government bodies, etc. These national frameworks are complementary to the European Interoperability Framework, yet they should be compatible with it. The EIF and the NIF's complement one another in the sense that the EIF is concerned with PEGS at EU level, whereas the NIF's are concerned with both PEGS and non-PEGS, but only at the national level.

¹⁵ Article 3b of the Decision 2004/387/EC of the European Parliament and of the Council on 21st of April 2004

¹⁶ <http://ec.europa.eu/idabc/en/document/3473>

¹⁷ Revision of the EIF and AG, <http://ec.europa.eu/idabc/en/document/7728>

All the more the high-priority nature of achieving interoperability in the cross-border, cross-sector domain becomes evident. To implement pan-European eGovernment services, the public sector must confront many challenges, some of which are quite daunting. The realisation of interoperability, especially of the cross-border and cross-sector type, is now recognised as being a key factor in securing these objectives.

Summing up, some of the most important challenges that modern public administrations have to face have interoperability as one their core dimensions, specifically:

- Rapid advances in ICT, including several paradigm shifts, have transformed the landscape in which administrations, businesses and citizens interact with one another to an unprecedented degree. As a result, citizens and businesses are demanding ever more and better services from their governments.
- At the political level, advancing EU integration has placed dramatically increased emphasis on the cross-border aspects of eGovernment service provision, PEGS.
- Globalization is creating an ever more integrated and competitive environment for EU businesses and workers, resulting in increasing economic pressures which have been followed by major priority shifts in EU policies putting interoperability as one of the key achievements (e.g., the Lisbon agenda, eGovernment 2005 Action Plan, i2010 Strategy Framework, etc.)
- Administrations are consequently under tremendous political pressure to streamline their activities, modernize their infrastructure, and integrate their activities all intended to provide better, faster, cheaper services to businesses and citizens; eGovernment programs have accelerated tremendously and moved to centre stage.

Collectively these forces have tremendously increased the importance of interoperability in all its aspects.

2.4 Focus of the Interoperability Guide

As it comes out from the previous analysis Interoperability is a vast domain, there is almost one definition for every scientific domain and several sub-definitions for the various sub-domains of Enterprise and eGovernment Interoperability. Consequently an effort to “squeeze” such a broad domain of knowledge in one single report in order to provide an all-inclusive guide would be futile and at best would render the document elaborate, lengthy, verbose and finally useless for its potential audience.

Making use of the fact that according to the G.I.C.’s description of work this Interoperability Guide will be published in six version, it was chosen that each version of the Guide would focus in a specific aspect of Enterprise and eGovernment Interoperability with the objective to provide a set of comprehensive reports and straightforward recommendations on this aspect of Interoperability that could be used by decision makers and policy makers in enterprises and organisations.

As the Interoperability domain quite extended and multilateral – in the APPENDIX and indicative taxonomy of the domain that is proposed by G.I.C. and will be used for organising the knowledge within the project is presented – the Interoperability Guide adopts the refinement of Interoperability into the sub-domains of organisational, legal, semantic and technical interoperability in order to concentrate in a specific theme in every section.

The following figure illustrated the focus of the current version of the Interoperability Guide.

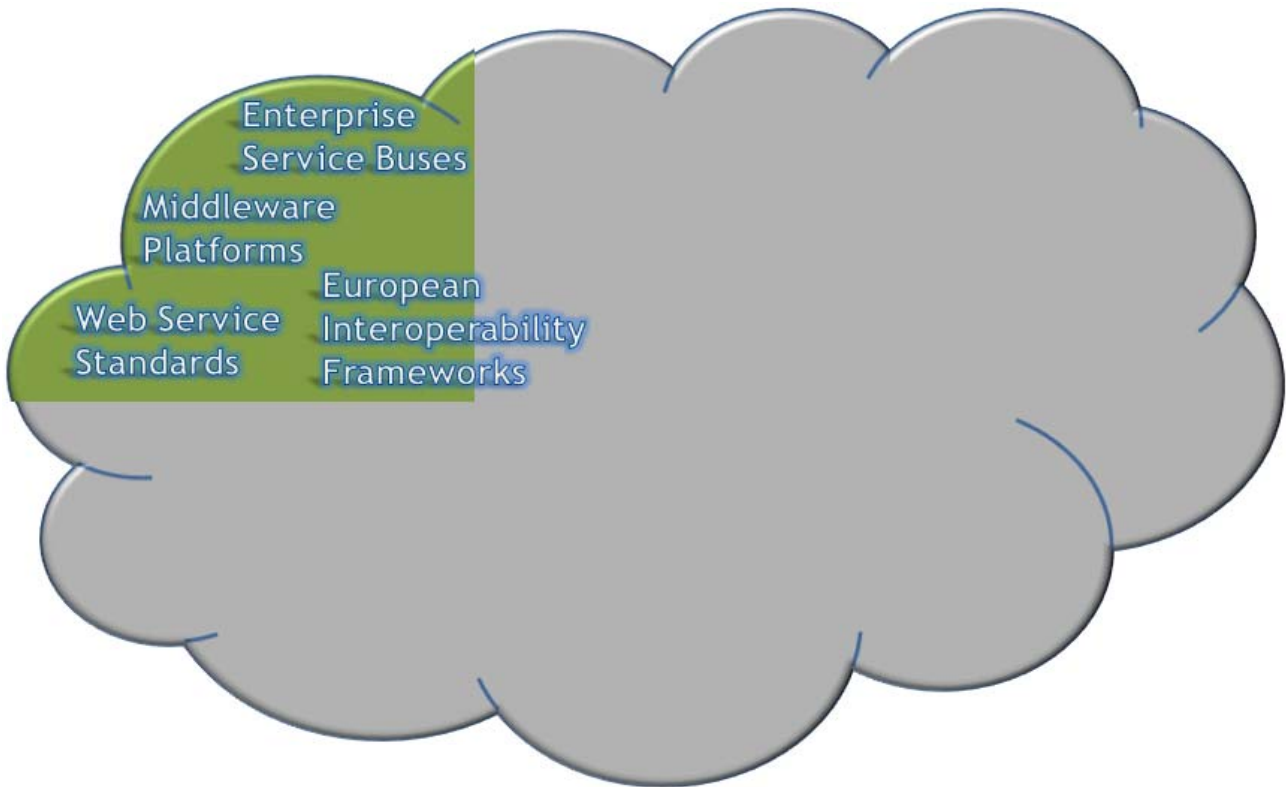


Figure 1: Focus of the Interoperability Guide

3. Guidance to Enterprises

The purpose of this section is to provide concise reports on the current state of the art and state of practice in Interoperability that can be used by businesses, organisations, small and medium enterprises and even individual entrepreneurs as a starting point for going after a solution that matches their interoperability needs. It is evident that the terms current “state of the art” and “state of practice” in interoperability denote a vast area of knowledge comprising technologies, products, tools, research results, best practices and any effort to put everything even in a codified form would be futile. Therefore, as it has been noted, each version of the Interoperability Guide will focus in this section in a particular sub-domain of interoperability trying to provide as much as possible an inclusive description of it.

For the purpose of this version of the Interoperability Guide the focus is in **Middleware Platforms and Architectures**.

3.1 Middleware Definition¹⁸

Middleware is a computer software that connects software components or applications. The software consists of a set of enabling services that allow multiple processes running on one or more machines to interact across a network. This technology evolved to provide for interoperability in support of the move to coherent distributed architectures, which are used most often to support and simplify complex, distributed applications. It includes web servers, application servers, and similar tools that support application development and delivery. Middleware is especially integral to modern information technology based on XML, SOAP, Web services, and service-oriented architecture. It sits "in the middle" between application software working on different operating systems. It is similar to the middle layer of a three-tier single system architecture, except that it is stretched across multiple systems or applications. Examples include database systems, telecommunications software, transaction monitors, and messaging-and-queuing software.

Two very important types of middleware with significant research and market applications are the Message Oriented Middleware (MOM) Platforms and the Enterprise Service Bus (ESB) architecture and its implementations.

*Message Oriented Middleware Platforms*¹⁹. Message-oriented middleware (MOM) is a client/server infrastructure that increases the interoperability, portability, and flexibility of an application by allowing the application to be distributed over multiple heterogeneous platforms. It reduces the complexity of developing applications that span multiple operating systems and network protocols by insulating the application developer from the details of the various operating system and network interfaces. APIs that extend across diverse platforms and networks are typically provided by the MOM. MOM is software that resides in both portions of client/server architecture and typically supports asynchronous calls between the client and server applications. Message queues provide temporary storage when the destination program is busy or not connected. MOM reduces the involvement of application developers with the complexity of the master-slave nature of the client/server mechanism. MOM comprises a category of inter-application communication software

¹⁸ <http://en.wikipedia.org/wiki/Middleware>

¹⁹ http://en.wikipedia.org/wiki/Message_Oriented_Middleware

that generally relies on asynchronous message-passing, as opposed to a request-response metaphor. Most message-oriented middleware depend on a message queue system, but there are some implementations that rely on broadcast or multicast messaging systems. Currently there are in the market quite a few many MOM product, from simple modelling tools to fully integrated commercial-of-the-shelf (COTS) products for the exchange of electronic transactions. In the following paragraphs we are going to examine some of the most prominent cases. The reader who is interested in finding a suitable MOM platform solution for its enterprise in terms of cost, size and performance can easily find more information in the literature and on the internet. The purpose of the following descriptions are to provide an indication of the interoperability capabilities of current MOM platforms for businesses.

*Enterprise Service Bus*²⁰. An enterprise service bus (ESB) refers to a software architecture construct. This construct is typically implemented by technologies found in a category of middleware infrastructure products, usually based on recognized standards, which provide fundamental services for more complex architectures via an event-driven and standards-based messaging engine (the bus). An ESB generally provides an abstraction layer on top of an implementation of an enterprise messaging system, which allows integration architects to exploit the value of messaging without writing code. Contrary to the more classical enterprise application integration (EAI) approach of a monolithic stack in a hub and spoke architecture, the foundation of an enterprise service bus is built of base functions broken up into their constituent parts, with distributed deployment where needed, working in harmony as necessary. An ESB does not implement a service-oriented architecture (SOA) but provides the features with which one may be implemented. Although it is a common belief, an ESB is not necessarily web-services based, is should be standards-based and flexible, supporting many transport mediums. Based on EAI rather than SOA patterns, it tries to remove the coupling between the service called and the transport medium. However most ESB providers now build ESBs to incorporate SOA principles and increase their sales.

As the ESB architecture is of crucial important in designing and implementing better interoperability solution we will examine its characteristics further in a following section.

3.2 Microsoft BizTalk Server²¹

Microsoft BizTalk is a business process management (BPM) server that enables companies to automate and optimize business processes. This includes tools to design, develop, deploy, and manage those processes.

Microsoft BizTalk Server can be used to design, build, and execute dynamic business interactions that span applications, platforms, and organizations. The current version of BizTalk supports the XLANG specification for modelling business processes. The product includes an orchestration engine for executing and monitoring processes. It also includes a very robust visual development environment, BizTalk Orchestration Designer, for defining and connecting processes. With BizTalk, a distinct separation is made between the process definition and implementation. This

²⁰ http://en.wikipedia.org/wiki/Enterprise_Service_Bus

²¹ <http://www.genesis-ist.eu>, Deliverable D5.1 State of the Art Analysis

provides additional flexibility for dynamically changing the process flow or the services used within the interaction.

Bindings within BizTalk are done using ports, similar to WSDL portTypes for sending and receiving messages. Process bindings can be built for COM components, MSMQ queues, script components, and other BizTalk server processes. New versions of BizTalk include more integrated support for .NET where XLANG schedules can be compiled as .NET assemblies within the framework. BizTalk provides the basic primitives for conditional branching, sequential, and parallel execution. It also includes support for long-running transactions. A transactional context can be defined with appropriate compensation logic if parts of the transaction fail. The product includes robust management and monitoring support, with the ability to query a process state, manage processes, and debug them.

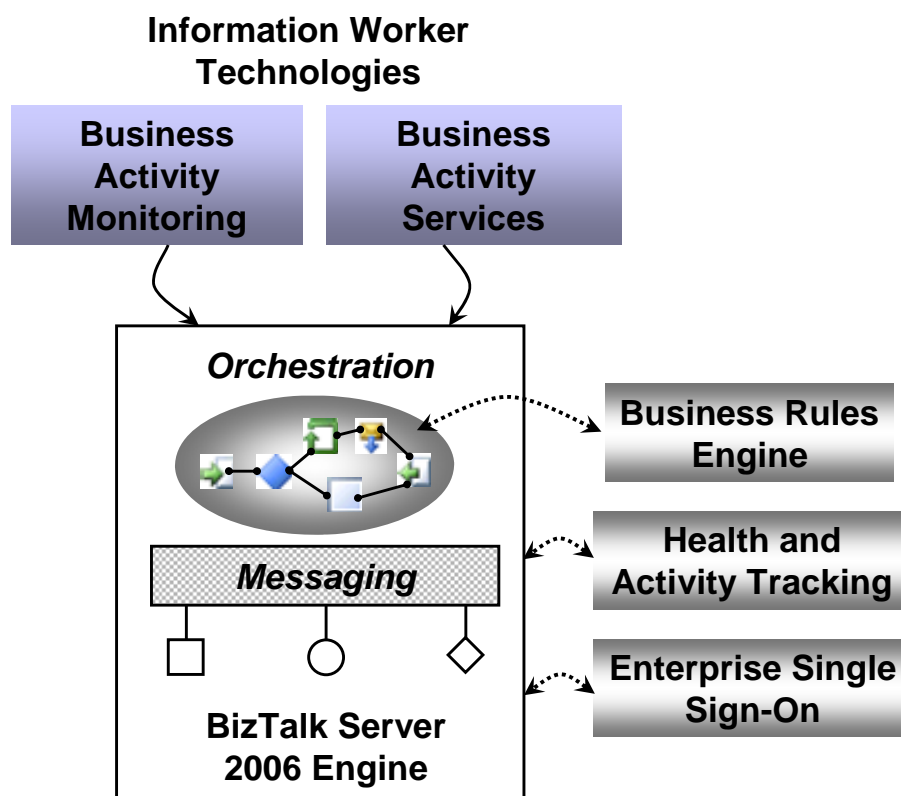


Figure 2: BizTalk Server architectural overview

The engine has two main parts:

- A messaging component that provides the ability to communicate with a range of other software. By relying on pluggable adapters for different kinds of communication, the engine can support a variety of protocols and data formats, including Web services and many others.
- Support for creating and running graphically-defined processes called orchestrations. Built on top of the engine's messaging components, orchestrations implement the logic that drives all or part of a business process.

