FP6-2004-27020

Access-eGov

Access to e-Government Services Employing Semantic Technologies

Instrument: STREP

Thematic Priority:
SO 2.4.13 Strengthening the integration
of the ICT research effort in an enlarged Europe

D2.1 State-of-the-art Report

Start date of project: January 1, 2006 **Duration:** 36 months

Date of submission: May 31, 2006

Lead contractor for this deliverable: University of Regensburg

Revision: Final 1.6

Dissemination level: PU

Acknowledgement: The Project is funded by European Commission DG INFSO under the IST programme, contract No. FP6-2004-27020.

FP6-2004-27020 **Access-eGov**
Access to e-Government Services Employing Semantic Technologies

D2.1 State-of-the-art Report

Workpackage:	WP2	**Task:**	T2.3
Date of submission:		May 31, 2006	
Lead contractor for this deliverable:		University of Regensburg (UR)	
Authors:		Jan Kolter (UR) Stefan Duerbeck (UR) Marian Mach (TUKE) Peter Bednar (TUKE) Jan Hreno (TUKE) Marek Skokan (TUKE)	
Version:		1.6	
Revision		Final	
Dissemination level:		PU	

Project partners:

Technical University of Kosice (TUK), Slovakia (Coordinator); University of Regensburg (UR), Germany; German University in Cairo (GUC, Egypt; Intersoft, a.s. (IS), Slovakia; EMAX S.A. (EMA), Poland; Kosice Self-Governing Region (KSR), Slovakia; Cities on Internet Association (COI), Poland; e-ISOTIS (ISO), Greece; Municipality of Michalovce (MI), Kosice; City Hall of Gliwice (GLI), Poland; State Government of Schleswig-Holstein (SHG), Germany.

Abstract:

This document aims at evaluating the current state-of-the-art in modern information technologies that may be used for the Access e- Gov project. In this field, web services architectures and adjoining techniques have proven to be adequate for electronic communication in grown, heterogeneous environments. Key technologies for our approach should necessarily cover semantic service annotation, execution and orchestration of workflows.

The report is structured according to the following technological domains:

- State-of-the-art in e-Government
- Semantic Web Services formalisms
- Frameworks and tools for semantic Web Services

Content

List of Figures

1 Introduction

Access e-Gov is envisioned to become a distributed service platform, supporting citizens with a pro-active personal assistant that will help them find their way to those government services that will suit the customer's demand best.

This document is intended to provide the reader with a short overview of those state-of-the-art technologies and current e-Government applications that are of interest to the Access-eGov project.

State-of-the-art is considered to be the highest degree - a certainly predominant position - in a field of current development of an art or technique in our days.

Web Services are supposed to be the technology of choice when it comes to realizing cross-governmental, integrated services. They constitute the common technology to fulfill application-to-application interoperability, based on XML message exchange that is capable of dynamically invoking remote software components with a minimum effort in interface description and customizing.

The question, to which extent other existing technologies might be useful and necessary in order to discover and orchestrate these services, will be the subject of this document. Many possible approaches and methodologies already exist to meet the basic requirements of our project.

When evaluating suited technologies for Access e-Gov, some project-specific issues have to be taken into consideration: the services platform will operate on many different administrative levels in a transnational context. Therefore, semantic technologies will play a dominant role to overcome language barriers in terms of service description and annotation (see Chapter 3.2). In semantically-enriched systems, ontologies or controlled vocabularies are used as conceptual underpinnings for providing information about resources and for accessing them. Public servants shall thus be empowered by Access e-Gov technology to annotate their agencies' services on their own, being provided with intuitive software and straight-forward reference manuals.

It will thus be a task of future research to test the most promising approaches against the Access-eGov practical requirements. Thus, although a principal evaluation of related methods and technologies has been done in this report, more investigation will be required for final decision. The result should determine the adequate level of complexity of the required formalisms that will lead to a customer-centered, efficient, and easy-to-use solution for providing integrated public services to the citizen.

It is also noteworthy that this report is to be regarded as an in-depth supplement, and not a replacement, to the Technical Annex (DoW), which also contains a State-of-the-Art section. It will further outline specific aspects in worldwide e-Government in a more detailed way and evaluate those cutting-edge technologies that are able to constitute the technological basis for the Access e-Gov platform.

1.1 About the document structure

The state-of-the-art report is subdivided into the following three parts:

- Chapter 2 provides the reader with a tabular synopsis of important e-government implementations and will furthermore outline those that are relevant to Access e-Gov.
- Chapter 3 will describe a selection of semantic Web Services formalisms. It presents standards techniques just as OWL-S, WSMF and WSDL-S.
- Chapter 4 is evaluating the most promising existing frameworks and tools for semantic Web Services.

Each technology will be categorized according to a SWOT-analysis matrix, to outline the most significant features in a condensed way.

An additional scope of this document is to build up a common understanding of the project partners concerning the useful technologies, and to establish a reference frame.

The reader will be provided with a temporary conclusion at the end of each technology's description and in the summary part of this report. Based on these evaluations, the project group will decide which technologies will be the most promising and adequate ones to be used in Access e-Gov.

1.2 Formalisms used in this document

In the "Interest for Access-eGov" part of the item description, the reader will find the S.W.O.T. matrices (Strengths, Weakness, Opportunities and Threads) that contain following blocks:

	Strengths	**Weaknesses**
Present	• What are currently the strengths of this item? Which arguments foster the usage of this item today?	• What are currently the weaknesses of this item? Which arguments don't foster the usage of this item today?
	Opportunities	**Threats**
Future	• What will be the opportunities of using this item tomorrow / in the future? Which arguments will foster the usage of this item tomorrow / in the future?	• What will be the threats of using this item tomorrow / in the future? Which arguments won't foster the usage of this item tomorrow / in the future?

2 State-of-the-art in e-Government

2.1 Catalogue of e-Government criteria

Modern e-Government landscapes can be categorized in many ways, be it according to the variety of solutions offered to their customers or according to their technical implementations.

Researching e-Government landscapes in 14 countries from all over the world, we figured out several types of criteria that will help the reader of these lines to gain an overview of existing state-of-the-art solutions. The complete list can be found at the end of chapter 2.2.

The types of criteria that the following list is based upon are inspired by the leading objectives of the Access e-Gov project, notably the easy accessibility of government services for its customers (businesses as well as citizens). Another crucial part that we examined was the extent to which information systems can interact with each other in modern e-Government landscapes.

The criteria of interoperability mainly encompass dedicated interconnection of information systems between several agencies on the same administrative level of government. Up to now there are hardly any cross-governmental information links on a mutual basis and on different levels of government. Those can only be observed in a few scenarios as it is the case in the UK and Australia for instance. Openness to external partners also includes the ability to interact on a technical layer with non-governmental or private organizations.

A technical Interoperability Framework is the most common denominator for all of the listed countries as it states the open and standardized technologies to be used in e-Government environments. These frameworks serve public agencies as guidelines for developing new ICT projects as they list technical policies and specifications that have formally been recognized by the government. Agencies will be encouraged to adhere to these frameworks in order to avoid a multitude of bilateral interoperability agreements and to allow for a seamless flow of data exchange between administrations.

Another common denominator of most of the analyzed countries is the implementation of government-wide middleware infrastructures based on XML compliant messages.

“Middleware” is a software layer designed to act as a sort of mediator in and between distributed and heterogeneous applications or information systems and at the same time to provide a set of basic services (messaging, directories, security, authentication, transaction and alike). From the point of view of electronic service delivery, the middleware can act as a bridge between multiple front-end interfaces or delivery channels and multiple back-end legacy applications and systems. Building on standards that can make information systems interoperable, the middleware thus provides the link that can enable them to effectively interoperate. Setting up a generic middleware layer is therefore perceived as a way to build joined-up service delivery capability at affordable cost.

Accessibility	Single-point-of-entry Portal (Gateway), (One-Stop-Shop-)Portal, Web-based Catalogues (Yellow Pages), Stand-alone
Accessibility for physically impaired citizens	e.g. according to W3C-guidelines like WAI ("Web Accessibility Initiative")
Multi channel support	(One-Stop-)Call-Center, Shop Front, Kiosk, Mobile, SMS, Email
Support of AAI functionality	Certificates (soft solution), Smart-cards including certificates (hard solution);
Search facilities	built-in (local & on-board), Portal, database (including metadata)
Interoperability/Openness to external partners	e.g. ID-management open to private partner organizations (banks), other agencies, other states or world-wide (even to individuals) via Internet
Standardization/Uniqueness of solutions	
Quality of service	forms-download, online fill-in, electronic payment, (partial) shop front substitution
User support	On-board-help (at stand-alone applications), Hotline (Call-Centre), Online-help
Usage Guides	Availability of implementation guidelines and best practice descriptions
Web Services/XML-based middleware	

2.2 E-Government solutions in selected countries

A number of countries is already using message-based information exchange services in order to communicate between back-end-systems that may be hosted by different agencies. Most of these systems are based upon open, widely spread XML-standards and are inspired by common Web Services architectures.