Several other technologies can also be used in concert with the engine, including:

- A Business Rules Engine that allows evaluating complex sets of rules.
- A Health and Activity Tracking tool that lets developers and administrators monitor and manage the engine and the orchestrations it runs.
- An Enterprise Single Sign-on facility, providing the ability to map authentication information between Windows and non-Windows systems.

On top of this foundation, BizTalk Server 2006 provides a group of technologies that address the more business-oriented needs of information workers. Those technologies are:

- Business Activity Monitoring, allowing information workers to monitor a running business process. The information is displayed in business rather than technical terms, and what gets displayed can be controlled directly by business people.
- Business Activity Services, allowing information workers to set up and manage interactions with trading partners.

Integrating existing applications, whether in a single company or across different organizations, into a single automated business process is a fundamental goal of BizTalk Server. Once those automated processes exist, the product also gives business people, not just technicians, visibility into what's happening inside the process. In the complex and diverse world of enterprise software today, this kind of integration is a necessity for many organizations.

The goal of BizTalk Server is to help organizations meet the challenges of creating automated business processes that rely on diverse systems. The product's foundation is the BizTalk Server engine, which provides core messaging and orchestration capabilities. From its initial roots in EAI and B2B integration, BizTalk Server has grown into the foundation for supporting a range of business processes.

3.3 IBM Websphere²²

IBM WebSphere is a suite of IBM software products that are designed to set up, operate and integrate e-business applications across multiple computing platforms using Web technologies. The suite includes the products IBM Websphere MQ, IBM Websphere Application Server, IBM Websphere Message Broker, IBM Websphere Enterprise Service Bus.

IBM WebSphere MQ delivers messaging for SOA. It provides the messaging backbone for SOA connectivity, as the ubiquitous, multi-purpose data transport for the enterprise service bus (ESB).

- Connects applications and Web services across many commercial IT systems, and provides full JMS (Java Message Service) support including Publish-Subscribe.

²² <http://www.genesis-ist.eu>, Deliverable D5.1 State of the Art Analysis

- Provides integrated support for Web services.
- Provides eclipse-based tooling – MQ Explorer – for Windows and Intel (x86) enabling remote and secure configuration of entire messaging backbone.
- Interoperates seamlessly with the messaging services of WebSphere Application Server.
- Supports industry standard Secure Sockets Layer (SSL) security and offers an Extended Security Edition with advanced security features.
- Enables to moving existing FTP infrastructure forward, ensuring reliable, secure file transfer over WebSphere MQ.

IBM WebSphere Message Broker is built for universal connectivity and transformation in heterogeneous IT environments. It distributes information and data generated by business events in real time to people, applications, and devices throughout the enterprise and beyond.

- Provides a SOA approach to, extending the enterprises firewall by supporting a broad range of multiple transport protocols and data formats
- Integrates multiple applications, networks, and device types using a platform-independent based enterprise service bus that lets enterprises conduct business reliably and securely
- Increases business agility and flexibility, extending to a Federated ESB model, while reducing development costs by separating integration logic from applications
- Improves the flow of information around the business, moving away from hard-coded point-to-point links to more flexible distribution mechanisms such as publish/subscribe and multi-cast
- Uses a simple programming model for connectivity and mediation, including a robust set of pre-built mediation function and ways to customize mediations
- Exploits the WebSphere MQ messaging infrastructure, and supports transformation options with graphical mapping, Java, ESQL, XSL, and WebSphere Transformation Extender
- Delivers extensive administration and systems management facilities for developed solutions

IBM WebSphere Application Server is a middleware software that is designed to set up, operate and integrate e-business applications across multiple computing platforms using Web technologies. Websphere Application Server is built using open standards such as J2EE, XML, and Web Services. Multiple world-wide IBM labs participate in creating WebSphere run-time products and development tools. It works with a number of Web servers including Apache HTTP Server, Netscape Enterprise Server, Microsoft Internet Information Services (IIS), IBM HTTP Server for i5/OS, IBM HTTP Server for z/OS, and IBM HTTP Server for AIX/Linux/Microsoft Windows/Solaris.

IBM WebSphere Enterprise Service Bus provides Web services connectivity and JMS messaging. It applies Web services connectivity and JMS messaging, improving flexibility through the adoption of service-oriented interfaces.

Provides a comprehensive approach to SOA, delivering a standards-based connectivity and integration solution that allows enterprises to create and deploy interactions quickly and easily between applications and services, with a reduced number and complexity of interfaces.

Offers easy-to-use tools that require minimal programming skills and is simple to install, configure, build and manage.

- Supports many ISV solutions via WebSphere Adapters.
- Provides leadership in SOA standards for service composition, mediation, and hosting.
- Increases business agility and flexibility extending to a Federated ESB model.
- Re-configures dynamically to meet changing business processing loads. Provides interactions with any JMS and HTTP applications.
- Integrates seamlessly with the WebSphere platform as well as products within the IBM SOA Foundation, such as IBM Tivoli Composite Application Manager for SOA.

3.4 Oracle Application Server²³

Oracle Application Server is a significant new release of the core service-oriented architecture platform underlying Oracle Fusion Middleware. It is designed to provide a standards-based, mission critical platform for organizations deploying service oriented architectures.

According to Oracle, a Responsive Software Infrastructure for Enterprise Applications must provide the ability to (i) Develop Enterprise Applications at Lower Cost; (ii) Enable Streamlined Business Processes that can be Quickly Optimized in Response to Events; and (iii) Make Employees more Productive by providing them an efficient Workplace to access information and do work. Oracle Application Server is designed to address these three challenges:

- *Service Oriented Development of Applications (SOA)*. It provides a productive and open Application Development Framework; a comprehensive J2EE standards-based SOA runtime, and facilities to service-enable existing applications and legacy systems without rewriting any of the applications.
- *Event-driven Business Process Optimization*. It provides facilities to synchronize data between systems; to integrate systems within the Enterprise (EAI) and with partners (B2B); to automate business processes (BPM); and to monitor and optimize business processes in response to events.

²³ <http://www.genesis-ist.eu>, Deliverable D5.1 State of the Art Analysis

- *Unified Workplace with Pervasive, Multi-channel Access.* It provides pervasive access from anywhere, any time, and from any device to an Enterprise Portal that provides unified access to Information, Services, Business Processes, and Business Intelligence; and a productive and collaborative Workplace for employees.

Grid Computing is a new software architecture designed to Pool Low Cost Modular Storage and Servers to create a “virtual computing resource” across which work can be transparently distributed. It allows computing capacity to be used very efficiently, at low cost, to deliver very high performance and high availability. The resources in a grid can include storage, servers, databases, and also Application Servers and Enterprise Applications. Oracle Application Server is designed to leverage Grid Computing to lower costs associated with deploying and Managing Enterprise Applications:

- *Enterprise Quality of Service on Commodity Computing Grids.* It provides enterprise-levels of Performance, Scalability, and High Availability using commodity hardware and storage. It saves costs by lowering computing capacity requirements and enabling modular, inexpensive capacity growth.
- *Lower Cost Systems Management.* It lowers management costs and provides better business continuity by automating Software Provisioning; centralizing Monitoring; and enabling Policy-based Administration of sets of systems.
- *Lower Cost Security Management.* It provides a secure platform for Enterprise Applications and lowers the cost of user management by centralizing identity and access management.

Oracle Application Server 10g R3 introduces the Oracle Enterprise Messaging Service (OEMS). This is the next generation infrastructure based on the JMS 1.1. foundation provided by the Oracle Containers for J2EE. OEMS is built on Java 2 Enterprise Edition (J2EE) standards such as the Java Message Service (JMS) and the J2EE Connector Architecture (JCA) and is designed to reduce the time, cost, and effort required to build message oriented integrated distributed applications.

A distributed messaging environment requires dependable and flexible message delivery between applications residing on local or remote servers. If the server for a message end point is down then the sending server should dependably store messages until the destination server is back up. Furthermore, these architectures often require the flexible integration of dissimilar messaging systems. The built-in JMS Router meets these requirements by providing guaranteed message propagation between the Oracle JMS in-memory and file-based systems to the Oracle JMS database system, WebSphereMQ, Tibco Enterprise JMS, and SonicMQ message systems.

Oracle Application Server 10g R3 ships Oracle Business Rules natively as part of the runtime. Oracle Business Rules allows application developers to add agility and transparency to their applications by allowing business analysts, without depending on programmers, to directly effect application changes reflecting new business policies. Oracle Business Rules is especially suited to deployment as part of BPEL applications in particular, SOA applications in general and other architectures where agility is important.

Oracle Application Server 10g R3 Web Services provides a new runtime infrastructure supporting J2EE 1.4 Web services. The Web services runtime fully leverages the scalability, reliability and performance characteristics of core Oracle Application Server 10g R3 environment. In addition to supporting publishing and consuming Java Web services the Oracle Application Server Web Services environment also enables declarative quality of service characteristics on those service

endpoints such as WS-Security, WS-Reliability, content based logging and auditing. The Oracle Application Server Web Services 10.1.3 framework is used across the Oracle platform in a variety of component areas such as the Oracle BPEL Process Manager, Oracle Application Development Framework, Enterprise Service Bus and Web Services Manager as foundation Web services infrastructure in addition to being a standalone developer platform for developing Web services.

A significant effort in J2EE 1.4 Web services was ensuring that Web services built with JAX-RPC and SAAJ could easily conform to the WS-I Basic Profile. By conforming to the WS-I Basic Profile, developers have a high certainty that their Web services will interoperate across heterogeneous Web services implementations. By default, Web services built with Oracle Application Server 10g R3 Web Services conform to the WS-I Basic Profile 1.1. Further, Oracle has also done the same interoperability certification with its WS-Security implementation conforming to the WS-I Basic Security Profile 1.0.

3.5 SAP Netweaver²⁴

SAP NetWeaver is an application builder platform from SAP for integrating business processes across various systems, databases and sources. It is the technological foundation for all SAP products of the SAP Business Suite. SAP NetWeaver is a service-oriented application and integration platform (i.e. SAP NetWeaver is the interface between SAP applications and is also their runtime environment). Furthermore, it can interoperate with various technologies and platforms, e.g. Microsoft .NET, Sun Java EE, and IBM WebSphere. SAP is fostering relationships with system integrators and technology providers, many of the latter becoming "Powered by SAP Netweaver". Being part of all current SAP Business Suite products, SAP NetWeaver will become the de-facto standard platform for enterprise applications.

SAP Netweaver can be seen as part of SAP's strong commitment to service orientation. It builds the technical foundation for SAP's E-SOA (Enterprise SOA). Future SAP products will be fully service-enabled. From a technical point-of-view, they build upon the SOA approach. E-SOA leverages the technical SOA by giving all services within an enterprise a clearly defined semantics.

NetWeaver is a stack of SAP tools and products that are provided under a single license. NetWeaver includes a group of relatively independent applications running on a single technical platform, SAP Web Application Server (Web AS). SAP Web AS is the runtime environment for all SAP enterprise applications. It provides runtime engines for ABAP and Java. The list of products is:

- SAP Web Application Server
- SAP Exchange Infrastructure (XI)
- SAP Enterprise Portal
- SAP Master Data Management (MDM)
- SAP Mobile Infrastructure (MI)
- SAP Business Intelligence (BI) aka SAP Business Information Warehouse

²⁴ <http://www.genesis-ist.eu>, Deliverable D5.1 State of the Art Analysis

- SAP Knowledge Warehouse (KW)
- Collaboration
- TREX Engine
- SAP Composite Application Framework - an environment for designing and using composite applications
- SAP Web Application Server (Web AS)

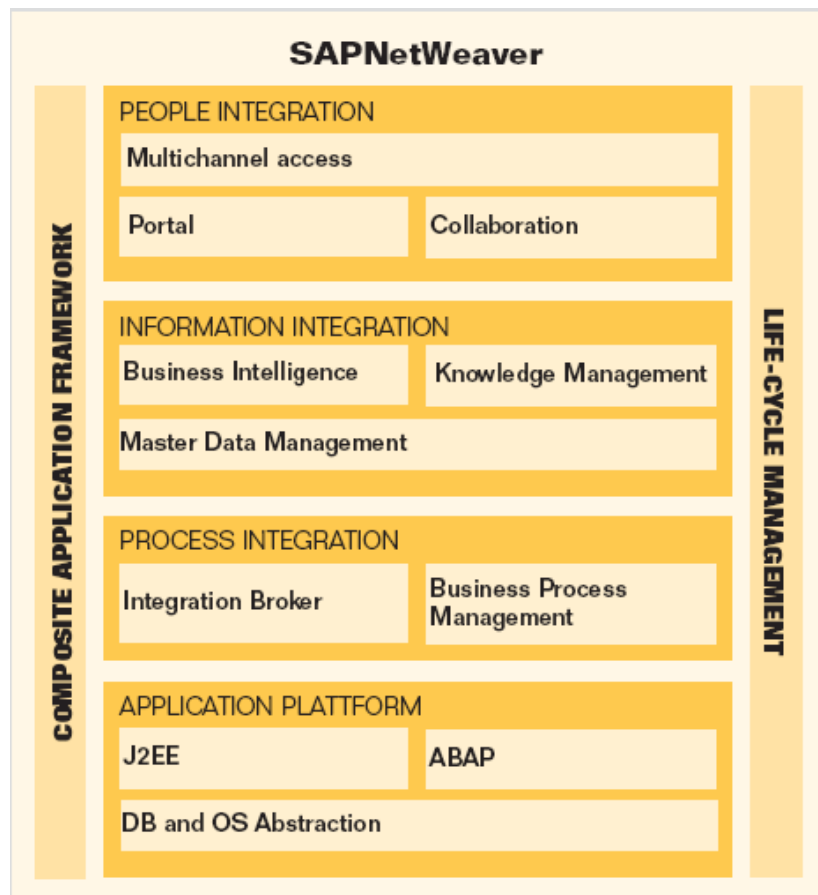


Figure 3: SAP NetWeaver Integration Framework

SAP NetWeaver promises interoperability with IBM and Microsoft solutions. SAP is cooperating with them in development strategies, field engagements, and competency and support centres. SAP has delivered an integration and application platform that is designed to be fully interoperable with IBM WebSphere and Microsoft .NET. This integration applies a holistic approach, that covers all three integration layers – people, information, and business processes – while relying on an application platform based on open standards.