The following chapter introduces some of the most advanced examples of modern service-oriented architectures and will also outline their limitations. The selection only depicts those countries whose e-Government solutions might be of interest to Access e-Gov.

In order to provide the reader with a representative overview of the most advanced implementations in e-Government and to go further beyond the usual Eurocentric point of view, we chose examples from four leading countries from around the world.

Some of them share common foundations with regards to technical aspects of interoperability such as the use of service-oriented architectures. Nonetheless we will further on focus upon their leading-edge approaches and direct the reader's interest to conceptual similarities where occurring.

2.2.1 Denmark – open service interfaces with OIOXML

The Danish government is playing a central role in facilitating the exchange of information among public and private organizations and citizens. The government is sanctioning a set of Extensible Markup Language (XML) schemas that model the types of information needed by various government organizations.

InfoStructureBase

The Danish government publishes the XML Schemas it is using via an openly accessible forum called InfoStructureBase (ISB)[1]. It was launched in 2003 to serve as a proliferator for XML-based government standards. One part of the ISB is the repository which contains XML schemas[2]. The purpose of the repository is to store standards and interface descriptions to be used in e-Government applications. This storage repository contains the Danish e-Government standard for addresses, personal names, and other information based on electronic business XML (ebXML) core components. It also contains the XML schema for the known Dublin Core Metadata Element Set that is used as a Danish e-Government standard.

The Danish XML project has chosen a decentralized approach for standardizing interface descriptions and e-Government standards. Users can upload suggestions for interface descriptions and e-Government standards to the repository.

Only users - often Communities of Practise (CoPs) - who are approved by the XML project secretary can upload XML Schemas to the repository. XML schemas require a one-month hearing process before they can be approved by the Danish XML committee and regarded as an e-Government standard.

CoPs that want to standardize an interface description or transform an existing standard to a Danish e-Government standard can use the hearing period for getting the proposed standard evaluated. Comments on the proposal must be addressed by the CoP before approved by the XML committee.

[1] http://isb.oio.dk/info/

[2] http://isb.oio.dk/repository/

Not only CoPs can suggest standards. The XML secretary can also suggest e-Government standards on behalf of the XML committee.

OIOXML

The OIOXML called interface collection is a central key in Danish e-Government efforts besides the InfoStructureBase. Defining a unanimous and publicly available standard for communicating, with and within the government, opens the market for vendors to create applications that comply with the standard. This will create a better competition on the market and better interoperability between different products and systems.

The objective of the standardization work is to determine standards for data exchange between public authorities on the one hand and between public and private institutions on the other one. The Danish XML-Committee has also worked on developing a handbook for Standardization to support the standardization process[3].

The idea is to develop data models, interfaces and Web Services, following a common coordinated method. Thereby creating better consistency throughout the projects implemented by various authorities, on a higher note, this should result in:

- Improved exchange of data within the public sector and between the public and private sectors
- Improved data processing and better access and reuse of data already captured
- Smoother implementation of e-services
- Flexible and integrated services

All XML-interface-descriptions published under the OIOXML-collection are accessible to the public and free to reuse in order to ensure a wide distribution among business- and non-profit-users.

Under the auspices of OIO, the Danish government launched a dedicated WSDL tool designed to facilitate the development of Web Services based on strong XML Schema types. This WSDL tool enables users to create Web Service definitions based on new or existing Schema definitions as they can be found in the InfoStructureBase repository, without having to deal with the many complexities of using XML Schemas in WSDL files.

The tool creates WSDL files that are easy to process using the most common development tools currently found in the market for creating Web Services. Furthermore, two Web Services pilots using OIOXML-interfaces to validate user requests are described and publicly accessible[4].

[3] http://isb.oio.dk/Info/News/OIOXML Naming and Design Rules.htm

[4] http://xmltools.oio.dk

Conclusion

The OIOXML-collection is the most comprehensive publicly available one. The corresponding standardization process ensures that only maturated interface descriptions will be recognized as community use and recommended. With regards to the innumerable quantity if published standards, the Core schemas (that are really related to the e-Government sector in a narrower sense of meaning) can be seen as a very useful, though generic basis for Web Services developments.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• open XML standards • growing Community of Practice • "lessons-learned" reports available	• purpose- and country-specific (Denmark) • interface descriptions only • no ontologies so far
	Opportunities	**Threats**
Future	• OIOXML standards will be a useful basis for own further development • Large community-support	• focus on interfaces only • software architecture based on OIOXML might be too inflexible for major design changes

2.2.2 Singapore - Public Service Infrastructure (PSi)

The government of Singapore[5] has developed the so-called Public Services Infrastructure, a Web-based one-stop e-Government resource for the country's residents.