3.6 ESB Architecture Overview²⁵

The ESB is an open, standards-based message bus designed to enable the implementation, deployment, and management of SOA-based solutions with a focus on assembling, deploying, and managing distributed SOA. The ESB provides the distributed processing, standards-based integration, and enterprise-class backbone required by the extended enterprise. The ESB is designed to provide interoperability between large grained applications and other components via standards-based adapters and interfaces. The bus functions as both transport and transformation facilitator to allow distribution of these services over disparate systems and computing environments.

Conceptually, the ESB has evolved from the store and-forward mechanism found in middleware products, e.g., Message Oriented Middleware, and combines conventional EAI technologies with Web services, XSLT, orchestration, and choreography technologies, e.g., BPEL, WS-CDL, and ebXML BPSS. Physically, an ESB provides an implementation backbone for an SOA. It establishes proper control of messaging as well as applies the needs of security, policy, reliability, and accounting, in an SOA architecture. The ESB, is responsible for the proper control, flow, and translations of all messages between services, using any number of possible messaging protocols. An ESB pulls together applications and discrete integration components to create assemblies of services to form composite business processes, which in turn automate business functions in an enterprise.

The following figure depicts a simplified architecture of an ESB that integrates a J2EE application using JMS, a .NET application using a C# client, an MQ application that interfaces with legacy applications, as well as external applications and data sources using Web services. The ESB architecture, as portrayed in the following figure, enables the more efficient value-added integration of a number of different application components, by positioning them behind a service-oriented facade and by applying Web services technology.

²⁵ M. P. Papazoglou, W.-J. van den Heuvel, Service oriented architectures: approaches, technologies and research issues, The VLDB Journal (2007) 16:389–415

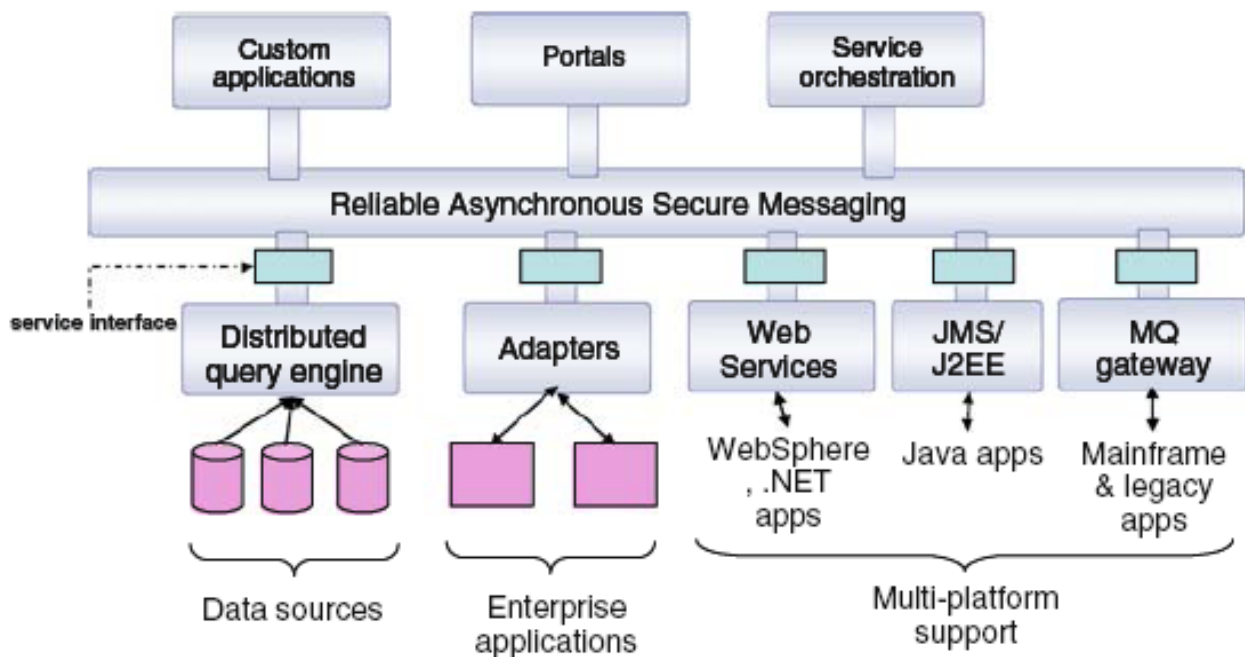


Figure 4: Enterprise Service Bus Architecture (Connecting Different Applications and Technologies)

Although most of the capabilities of a fully-functioning ESB still constitute major research issues, in the following section we elaborate on them in order to assist potential users in evaluating among the different ESB-“like” products and solutions that are (or will be) available in the market.

The following ESB capabilities list to be necessary to support the functions of a useful and meaningful ESB.

Leveraging existing assets Legacy applications are technically obsolete mission critical elements of an organization’s infrastructure – as they form the core of larger enterprise’s business processes – but are too frail to modify and too important to discard and thus must be reused. Strategically, the objective is to build a new architecture that will yield all the value hoped for, but tactically, legacy assets must be leveraged and integrated with modern technologies and applications.

Service communication capabilities A critical ability of the ESB is to route service interactions through a variety of protocols, and to transform from one protocol to another where necessary. Another important aspect of an ESB implementation is the capacity to support service messaging models consistent with the SOA interfaces, as well as the ability of transmitting the required interaction context, such as security, transaction, or message correlation information.

Dynamic connectivity capabilities Dynamic connectivity pertains to the ability to connect to Web services dynamically without using a separate static API or proxy for each service. Most enterprise applications today operate on a static connectivity mode, requiring some static piece of code for each service. Dynamic service connectivity is the key capability for a successful ESB implementation. The dynamic connectivity API is the same regardless of the service implementation protocol (Web services, JMS, EJB/RMI, etc.).

Topic/content-based routing capabilities The ESB should be equipped with routing mechanisms to facilitate not only topic-based routing but also, more sophisticated, content-based routing. Topic-based routing assumes that messages can be grouped into fixed, topical classes, so that subscribers can explicate interest in a topic and as a consequence receive messages associated with that topic. Content based routing on the other hand allows subscriptions on constraints of actual properties (attributes) of business events. The content of the message thus determines their routing to different endpoints in the ESB infrastructure. For example, if a manufacturer provides a wide variety of products to its customers, only some of which are made in-house, depending on the product ordered it might be necessary to direct the message to an external supplier, or route it to be processed by an internal warehouse fulfillment service. In a typical application, a message is routed by opening it up and applying a set of rules to its content to determine the parties interested in its content. Content-based routing is often dependant on the message constructed in XML, and will usually be built on top of Message Oriented Middleware, or JMS based messaging. Such ESB capabilities could be based on emerging standard efforts such as WS-Notification. WS-Notification defines a general, topic-based Web service system for publish and subscribe interactions, which relies on the WS-Resource framework . WS-Notification is a family of related specifications that define a standard Web services approach to notification using a topic-based publish/ subscribe pattern. The WS-Notification specification defines standard message exchanges to be implemented by service providers that wish to participate in notifications and standard message exchanges—allowing publication of messages from entities that are not themselves service providers. It also allows expressing operational requirements expected of service providers and requesters that participate in notifications. WS-Notification allows notification messages to be attached to WSDL Port-Types. The current WS-Notification specification provides support for both brokered as well as peer to-peer publish/subscribe.

Endpoint discovery with multiple quality of service capabilities The ESB should support the fundamental SOA need to discover, locate, and bind to services. Increasingly these capabilities will be based around Web services standards such as WSDL, SOAP, UDDI, and WS-PolicyFramework. As many network endpoints can implement the same service contract, the ESB should support service interactions with different values to the business. Several scenarios make it desirable for the client to select the best endpoint at run-time, rather than hardcoding endpoints at build time. The ESB should therefore be capable of supporting various qualities of service. Clients can query a Web service, such as an organizational UDDI service, to discover the best instance with which to interact based on QoS properties. Ideally, these capabilities should be controlled by declarative policies associated with the services involved using a policy standard such as the WS-PolicyFramework.

Integration capabilities. To support SOA in a heterogeneous environment, the ESB needs to integrate with a variety of systems that do not directly support service-style interactions. These may include legacy systems, packaged applications, COTS components, etc. When assessing the integration requirements for ESB, several types or “styles” of integration must be considered.

Transformation capabilities The various components hooked into the ESB have their own expectations of messaging models and data formats, and these might differ from other components. A major source of value in an ESB is that it shields any individual component from any knowledge of the implementation details of any other component. The ESB transformation services make it possible to ensure that messages and data received by any component is in the format it expects, thereby removing the burden to make changes. The ESB plays a major role in transformations between different, heterogenous data and messaging models, whether between legacy data formats (e.g., aCOBOL/VSAM application, running on an OS/390 host) and XML, between basic XML

formats, and Web services messages, or between different XML formats (e.g., transforming an industry standard XML message to a proprietary or custom XML format).

Reliable messaging capabilities Reliable messaging refers to the ability to queue service request messages and ensure guaranteed delivery of these messages to the destination. It also includes the ability to respond, if necessary, back to the requestor with response messages. Reliable messaging supports asynchronous store-and-forward delivery as well as guaranteed delivery capabilities. Primarily used for handling events, this capability is crucial for responding to clients in an asynchronous manner, and for a successful ESB implementation.

Security capabilities Generically handling and enforcing security is a key success factor for ESB implementations. The ESB needs to both provide a security model to service consumers and integrate with the (potentially varied) security models of service providers. Both point-to-point (e.g., SSL encryption) and end-to-end security capabilities will be required. These end-to-end security capabilities include federated authentication, which intercepts service requests and adds the appropriate username and credentials; validation of each service request and authorization to make sure that the sender has the appropriate privilege to access the service; and, lastly, encryption/decryption of XML content at the element level for both message requests and responses. To address these intricate security requirements trust models, WS-Security and other security related standards have been developed.

Long running process and transaction capabilities. Service-orientation, as opposed to distributed object architectures such as .NET or J2EE, make it possible to more closely reflect real-world processes and relationships. It is believed that SOA represents a much more natural way to model and build software that solves real-world business processing needs. Accordingly, the ESB should provide the ability to support business processes and long running services – services that tend to run for long duration, exchanging message (conversation) as they progress. Typical examples are an online reservation system, which interacts with the user as well as various service providers (airline ticketing, car reservation, hotel reservation, etc.). In addition, it is of vital importance that the ESB provides certain transactional guarantees. More specifically, the ESB needs to be able to provide a means for various applications to interact and message with each other and to recover should some form of technical or process failure occur. The challenge at hand is to ensure that complex transactions are handled in a highly reliable manner and if failure should occur, transactions should be capable of rolling back processing to the original, pre-request state.

Management and monitoring capabilities In an SOA environment, applications cross system (and even organizational) boundaries, they overlap, and they can change over time. Managing these applications is a serious challenge. Examples include dynamic load balancing, fail-over when primary systems go down, and achieving topological or geographic affinity between the client and the service instance, and so on. Effective systems and application management in an ESB require a management framework that is consistent across an increasingly heterogeneous set of participating component systems, while supporting complex aggregate (cross-component) management use cases, like dynamic resource provisioning and demand-based routing, service-level agreement enforcement in conjunction with policy based behavior. The latter implies the ability to select service providers dynamically based on the quality of service they offer compared to the business value of individual transactions. An additional requirement for a successful ESB implementation is the ability to monitor the health, capacity, and performance of services. Monitoring is the ability to track service activities that take place via the bus and accommodate visibility into various metrics and statistics. Of particular significance is the ability to be able to spot problems and exceptions in

the business processes and move toward resolving them as soon as they occur. Process monitoring capabilities are currently provided by toolsets in platforms for developing, deploying and managing service applications, such as, for instance, Web-Logic Workshop.

Scalability capabilities With a widely distributed SOA, there will be the need to scale some of the services or the entire infrastructure to meet integration demands. For example, transformation services are typically very resource intensive and may require multiple instances across two or more computing nodes. At the same time, it is necessary to create an infrastructure that can support the large nodes present in a global service network. The loose coupled nature of an SOA requires that the ESB uses a decentralized model to provide a cost effective solution that promotes flexibility in scaling any aspect of the integration network. A decentralized architecture enables independent scalability of individual services as well as the communications infrastructure itself.

3.7 Mule²⁶

Mule is a lightweight Java-based messaging framework that allows quick and easy applications connection and enables them to exchange data. Mule uses a service-oriented architecture (SOA), enabling easy integration of existing systems. Regardless of the different technologies the applications use, including JMS, Web Services, JDBC, HTTP, and more, Mule seamlessly handles interactions among them all.

Mule is based on ideas from Enterprise Service Bus (ESB) architectures. The key advantage of an ESB is that it allows different applications to communicate with each other by acting as a transit system for carrying data between applications within intranet or across the Internet. Mule is also vendor-neutral, so different vendor implementations can plug in to it.

Mule provides many advantages over competitors, including:

- Mule components can be of any type. It can integrate anything from a "plain old Java object" (POJO) to a component from another framework.
- Mule and the ESB model enable significant component reuse. Unlike other frameworks, Mule allows existing components use without any changes. Components do not require any Mule-specific code to run in Mule, and there is no programmatic API required. The business logic is kept completely separate from the messaging logic.
- Messages can be in any format from SOAP to binary image files. Mule does not force any design constraints on the architect, such as XML messaging or WSDL service contracts.
- Mule can be deployed in a variety of topologies, not just ESB. Because it is lightweight and embeddable, Mule can dramatically decrease time to market and increases productivity for projects to provide secure, scalable applications that are adaptive to change and can scale up or down as needed.

²⁶ mule.mulesource.org

MuleSource also provides administration tools that allow deployments management(Mule HQ), monitor transactions that flow through the system (Mule Saturn), and infrastructure control (Mule Galaxy).

One difference between Mule and a traditional ESB is that Mule only converts data as needed. With a typical ESB, an adapter has to be created for every application connected to the bus and convert the application's data into a single common messaging format. The development of these adapters and the time required to process every message requires a lot of time and effort. Mule eliminates the need for a single message format. The information is sent on any communication channel, such as HTTP or JMS, and is translated only as needed along the way. Therefore, Mule increases performance and reduces development time over a traditional ESB.

The following points provide useful data in a step-by-step approach of Mule architecture.

About SOA: Mule is based on the concept of a *service-oriented architecture* (SOA). The SOA approach to development allows IT organizations to create applications by bringing together components of application functionality, or services.

Processing the Data: When a message is sent from an application, Mule picks it up, sends it to a service that processes it using some specific business logic, and then routes it to the correct application. Mule contains many individual parts that handle the processing and routing of the message. The key part of the service is the service component which executes business logic on messages.

Routing Messages Between Service Components: The service component contains business logic for processing the data in the message but no information about how to receive or send messages themselves. To ensure that the service component receives the right messages and routes them properly after processing, an inbound router and an outbound router should be specified when configuring Mule. Inbound routers specify which messages the service component will process. They can filter incoming messages, aggregate them, and resequence them before routing them to a service component. For example, if a service component subscribes to an RSS feed, the inbound router could filter which messages it receives from that feed. After a service component has processed a message, the outbound router specifies where to dispatch the message.

Separating Business Logic from Messaging: One of the many advantages of Mule is that it can handle messages that are sent via a variety of protocols. If the service component handles only business logic and works with the data, not the message itself, how does it know how to read the various formats in which the message might arrive? The answer is that service components don't know how to read the messages, because by default, service components are completely shielded from the message format. Instead, a transport carries the message along, and transformers change the message's payload as needed to a format the service component can read before the router passes the message to the service component. For example, if an XML invoice is sent over HTTP, the HTTP transport carries the message along, routers direct the message to each service component that needs to process it, and transformers change the invoice along the way (such as from XML to a Java object) as required by each service component. All the transporting, transforming, and routing of the message are completely transparent to the service component.

Wiring Everything Together: Endpoints are configuration elements that are the key to wiring together all the services. They are specified in the inbound and outbound routers to tell Mule which

transport to use, where to send messages, and which messages a service component should receive. The primary part of an endpoint is the address, expressed as a uniform resource indicator (URI), which indicates the transport to use, the location (a transport-specific resource), and any additional parameters.

For example, if a service's inbound router specifies the endpoint `http://myfirm.com/mule`, the HTTP transport will dispatch to that service any messages that have been sent to that URL. If the inbound router specifies `file://myserver/files/`, the File transport, which is watching that directory, dispatches any new files created in that directory to the service. The endpoint specified on the outbound router indicates where the message will go next--it goes to the service with the same inbound endpoint as the previous component's outbound endpoint.

3.8 Apache ServiceMix²⁷

Apache ServiceMix is an enterprise class open source distributed enterprise service bus (ESB) and service-oriented architecture (SOA) toolkit. It was built from the ground up on the semantics and APIs of the Java Business Integration (JBI) specification JSR 208 and released under the Apache License. ServiceMix is lightweight and easily embeddable, has integrated Spring support and can be run at the edge of the network (inside a client or server), as a standalone ESB provider or as a service within another ESB. ServiceMix can be used in Java SE or a Java EE application server. ServiceMix uses ActiveMQ to provide remoting, clustering, reliability and distributed failover. In the following figure, ServiceMix integration with components and services is shown.

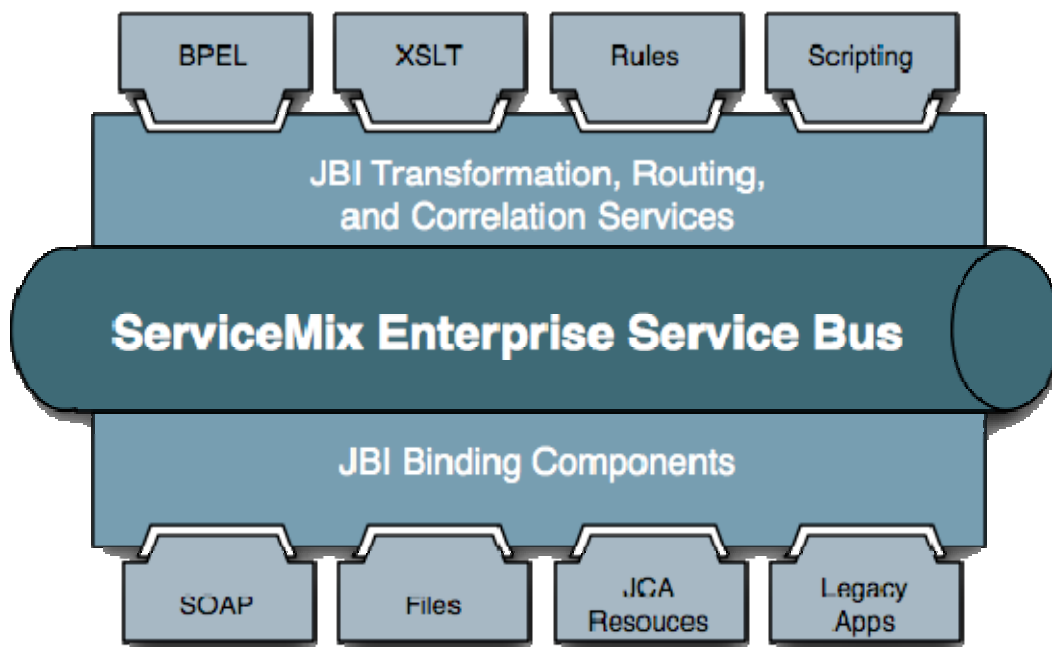


Figure 5: Apache ServiceMix integrating components and services

²⁷ <http://servicemix.apache.org>

ServiceMix is completely integrated into Apache Geronimo, which allows deploy JBI components deployment and services directly into Geronimo. ServiceMix is being JBI certified as part of the Geronimo project. Other J2EE application servers ServiceMix has been integrated with include JBoss, JOnAS with more to follow.

ServiceMix includes a complete JBI container supporting all parts of the JBI specification including:

- Normalized Message Service and Router
- JBI Management MBeans
- Ant Tasks for management and installation of components
- full support for the JBI deployment units with hot-deployment of JBI components

ServiceMix also provides a simple to use Client API for working with JBI components and services and in addition it provides an implementation of WS Notification.

3.9 FUSE ESB²⁸

FUSE ESB is an enterprise version of Apache ServiceMix that is tested, certified and supported. FUSE ESB is an open source enterprise service bus (ESB) that provides a standardized methodology, server, and tools to deploy integration components, freeing architects from the dependencies that have traditionally locked enterprises into proprietary middleware stacks. It is part of a family of open source SOA infrastructure tools that include FUSE Message Broker (enterprise release of Apache ActiveMQ), FUSE Services Framework (enterprise release of Apache CXF) and FUSE Mediation Router (enterprise release of Apache Camel).

Built from the ground up to support the JBI specification (JSR 208), FUSE ESB provides a structured environment to manage and deploy the components that developers create using FUSE Services Framework and FUSE Mediation Router, as well as additional JBI-compliant components like BPEL. FUSE ESB uses FUSE Message Broker as its underlying messaging infrastructure.

FUSE ESB benefits from the following advantages:

- *Ease of adoption and evaluation:* Organizations can download FUSE ESB immediately and evaluate it in their SOA environment, in any number of scenarios over any length of time.
- *Standards-based design:* FUSE ESB is designed to work with other integration components utilizing industry standards including JMS, JCA, JMX, enabling straightforward integration with existing middleware solutions, both commercial and open source.

²⁸ <http://open.iona.com/products/enterprise-servicemix/>

- *The right open source license:* FUSE ESB is available under an Apache-based license, Version 2.0, allowing organizations to incorporate it into any type of solution, without restrictions.
- *Cost-effective growth:* Over time FUSE ESB is deployed across geographies, business units and systems architectures, without incurring the cost of additional license fees.
- *Ready for the enterprise:* FUSE ESB is used and supported by IT organizations worldwide.
- *A Complete SOA Solution:* FUSE ESB provides the foundation for a comprehensive SOA runtime environment comprising all-open-source components. It's the fastest, most cost-effective way to build a complete foundation for SOA-based integration.

In the following figure, the internal, JBIbased FUSE ESB runtime environment is depicted. Local providers and protocol handlers are incorporated through standards-based interfaces, to act as Service Engines and Binding Components for messageprocessing. External services are connected via the Binding Components. The environment is deployed, managed and monitored through standard tools and protocols. FUSE ESB instances can be distributed, embedded with applications, clustered, and deployed to virtually any Java runtime environment.

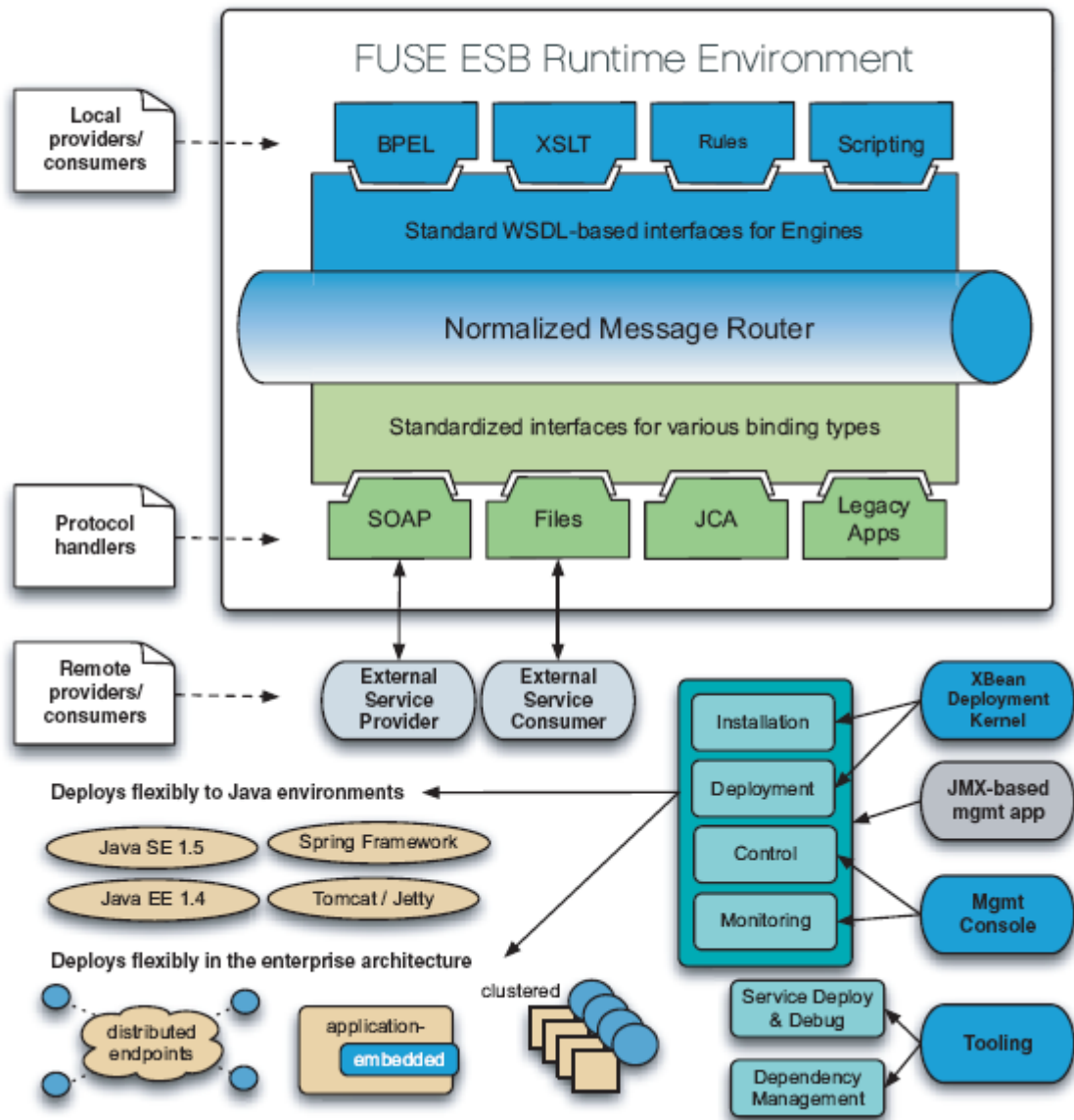


Figure 6: FUSE ESB Runtime Environment

4. Guidance to Governments

Accordingly to the previous section, the present provides reports on the current state of the art and state of practice in Interoperability, directed mainly towards a governmental audience, that is officials and IT experts of public administrations, central and regional government agencies and local municipalities.

It is widely accepted, both in literature and practice, that in establishing efficient eGovernment practices either in national or cross-country level one of the cornerstone success factors is the existence, acceptance and application of eGovernment Interoperability Frameworks. Therefore in the series of this Interoperability Guide eGovernment Interoperability Frameworks will be in the centre of our analysis. For the purpose of this version of the Interoperability Guide the focus is in the **Technical Dimension of the European Interoperability Framework**.