This initiative provides a development and hosting platform for government agencies, offering back-end integration with databases and systems owned by different agencies. It also consists of a set of e-service design tools to help public servants easily build domain services on their own without or with only minor help from software developers. The PSi-suite functions as a Web Service-based e-Government service for Singapore's residents, who usually access its services via a one-stop portal.

It aims to electronically integrate services across agencies, together with their various systems and applications, in a large-scale, government-wide infrastructure. Its goal is to provide one-stop services to Singapore's businesses and citizens through the Web and other channels.

With the use of so-called generic "building blocks" or "application services", PSi helps in drastically reducing the development times of e-services. Different agencies are able to reuse software components like e-payment or user authentication following a "built-once, reuse-always" approach. Sharing such components reduces incremental costs for the

[5] http://www.gov.sg

implementation of new e-services and reduces the time needed for design and development. It also offers the flexibility to change business requirements in services easily and provides services via multiple concurrent channels. In some cases, development times could be reduced from a month to just a few days.

The PSi consists of three components that are built upon each other: an infrastructure layer, an intermediate zone, and an applications layer.

The application layer is a Service-Oriented Platform containing security and authentication services, allowing the use of digital signatures. Its applications are composed using an e-service-Generator, an Integrated Development Environment for public servants and domain experts.

Infrastructure layer and Intermediate Zone

This PSi-component consists of a multitude of scalable and highly available server applications which are to host the e-services that are made accessible via the government portal. Internally, Web Services are used to access data pools and remote application servers.

This layer is based upon the already existing government backbone network SGNet. The Intermediate Zone (IZ) is in charge of all operations related to data storage and temporarily holds often used and transferred data in so-called data pools. Alternatively, all agencies are free to maintain their own (legacy) databases as well and to directly bind them to their domain's own e-services in PSi without intermediate data storage.

As integral part of the infrastructure layer, the browser-based System Management Console (SMC) allows privileged users and domain experts to administer and deploy all layer-services and user-accounts. For example, privileged users can monitor the migration process between the different PSi environments. According to its current state of development, an e-service is hosted in one of four distinct, but identically equipped environments:

- the “Development environment”, for e-services in an early stage where they're not executable and currently subject to generator-based development
- the “Testing environment” hosts already executable e-services for integration testing purposes
- e-services in the “Quality Assurance” environment are accessible for user acceptance testing
- the “Production environment”, for the delivery and provision of e-services to Internet users via the one-stop- portal

The infrastructure has a Migration Service to manage the migration of e-services from the development environment to the testing, quality assurance and production environment. Each government agency can specify the people to evaluate each e-service before it is allowed to migrate to the next environment in the development cycle.

Application services/building blocks

The application services layer comprises a generic set of common, pre-defined services, so-called “building blocks” that most e-services would utilize. Typically, a Government agency will need to authenticate users, collect payments or grant access to databases from their e-services. PSi satisfies these requirements by providing these three groups of security, e-payment and electronic data exchange services.

Among other services, building blocks offer convenient e-payment modes—such as online payment, cash cards, credit cards, and direct debit—as well as authentication modes, using a nationally unique ID and PIN. Citizens do not have to remember multiple IDs and PINs but can use a single ID and PIN to potentially access all government services. When a payment collection process is invoked within an e-service, the PSi e-payment service will prompt the Internet user to select the desired payment mechanism. PSi provides hardware, software and network connectivity to the relevant service providers to process payments electronically. Government agencies don't need to develop their own e-payment services or make separate arrangements with electronic payment service providers.

The domain expert or e-service developer can also specify the level of security required for the e-service via the PSi e-Service Generator. For example, a developer can specify that an e-service requires authentication of at least a User ID and password. Users of the e-service can then use their User ID and password or a stronger authentication mechanism, (e.g., their smart card) to authenticate themselves.

There are also dedicated building blocks providing support for integration with the database and legacy systems of Government agencies via the EDX service. Each EDX service facilitates data transfer between the agency’s database and its e-services built on PSi. The EDX service acts as a trusted proxy for e-services to query and write information to the database and legacy systems of the agencies.