4.1 European Interoperability Frameworks (EIFs)

An Interoperability Framework describes the way in which organisations have agreed, or should agree, to interact with each other, and how standards should be used. In other words, it provides policies and guidelines that form the basis for selection of standards⁶. It may be contextualised (i.e., adapted) according to the socio-economic, political, cultural, linguistic, historical and geographical situation of its scope of applicability in a specific circumstance/situation (a constituency, a country, a set of countries, etc).

In an effort to promote the provision of Pan European eGovernment Services, EU published in 2004 the 1st version of the European Interoperability Framework – for which recently put on consultation a 2nd draft version.

4.2 European Interoperability Framework v1.0²⁹

Internet-based services, including government eServices are available in a myriad of forms and appearances and offer a variety of interaction types, ranging from simple websites to interactive ways of doing business. In the context of eGovernment services, a commonly used classification of these interaction types distinguishes the following sophistication levels:

- Stage 1: Online services only provide information. The consumer can read this information online or download it.
- Stage 2: Forms are available online. These can be downloaded and returned by post, fax or e-mail.
- Stage 3: Individual transactions between an administration and an enterprise or citizen are possible. Forms can be completed online and orders can be placed and paid for.

²⁹ <http://ec.europa.eu/idabc/en/document/3473>

- Stage 4: Multiple transactions are possible, services are integrated and transactions between administrations and enterprises and citizens are fully automated.

Although each of these levels describes eServices, the most challenging requirements for electronic interoperability are at the fourth level. Stage 1 and Stage 2 mainly concern the interaction of the eGovernment service with the user (front-office) where there is no automated electronic processing of the forms performed, whilst Stage 3 and especially Stage 4 involve background electronic processing of the information provided and possibly electronic interactions with external systems from other administrations and/or from enterprises (back-office interoperability). The main focus of Stage 1 / Stage 2 services is the provision of information to citizens and enterprises.

The most common way to delivering eServices to citizens is to set up a portal in front of the government applications, although mobile phones, PDAs etc. are also becoming increasingly important. The portal handles the communication with the users (user identification and authentication, presentation of a coherent view of the multitude of government services involved, provision/collection of data to/from the user, communication with the government applications, etc.). Additional portal components include forms servers and distributed content management systems. The communication between the portal and the applications, or between the application themselves, is then provided by specific middleware components which ensure the interoperability between the diverse systems. Some very good some middleware solutions can be found in in the cases of Sweden³⁰ and Germany³¹. In the context of pan-European eGovernment services, this means connecting applications which belong to different administrations and which are located in different Member States. The following figure considers the most complex interaction type (Stage 4) that encompasses the other models.

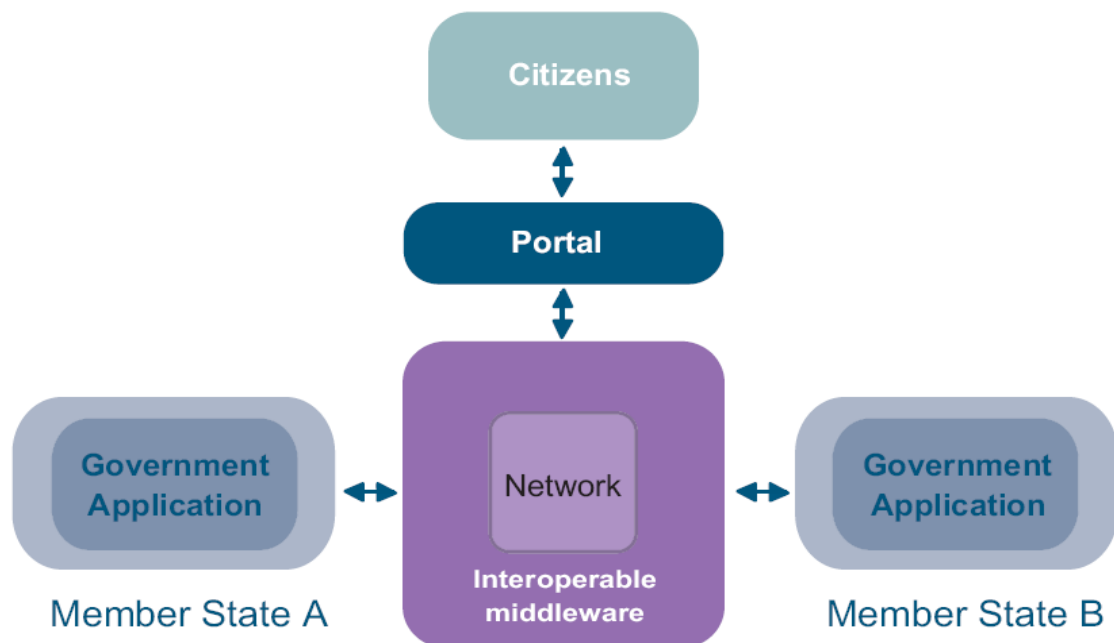


Figure 7: G2C and G2G Interactions through a Governmental eService Portal

³⁰ SHS: (<http://www.statskontoret.se/shs/pdf/1-1documentation.pdf>)

³¹ OSCI: (<http://www.osci.de/>)

Another way to enable communication between enterprises and public administrations is to directly interconnect their respective applications with adequate middleware components, as illustrated in the following figure.

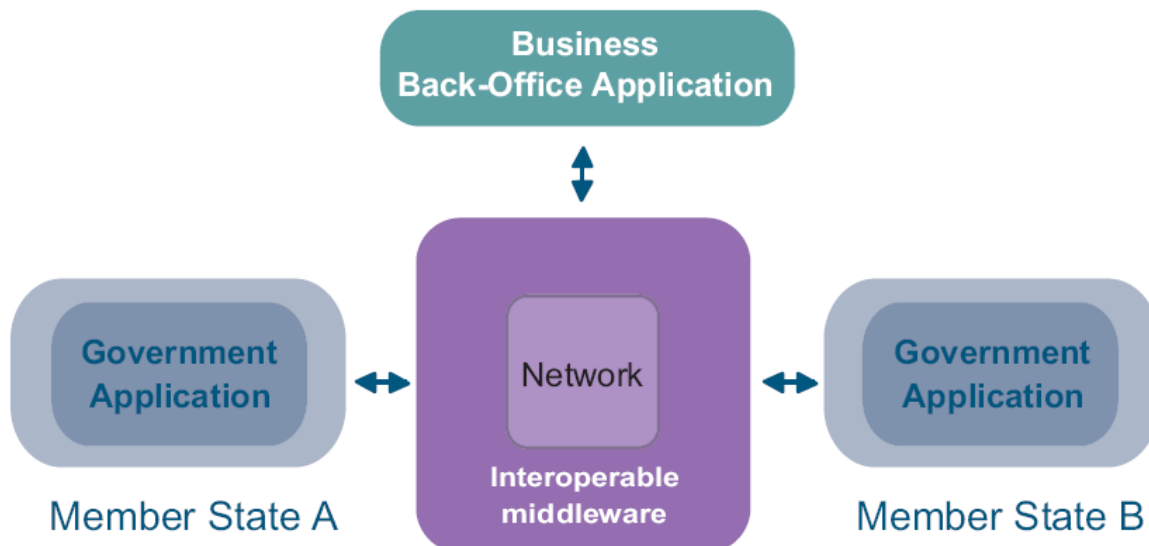


Figure 8: Interactions among Middleware Components

There is a commonality of standards for transport (e.g. networking LAN/WAN) and for presentation (e.g. file / hypertext / message transfer / character sets) of information. There is also a high degree of commonality in standards for domain naming, web browsers and viewers. This is because the national eGIFs, in effect, implement Internet standards at these levels. The use of the XML family of standards is recommended in national eGIFs for data integration. This is usually supplemented with recommendations for supporting standards such as UML or RDF for data modelling, XSLT for data transformation, Dublin Core (possibly with national extensions) for metadata, etc. Some Member States also make reference to the interoperability of Web Services.

These results provide for a very positive and favourable technical ground to the establishment of interoperable pan-European eGovernment services. The technical solutions adopted for such services will need to respect the capability of each partner concerned to organise their data processing systems and networks in the way that is best suited to their practices (i.e. technological approach, legal framework, principles of management, etc.). Technical interoperability should then be achieved on the basis of common guidelines that will enable the adoption of technical solutions that work on a multilateral basis.

Furthermore, multilingualism is a well-known characteristic of Europe and therefore a demanding aspect to be taken into account when designing technical solutions for pan-European eGovernment services.

4.3 European Interoperability Framework v2.0³²

At the moment the EIF v2.0 is in a draft version and under public consultation until the end of September. However some of its aspects concerning technical interoperability are presented in the present version of the Interoperability Guide in order to show the evolution of the framework since its first 2004 version.

According to EIF v2.0 technical interoperability covers the technical aspects of linking computer systems and services. It includes key aspects such as open interfaces, interconnection services, data integration and middleware, data presentation and exchange, accessibility and security services. In EIF v2.0 the following areas of technical nature have been identified as targets where standardization is needed in order to provide efficient Pan-European eGovernment Services (PEGS).

Presentation. Interoperability at the presentation layer in the EU context is basically concerned with issues such as accessibility, multilingualism and language neutrality. The question of cultural neutrality (via the use of language-independent symbols for such items as common operations, universal services, types of information, etc.) as an objective has also been raised, but should be the focus of further study.

Data Representation/Encoding. Data interoperability concerns the selection of standards for data formats, such as character sets, etc. For example, in the case where textual data originating in one IT system must be understandable by another IT systems, this implies the selection and use of the appropriate³³ multi-language supporting character set for the representation of such data to ensure that meaning and usability of the data is preserved.

Middleware. In the case where loosely coupled IT systems (or some subset of components and/or functionality therein) must collaborate, the question of middleware may arise as one possible solution. In the cross-border context, the requirement can often be expressed as the need to provide certain services to IT systems in other Member States, implying the exposure of certain interfaces of particular National IT systems data and/or functionality to a closed community of systems comprised of all the member states. A standardized approach to selection and use of middleware at community level proving this type of loose integration is recommended and will help to avoid the occurrence of new ad-hoc, bi-lateral and/or non-reusable solutions.

Platforms. The selection of platforms is generally of great interest to individual EU Member States administrations. Their national or departmental IT strategies will properly be concerned with issues such as portability as well as the maintenance implications of their selections. Interoperability aspects of such platforms are addressed in relevant standards and technical specifications such as POSIX, or IETF-produced specifications

Database and Data Models. Also largely a national matter, excepting that efficient, effective interoperability will depend to some degree on across-the-board use of the relational model for modelling data. There are hardly any systems being planned or developed (or even on the horizon) in EU Member States that do not conform to this model. This aspect of technical interoperability is closely related to semantic interoperability, as there may be sector-specific formats for exchange of

³² Revision of the EIF and AG, <http://ec.europa.eu/idabc/en/document/7728>

such structured data sets (such as ebXML) that depend on the selection of specific lower level standards (such as XML).

Networks. The ability to transmit and receive data reliably and intelligibly is fundamental to interoperability. The EU Member States will need to agree on common standards in this area.

Programming Languages. The portability of source code has an interoperability aspect. There exist relevant technical specifications governing such interoperability such as the ANSI specifications for the C Language, etc. or the Java Community Process defining the Java Programming language, and its various extensions.

Public administrations need to have a clear and accurate view of the technologies in use, the technical expertise and capabilities of their staff which are available for leveraging, and how IT supports their main business activities expressed as documented business processes.

Many legacy systems have been designed to be tightly coupled internally, providing for very little or no interaction with external IT systems. Direct access to the services provided by or the data managed by these systems is difficult and in some cases impossible. These so-called "silo", "islands" or "closed" systems constitute one of the key barriers to interoperability as there are significant difficulties involved in repurposing the technology and data as building blocks.

A key factor in judging the viability of technologies is the availability of professionals, inside and outside the public sector, with the necessary ICT expertise. Public Administrations need to assess the availability of professionals in the public/private sector and factor it into their strategic choices. Independently of the viability or utility of any specific technologies or standards, administrators need to cope with the existing set of standards and assets that are already in place / in use.

5. Interoperability Standards

The purpose of this section is to provide the readers with information about the results and activities in the standardisation domain in relation to interoperability. Under this scope, in the series of the Interoperability Guide material about existing standards in the various sub-domains of interoperability and reports on the activities of prominent standardisation organisations and initiatives that impact or will impact the interoperability domain will be presented.

For the purpose of this version of the Interoperability Guide the area of **Web Service Standards** is selected.

5.1 Web Service Standards

Standardisation in the Web Services domain constitutes quite an extended field comprising numerous existing standards as well as a number of already widely accepted efforts that are currently under standardisation procedure. The purpose of this section is not so to provide an in-depth analysis of all the existing Web Services Standards as to give the reads “a glimpse” on the landscape of technical standards related to web services. For this purpose instead of verbose descriptions a series of diagrammatic images have been drafted that group standards in families and outline their significance to interoperability. The interested reader can easily obtain more information about a particular standard, its specification, application, best practises and implementation guidelines, through a multitude of existing sources on the web and in literature.

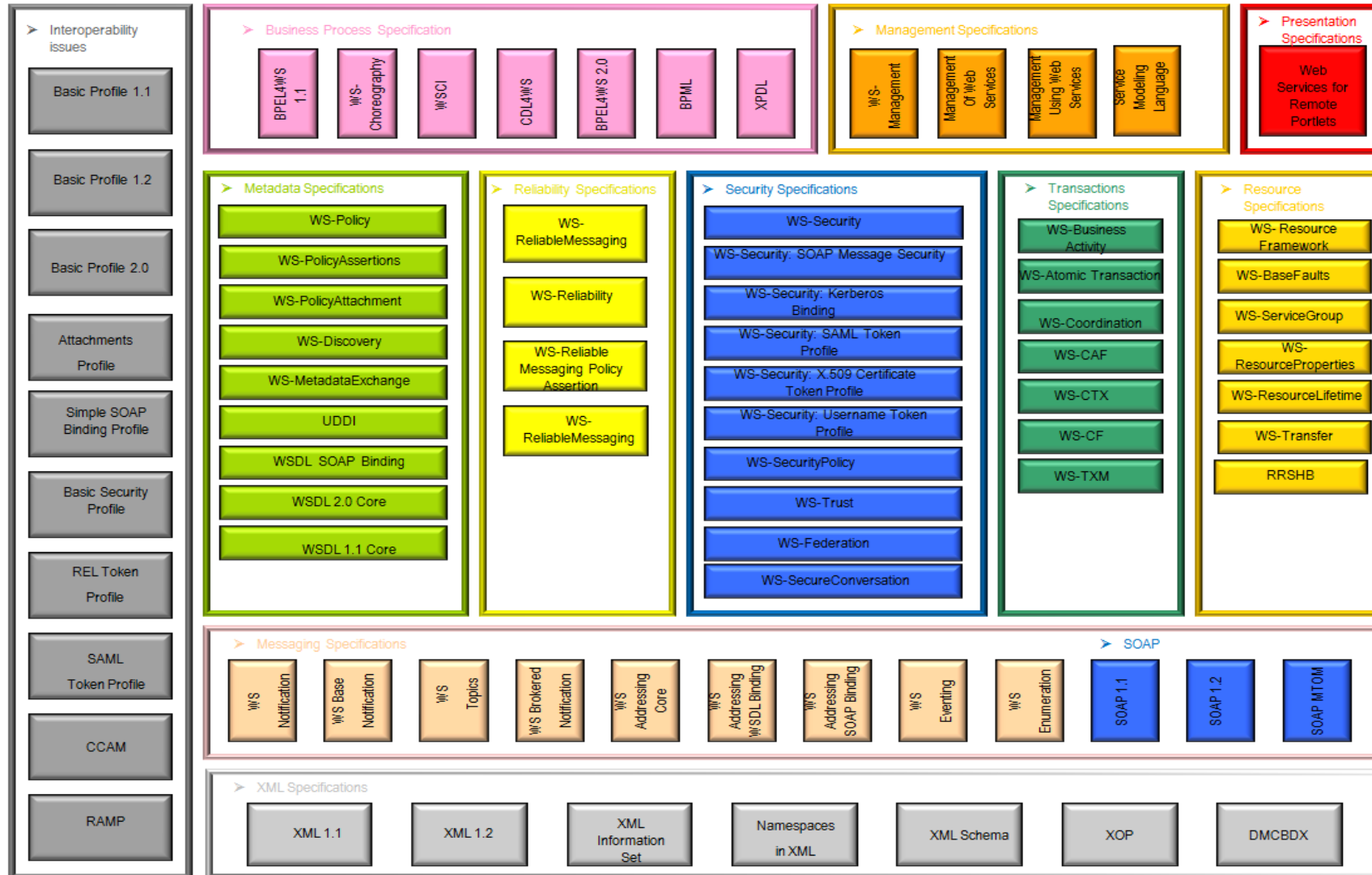


Figure 9: Web Services Standards Overview

- Business Process Execution Language for Web Services 1.1(BPEL4WS) provides a language for the formal specification of business processes and business interaction protocols using Web Services
- Web Service Choreography Description Language (CDL4WS) is to specify a declarative, XML based language that defines from a global viewpoint the common and complementary observable behaviour, where message exchanges occur, and when the jointly agreed ordering rules are satisfied.
- Web Service Choreography Interface (WSCI) describes how Web Service operations can be choreographed in the context of a message exchange in which the Web Service participates.
- WS-Choreography Model Overview defines the format and structure of the (SOAP) messages that are exchanged, and the sequence and conditions in which the messages are exchanged.
- Business Process Management Language (BPML) provides a meta-language for expressing business processes and supporting entities.
- Business Process Execution Language for Web Services 2.0 (BPEL4WS) provides a language for the formal specification of business processes and business interaction protocols using Web Services.
- XML Process Definition Language (XPDL) provides an XML file format that can be used to interchange process models between tools.

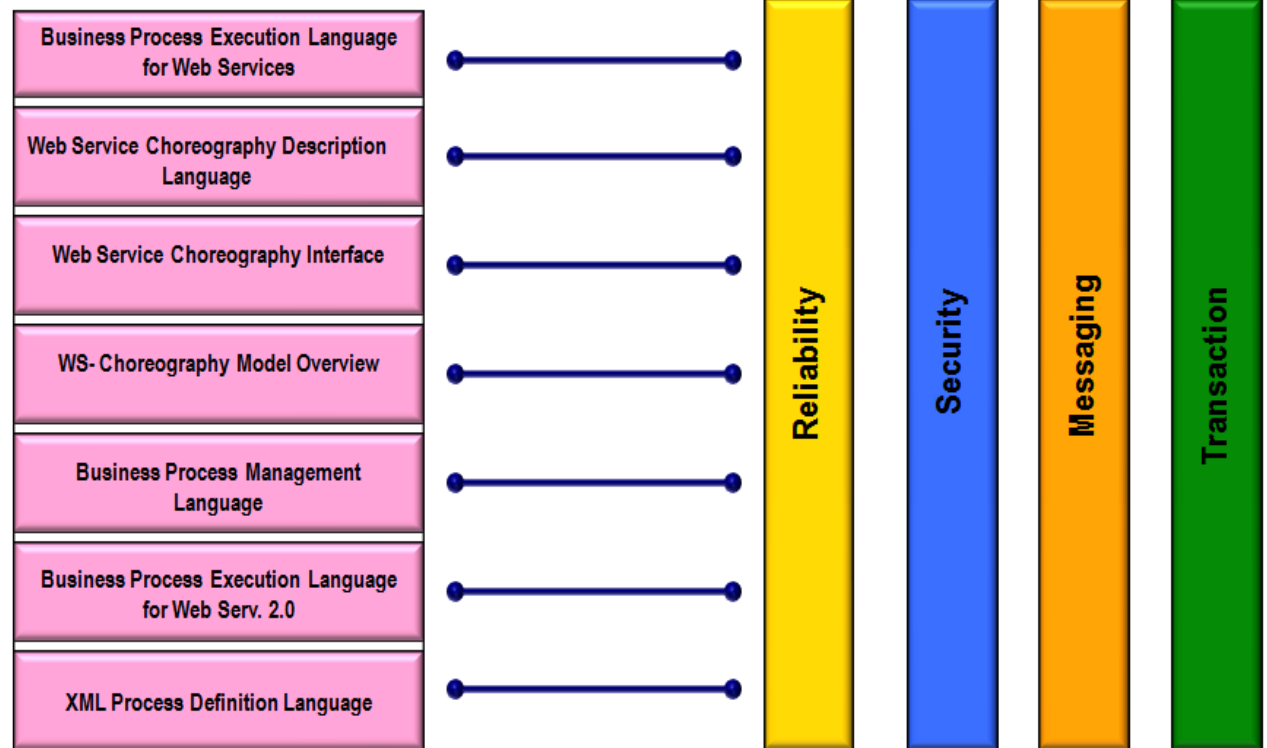


Figure 10: Dependencies of Business Process Specification Standards to other Categories of Web Service Standards

- WS-Policy describes the capabilities and constraints of the policies on intermediaries and endpoints (e.g. Business rules, required security tokens, supported encryption algorithms, privacy rules).
- WS-PolicyAssertions WS-PolicyAssertions provides an initial set of assertions to address some common needs of Web Services applications.
- WS-PolicyAttachment defines two general-purpose mechanisms for associating policies with the subjects to which they apply; the policies may be defined as part of existing metadata about the subject or the policies may be defined independently and associated through an external binding to the subject.
- WS-Discovery defines a multicast discovery protocol for dynamic discovery of services on ad-hoc and managed networks.
- WS-MetadataExchange enables a service to provide metadata to others through a Web services interface. Given only a reference to a Web service, a user can access a set of WSDL /SOAP operations to retrieve the metadata that describes the service.
- Universal Description, Discovery and Integration (UDDI) defines a set of services supporting the description and discovery of businesses, organizations, and other Web services providers, the Web services they make available, and the technical interfaces which may be used to access those services.
- Web Service Description Language 1.1, 2.0 Core is an XMLbased language for describing Web services and how to access them. It specifies the location of the service and the operations (or methods) the service exposes.
- Web Service Description Language SOAP Binding describes the concrete details for using WSDL 2.0 in conjunction with SOAP 1.1 protocol.

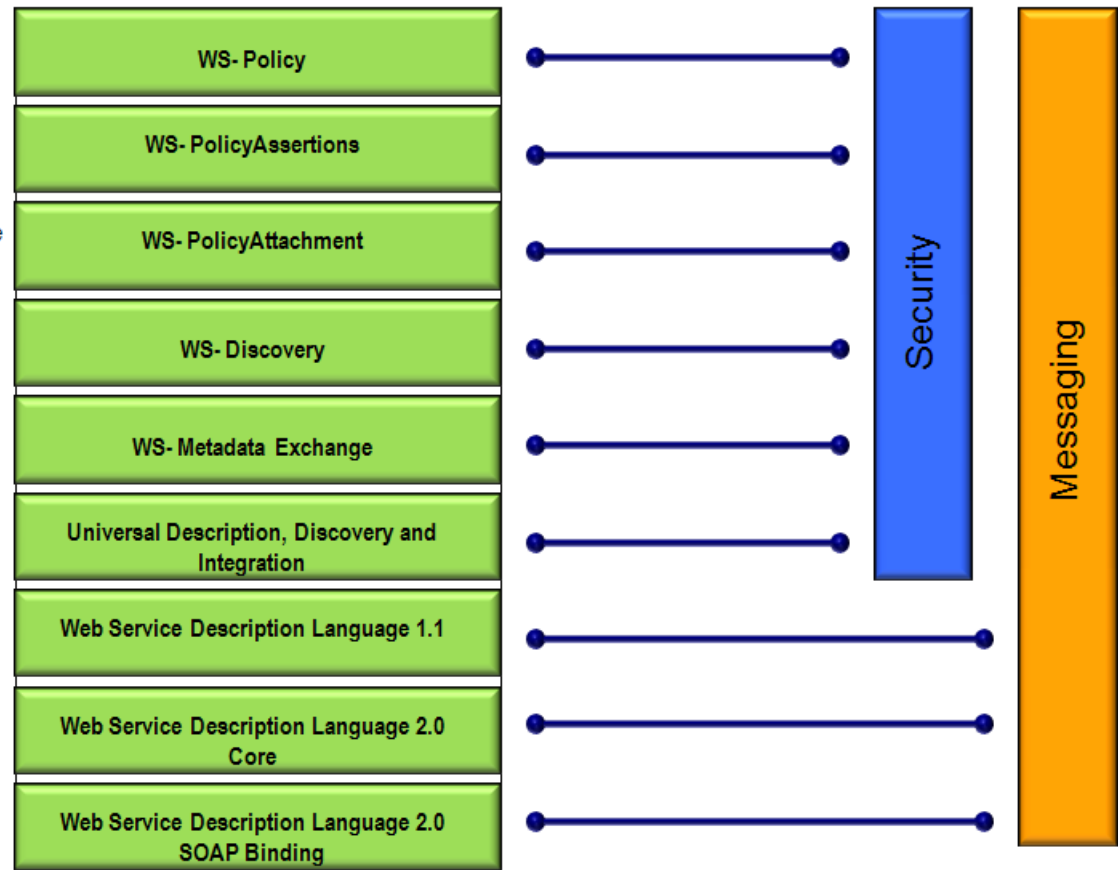


Figure 11: Dependencies of Metadata Specification Standards to other Categories of Web Service Standards

> WS-ReliableMessaging describes a protocol that allows Web services to communicate reliable in the presence of software component, system, or network failures. It defines a SOAP binding that is required for interoperability.

> WS-Reliability is a SOAP-based protocol for exchanging SOAP messages with guaranteed delivery, no duplicates, and guaranteed message ordering. WS-Reliability is defined as SOAP header extensions and is independent of the underlying protocol. This specification contains a binding to HTTP.

> Web Services ReliableMessaging Policy Assertion (WS-RM Policy) describes a domain-specific policy assertion for WS-ReliableMessaging that that can be specified within a policy alternative as defined in WS-Policy Framework.



Figure 12: Dependencies of Reliability Specification Standards to other Categories of Web Service Standards

- SOAP 1.1, 1.2: SOAP is a lightweight, XML-based protocol for exchange of information in a decentralized, distributed environment.
- SOAP Message Transmission Optimization Mechanism describes an abstract feature for optimizing the transmission and/or wire format of a SOAP message.
- WS-Notification is a family of related white papers and specifications that define a standard Web services approach to notification using a topicbased publish/subscribe pattern.
- WS-BaseNotification standardizes the terminology, concepts, operations, WSDL and XML needed to express the basic roles involved in Web services publish and subscribe for notification message exchange.
- WS-Topics defines three topic expression dialects that can be used as subscription expressions in subscribe request messages and other parts of the WS-Notification system.
- WS-BrokeredNotification defines the interface for the NotificationBroker. A NotificationBroker is an intermediary, which, among other things, allows publication of messages from entities that are not themselves service providers.
- WS-Addressing – Core provides transport-neutral mechanisms to address Web services and messages. This specification defines XML elements to identify Web service endpoints and to secure end-to-end endpoint identification in messages.
- WS-Addressing – WSDL Binding defines how the abstract properties defined in Web Services Addressing – Core are described using WSDL.
- WS-Addressing – SOAP Binding provides transportneutral mechanisms to address Web services and messages.
- WS-Eventing defines a baseline set of operations that allow Web services to provide asynchronous notifications to interested parties.
- WS-Enumeration describes a general SOAP-based protocol for enumerating a sequence of XML elements that is suitable for traversing logs, message queues, or other linear information models.

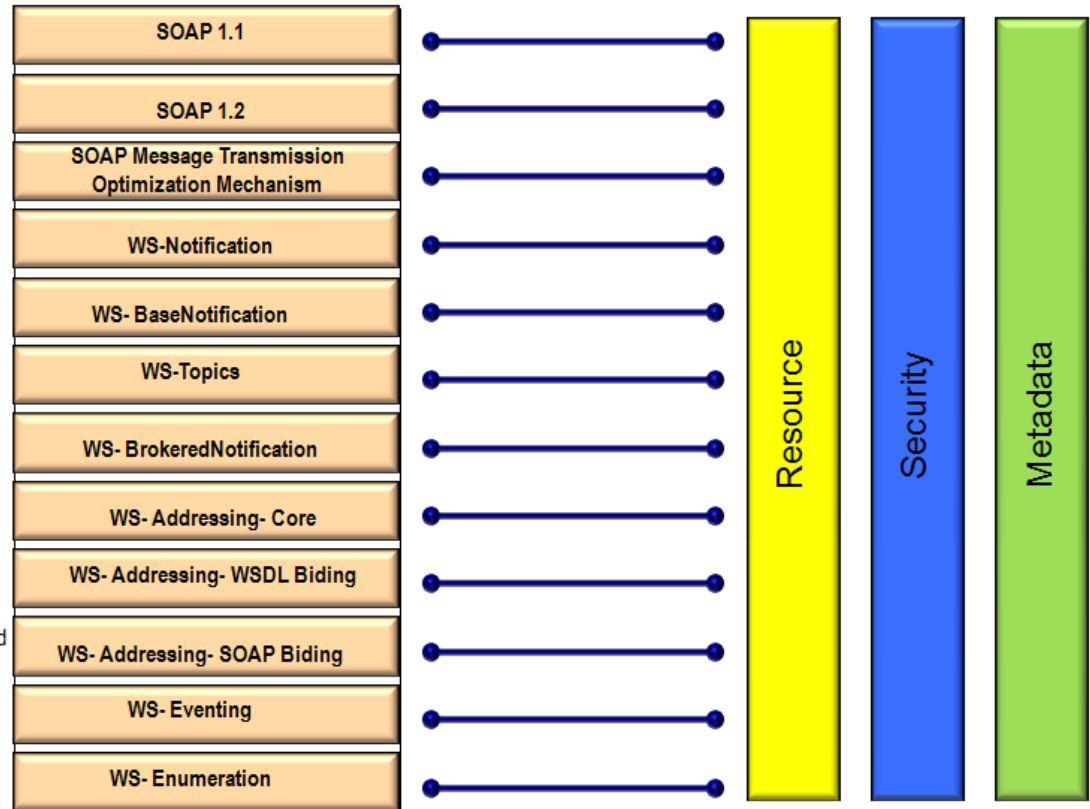


Figure 13: Dependencies of Messaging Specification Standards to other Categories of Web Service Standards

➤ WS-Management describes a general SOAP-based protocol for managing systems such as PCs, servers, devices, Web services and other applications, and other manageable entities.

➤ Web Service Distributed Management: Management Of Web Services (WSDM-MOWS) addresses management of the components that form the network, the Web services endpoints, using Web services protocols.

➤ Web Service Distributed Management: Management Using Web Services (WSDM-MUWS) defines how an IT resource connected to a network provides manageability interfaces such that the IT resource can be managed locally and from remote locations using Web services technologies.

➤ Service Modeling Language (SML) is used to model complex IT services and systems, including their structure, constraints, policies, and best practices.

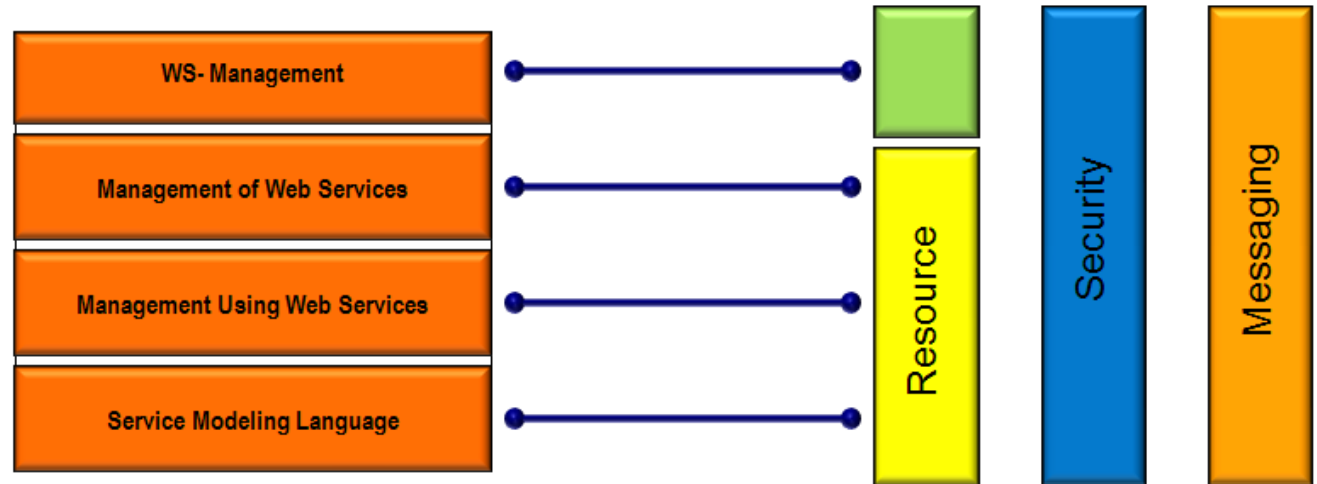


Figure 14: Dependencies of Management Specification Standards to other Categories of Web Service Standards

➤ Web Services for Remote Portlets (WSRP) defines a set of interfaces and related semantics which standardize interactions with components providing user-facing markup, including the processing of user interactions with that markup.

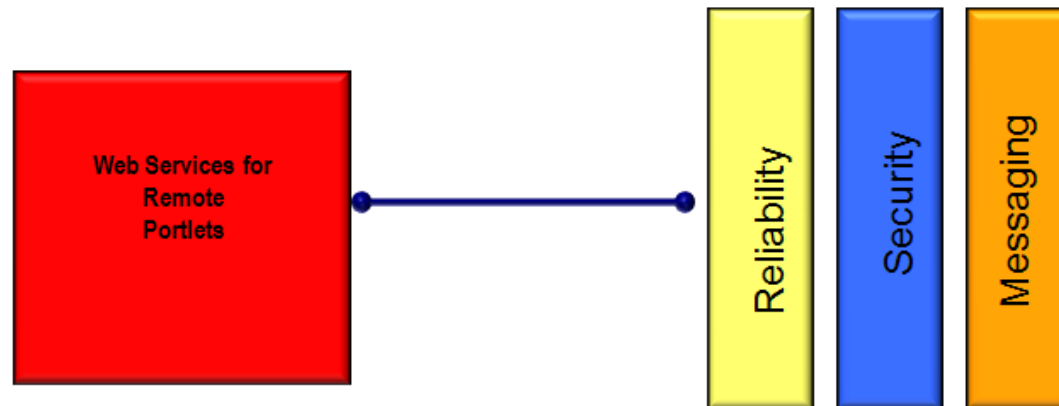


Figure 15: Dependencies of Presentation Specification Standards to other Categories of Web Service Standards

- WS-Security is a communications protocol providing a means for applying security to Web Services.
- WS-Security: SOAP Message Security describes enhancements to SOAP messaging to provide message integrity and confidentiality. Specifically, this specification provides support for multiple security token formats, trust domains, signature formats and encryption technologies. The token formats and semantics for using these are defined in the associated profile documents.
- WS-Security: Kerberos Binding defines how to encode Kerberos tickets and attach them to SOAP messages. As well, it specifies how to add signatures and encryption to the SOAP message, in accordance with WS-Security, which uses and references the Kerberos tokens.
- WS-Security: SAML Token Profile defines the use of Security Assertion Markup Language (SAML) v1.1 assertions in the context of WSS: SOAP Message Security including for the purpose of securing SOAP messages and SOAP message exchanges.
- WS-Security: X.509 Certificate Token Profile describes the use of the X.509 authentication framework with the WS-Security: SOAP Message Security specification.
- WS-Security: Username Token Profile describes how a Web Service consumer can supply a Username Token as a means of identifying the requestor by username, and optionally using a password (or shared secret, etc.) to authenticate that identity to the Web Service producer.
- WS-SecurityPolicy defines how to describe policies related to various features defined in the WS-Security specification.
- WS-Trust describes a framework for trust models that enables Web Services to securely interoperate. It uses WS-Security base mechanisms and defines additional primitives and extensions for security token exchange to enable the issuance and dissemination of credentials within different trust domains.
- WS-Federation describes how to manage and broker the trust relationships in a heterogeneous federated environment including support for federated identities.
- WS-SecureConversation specifies how to manage and authenticate message exchanges between parties including security context exchange and establishing and deriving session keys.



Figure 16: Dependencies of Security Specification Standards to other Categories of Web Service Standards

➤ WS-Business Activity provides the definition of the business activity coordination type that is to be used with the extensible coordination framework described in the WS-Coordination specification.

➤ WS-Atomic Transaction defines protocols that enable existing transaction processing systems to wrap their proprietary protocols and interoperate across different hardware and software vendors.

➤ WS-Coordination describes an extensible framework for providing protocols that coordinate the actions of distributed applications.

➤ WS-Composite Application Framework (WS-CAF) is a collection of three specifications aimed at solving problems that arise when multiple Web Services are used in combination. It proposes standard, interoperable mechanisms for managing shared context and ensuring business processes achieve predictable results and recovery from failure.

➤ WS-Transaction Management (WS-TXM) defines a core infrastructure service consisting of a Transaction Service for Web Services.

➤ WS-Context (WS-CTX) is intended as a lightweight mechanism for allowing multiple Web Services to share a common context.

➤ WS-Coordination Framework (WS-CF) allows the management and coordination in a Web Services interaction of a number of activities related to an overall application.

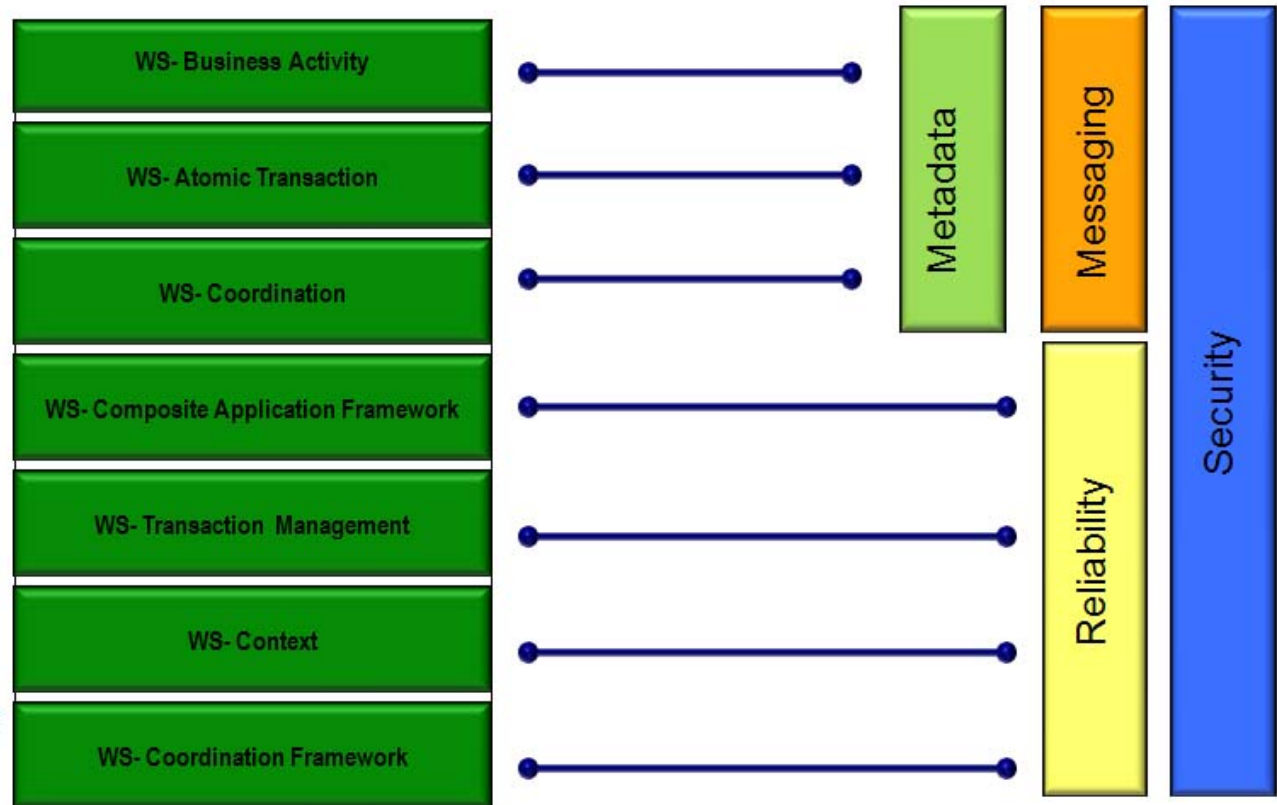


Figure 17: Dependencies of Transaction Specification Standards to other Categories of Web Service Standards

➤ Web Services Resource Framework (WSRF) defines a family of specifications for accessing stateful resources using Web Services.

➤ WS-BaseFaults (WSRF) defines a base set of information that may appear in fault messages. WS-BaseFaults defines an XML schema type for base faults, along with rules for how this base fault type is used and extended by Web Services.

➤ WS-ServiceGroup (WSRF) defines a means by which Web Services and WS-Resources can be aggregated or grouped together for a domain specific purpose.

➤ WS-ResourceProperties specifies the means by which the definition of the properties of a WS-Resource may be declared as part of the Web Service interface. The declaration of the WS-Resource properties represents a projection of or a view on the WS-Resource state.

➤ WS-ResourceLifetime is to standardize the terminology, concepts, message exchanges, WSDL and XML needed to monitor the lifetime of, and destroy WS-Resources. Additionally, it defines resource properties that may be used to inspect and monitor the lifetime of a WS-Resource.

➤ WS-Transfer describes a general SOAP-based protocol for accessing XML representations of Web service-based resources.

➤ Resource Representation SOAP Header Block (RRSHB) complements MTOM by defining mechanisms for describing and conveying non-XML resource representations in a SOAP 1.2 message.

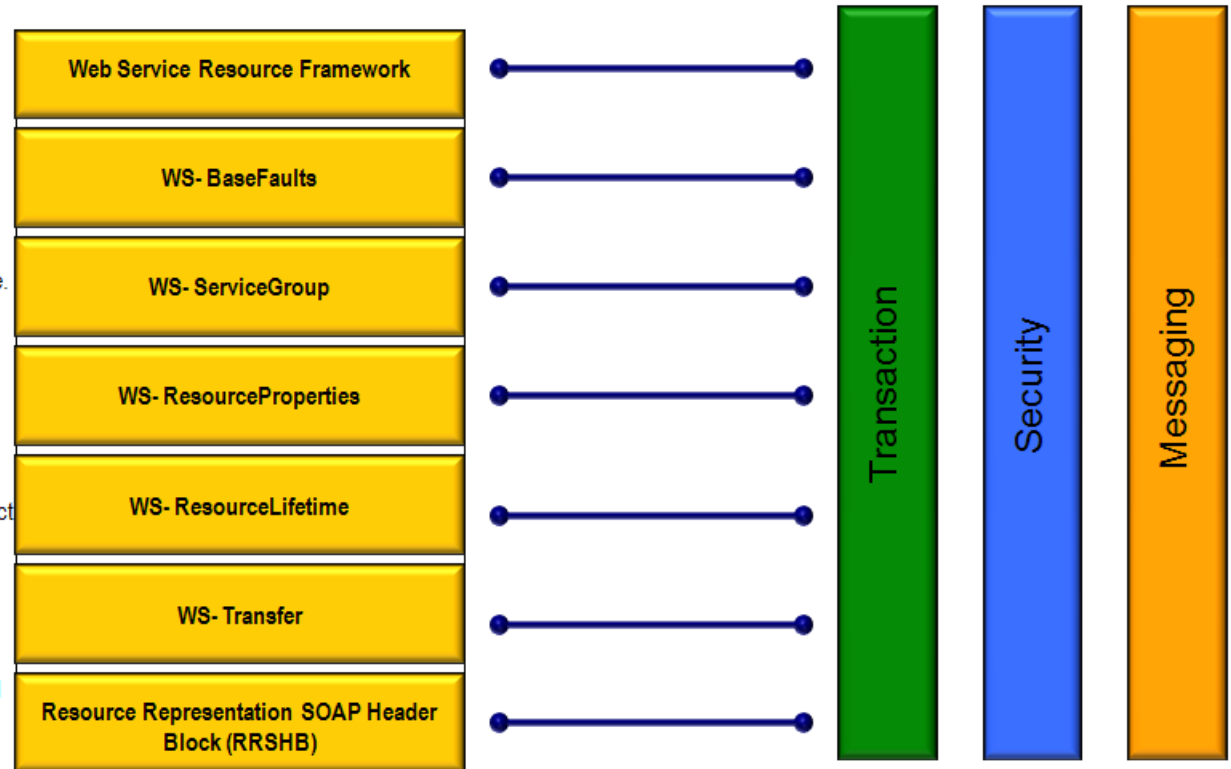


Figure 18: Dependencies of Resource Specification Standards to other Categories of Web Service Standards

6. Recommendations

The purpose of this section is to concretise the conclusions that can be drawn based on the material provided in chapters 3, 4 and 5 and give specific and straightforward guidelines that can be used by decision makers in enterprises and organisations and policy makers of central regional and local governments.

6.1 For Decision Makers

For the decision makers in enterprises and organisations this version of the Interoperability Guide outlines a series of IT solutions that can be deployed in achieving interoperability among their underlying ICT infrastructures.

Decision Makers should bear in mind:

- Today a number of commercial integrated message oriented middleware (MOM) platforms exist that can be readily deployed to support the interoperability needs of enterprises and organisations. The term “integrated” signifies that these platforms incorporate a number of capabilities such as business process integration, data integration, service monitoring and management, security, etc in order to provide a holistic interoperability solution to the adopting organisation or business.
- The existing message oriented middleware (MOM) platforms can cover the interoperability needs of a vast category of enterprises spanning almost the entire spectrum of small and micro enterprises to large industries. Before opting to acquire such a tool enterprises should pinpoint their particular interoperability needs (e.g. process integration, data exchange, or other) in relation to their size and not necessarily turn to such an integrated solution but seek a tool more focused to their particular need in order to get better value for their money.
- Adopt a sceptical attitude before proceeding with solutions that proclaim to be fully functional and efficient ESBs. Such an ESB should present a multitude of features a lot of which are still under research, therefore decision makers should check whether the envisaged ESB-“like” tool incorporates the necessary features to support their needs.
- Open source solutions are constantly gaining momentum even when it comes to elaborate ICT infrastructures to be used for heavy commercial use such as the ESB architecture. Decision Makers should seriously concern open source alternatives, both in readily available products and technologies, before choosing or implementing an interoperability solution.
- Web Services constitute today a mature technology for implementing interoperability solutions that already has a number of widely acceptable and established standards. Decision Makers that opt among different technologies for implementing interoperability solutions in the scope of their business or organisation should seriously leverage the potential of adopting web services as a candidate technology.

6.2 For Policy Makers

For the policy maker (and also IT officers) of central, regional and local government this version of the Interoperability Guide presented the technical dimension of the two versions of the European Interoperability Framework – for the 2nd version a more detailed description of its various aspects will be included in the following versions of the guide.

Policy Makers should give attention to:

- At back-office and front-office levels technical interoperability aspects should be considered for a number of areas – e.g. data presentation and exchange, accessibility -interface design principles, multi-channel access, file type and document formats, etc for front office level, data integration and middleware, XML-based standards, web services, etc for back-office.
- Security always constitutes an aspect that has to be considered individually and in relation with specific standards, best practices and international guidelines.
- For the Pan-European services provided via portals, or other ICT infrastructures of administrations and governmental agencies multilingualism is one of the vital issues that has to be addresses. Towards this direction, solutions such as providing full multilingualism up to a certain level in governmental portals or the deployment of specific tools that can provide rough translations of the provided material to interested user can be applied. Additionally, ICT infrastructures (such as commercial portal building platforms) that provide native multilingual capabilities should also be preferred.
- Member State administrations and EU agencies and organisations should develop (or agree on) and use common guidelines for the technical interoperability of pan-European networks, applications and services in the context of eGovernment. These common guidelines should be based on recognized open standards. For this purpose web services today constitute a technology that already has a big number of widely acceptable and established standards.
- Public Administrations need to assess the availability of professionals in the public/private sector and factor it into their strategic choices before designing and implementing any specific eGovernment initiative.

References

Article 3b of the Decision 2004/387/EC of the European Parliament and of the Council on 21st of April 2004
<http://ec.europa.eu/idabc/en/document/3473>

eBusiness Roadmap: addressing key eBusiness standards issues 2006-2008,
<http://www.cen.eu/cenorm/businessdomains/businessdomains/iss/activity/ebusfinal.pdf>

eGovRTD2020 Project,
http://www.egovrtd2020.org/EGOVRTD2020/navigation/work_packages/wp4_roadmapping/D41

European Commission, 2003, 'The role of eGovernment for Europe's future' Communication from the Commission to the Council, the European Parliament, the European Economic and Social Committee and the Committee of the Regions, Brussels, 26.9.2003, COM(2003) 567 Final.

European Commission, Communication on a European eHealth Area Com (2004) 356

European Commission Enterprise Interoperability Research Roadmap, http://cordis.europa.eu/ist/ict-ent-net/ei-roadmap_en.htm

European Interoperability Framework, <http://ec.europa.eu/idabc/servlets/Doc?id=19528>

Federal Standard 1037C, entitled Telecommunications: Glossary of Telecommunication Terms

Greek Digital Strategy 2006-2013, <http://www.infosoc.gr/infosoc/en-UK/sthnellada/committee/default1/top.htm>

<http://ec.europa.eu/idabc/en/document/3473>

http://en.wikipedia.org/wiki/Enterprise_Service_Bus

http://en.wikipedia.org/wiki/Message_Oriented_Middleware

<http://en.wikipedia.org/wiki/Middleware>

<http://open.iona.com/products/enterprise-servicemix/>

<http://mule.mulesource.org>

<http://servicemix.apache.org>

<http://www.genesis-ist.eu>, Deliverable D5.1 State of the Art Analysis

IDABC, <http://ec.europa.eu/idabc/>

IDABC 2004. European Interoperability Framework for pan-European eGovernment Services. Luxembourg, European Communities.

Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY: 1990.

ISO/IEC 2382-01, Information Technology Vocabulary, Fundamental Terms

M. P. Papazoglou, W.-J. van den Heuvel, Service oriented architectures: approaches, technologies and research issues, The VLDB Journal (2007) 16:389–415

OSCI: (<http://www.osci.de/>)

Revision of the EIF and AG, <http://ec.europa.eu/idabc/en/document/7728>

SHS: (<http://www.statskontoret.se/shs/pdf/1-1documentation.pdf>)

The i2010 Strategy Framework, http://ec.europa.eu/information_society/eeurope/i2010/index_en.htm

Appendix- Interoperability Knowledge Map

The collaboration with related projects from the aspect of knowledge exchange has provided the ability to distinguish the specific research fields in the Interoperability domain and provide a further mapping based on three main categories, namely, the Horizontal Issues, The Vertical Issues and the Country-Related Issues

Horizontal Issues

The Horizontal Issues mapping refers to the research fields of Interoperability based on the concepts that are common between all industries, and include aspects of Organisational, Semantic and Technical Interoperability, Interoperability Standardisation, Assessment and Training, Legal Issues on Interoperability, Interoperability Business Models and Interoperability Policies. These are presented in the following Table:

1.	Organisational Interoperability
1.1.	Enterprise Modelling
1.2.	Business Process Modelling
1.3.	Business Process Reengineering
1.4.	Business Process Management
1.5.	Organisational Alignment
2.	Semantic Interoperability
2.1.	Ontology Engineering
2.2.	Knowledge Management
2.3.	Semantic Annotation
2.4.	Data Modelling
2.5.	Core Components Modelling
2.6.	XML Schema Creation
3.	Technical Interoperability
3.1.	Information Storage
3.2.	Information Communication

3.3.	Information Presentation
3.4.	Web Services
4.	Interoperability Standardisation
4.1.	ISO Standards
4.2.	UN Standards
4.3.	CEN Standards
4.4.	OASIS Standards
4.5.	EC Interoperability Frameworks
4.6.	National Interoperability Frameworks
5.	Interoperability Assessment
5.1.	Interoperability Assessment Models
5.2.	Interoperability Impact Assessment Models
5.3.	Interoperability Metrics
6.	Interoperability Training
6.1.	Curricula development
6.2.	Lessons development
6.3.	Assessment methods
7.	Legal Issues on Interoperability
8.	Interoperability Business Models
9.	Interoperability Policies
9.1.	United Nations
9.2.	European Union
9.3.	National Policies

Table 1: Horizontal Issues of the I-KMap

Vertical Issues

The vertical Issues mapping refers to the Industry-Specific research fields on Interoperability. These include issues on eGovernment, eBusiness, eHealth, eLearning and eParticipation. These are presented in the following Table:

1.	eGovernment
1.1.	Central Government – specific issues
1.2.	Local Government – specific issues
1.3.	Government Service Portals
1.4.	Government Back-office Systems
2.	eBusiness
2.1.	Large Enterprises – Specific Issues
2.2.	SME – Specific Issues
2.3.	VSE – Specific Issues
2.4.	Business Portals
2.5.	Enterprise Software Applications
2.6.	Specific Industry Sectors
2.6.1.	Aerospace and Defence
2.6.2.	Biotechnology
2.6.3.	Bulk Products and Materials
2.6.4.	Chemicals
2.6.5.	Communications
2.6.6.	Construction
2.6.7.	Electronics
2.6.8.	Environment
2.6.9.	Financial Services
2.6.10.	Food and Drink

2.6.11.	Media
2.6.12.	Pharmaceuticals
2.6.13.	Power
2.6.14.	Retail
2.6.15.	Software
2.6.16.	Transport
2.6.17.	Water
3.	eHealth
4.	eLearning
5.	eParticipation

Table 2: Vertical Issues of the I-KMap

Country – Related Issues

The Country-Related Issues mapping refers to the research efforts made according to the country or region where they apply, with a categorisation based on trans-regional research initiatives and/or knowledge sharing practices. The main regions identified are The European Union, EU Candidate Countries, Other European Countries, The Americas, Middle East and Africa, Asia and Australia. This mapping is presented in the following Table:

1.	European Union
1.1.	Austria
1.2.	Belgium
1.3.	Bulgaria
1.4.	Cyprus
1.5.	Czech Republic
1.6.	Denmark
1.7.	Estonia
1.8.	Finland

1.9.	France
1.10.	Germany
1.11.	Greece
1.12.	Hungary
1.13.	Ireland
1.14.	Italy
1.15.	Latvia
1.16.	Lithuania
1.17.	Luxembourg
1.18.	Malta
1.19.	Netherlands
1.20.	Poland
1.21.	Portugal
1.22.	Romania
1.23.	Slovakia
1.24.	Slovenia
1.25.	Spain
1.26.	Sweden
1.27.	United Kingdom
2.	EU Candidate Countries
2.1.	Croatia
2.2.	Fyrom
2.3.	Turkey
3.	Other European Countries
4.	Americas

4.1.	Canada
4.2.	US
5.	Middle East and Africa
6.	Asia
7.	Australia

Table 3: Regional Dimensions of the I-KMap