At the moment, PSi offers a multitude of generic data access services to Business Registry or even grants citizens limited access to their personal entry at the Ministry of Home Affairs.

e-service-Generator

With the help of this abstract development environment, developers (i.e. domain experts just as public servants) can predefine the work flow and the optical look & feel of the e-service that will later on be offered to the citizens via the one-stop-portal.

The business logic and the work flow of the electronic service can be defined with a high-level Business Rule Language that will hence be directly translated into the required platform code for the application. The required user variables can directly be associated in the graphical developer's view with the database entries needed to perform the public service using a drag & drop-metaphor.

The e-service-Generator, the Application services and all domain specific e-services are deployed and executed on the infrastructure layer.

Conclusion

Most government agencies develop services that perform fairly similar functions - collecting payment, authenticating customers, ensuring security, collecting or exchanging data with other agencies. This can be a fairly long development life cycle that is likely to be repeated for most e-services. By introducing software components that are called "building blocks", Singapore's Public Service Infrastructure PSi shortens the development cycle. Its infrastructure, application services and e-service development environment allows agencies to rapidly develop e-services. Components such as payment gateways, electronic data exchange, authentication, and other security features are "built-once, reuse-always" services that agencies do not need to develop on their own, but simply share. By leveraging these building blocks, development time is drastically reduced from months to days.

The platform initiative adopts a thin-client approach. This means that developers do not need to install any software on their machines to start their e-service development. All they require is an Internet browser.

PSi provides the Singapore Government with a highly-integrated and centralized software platform to rapidly develop, deploy and operate public e-services on the Internet. Shared infrastructures and services can be easily reused by all the government agencies, allowing effective and efficient development and life cycle management of e-services.

The Singapore Government's PSi initiative has won several internationally renowned prizes over the last couple of years.

Nonetheless this solution is especially tailored to fit Singapore's flat administrative hierarchy and can hardly be transplanted into historically grown administrative surroundings with federal or even supra-national commitments as it is the case within the borders of the European Union.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• easy-to-use development environment for integrated e-services • covers the complete software life-cycle	• especially tailored to fit Singapore's needs • relying on already existing Singaporean infrastructure • technical underpinnings unknown
	Opportunities	**Threats**
Future	• PSi is an elaborate and already mature e-Government suite and will thus save efforts in basic research	• high tailoring efforts and expenses • bound to proprietary software • lack of clarity regarding the technologies used

2.2.3 Sweden – Government eLink middleware SHS

The "Government eLink" (GeL) initiative is a central infrastructure to communicate via XML-messages between different government agencies, but also with external partners just as

citizens or businesses. It is merely a set of technical specifications that are defining generic services for "interoperable information exchange servers" and can therefore be considered an alternative approach to common middleware for public administration. Each participating agency has to implement its connection "socket" on its own and under own responsibility[6].

All services are registered in a central metadata repository and can be searched by all communication partners.

In 2002, GeL served as one of the models for the European Union eLink-project aiming at establishing a trans-European information exchange network for government agencies based on message-oriented middleware-technologies.

Concrete specifications have been conceived for the GeL-hub-structure, then called SHS, by Swedish government agencies and private companies. The following descriptions are taken from the SHS whitepapers as first published in 2003.

SHS architecture

SHS is a soft-hub architecture (i.e. software-based in contrast to hardware-implemented) to guarantee secure and reliable information exchange based on a Virtual Private Network (VPN) between government agencies but also to private businesses. It encompasses an architecture model and its own XML-standard for data exchange via the hub-structure that has been developed and is still maintained by a Public-Private-Partnership.

The purpose is to facilitate easy access, with high security based on standard protocols to enable secure information exchange across open networks (e.g. the Internet) at a reasonable price. Another purpose is to have an architecture that is extensible to accommodate information exchange between Swedish citizens and public authorities. The architecture is a modular one and easy to extend to let other external partners join the "communication pool" in the future (as it is envisioned to let all regional and community authorities participate in the system). However, the SHS specifications do not cover the communication infrastructure to provide for the "last mile" connectivity to the citizens.

GeL/SHS architecture is widely compliant to W3C and industry standards and will further be developed following the paradigm of most possible conformity, also with regards to envisioned trans-national middleware-communication under auspices of the European Union.

The image below provides an overview of the application architecture (see Figure 2.1).

[6] http://www.statskontoret.se/gel/

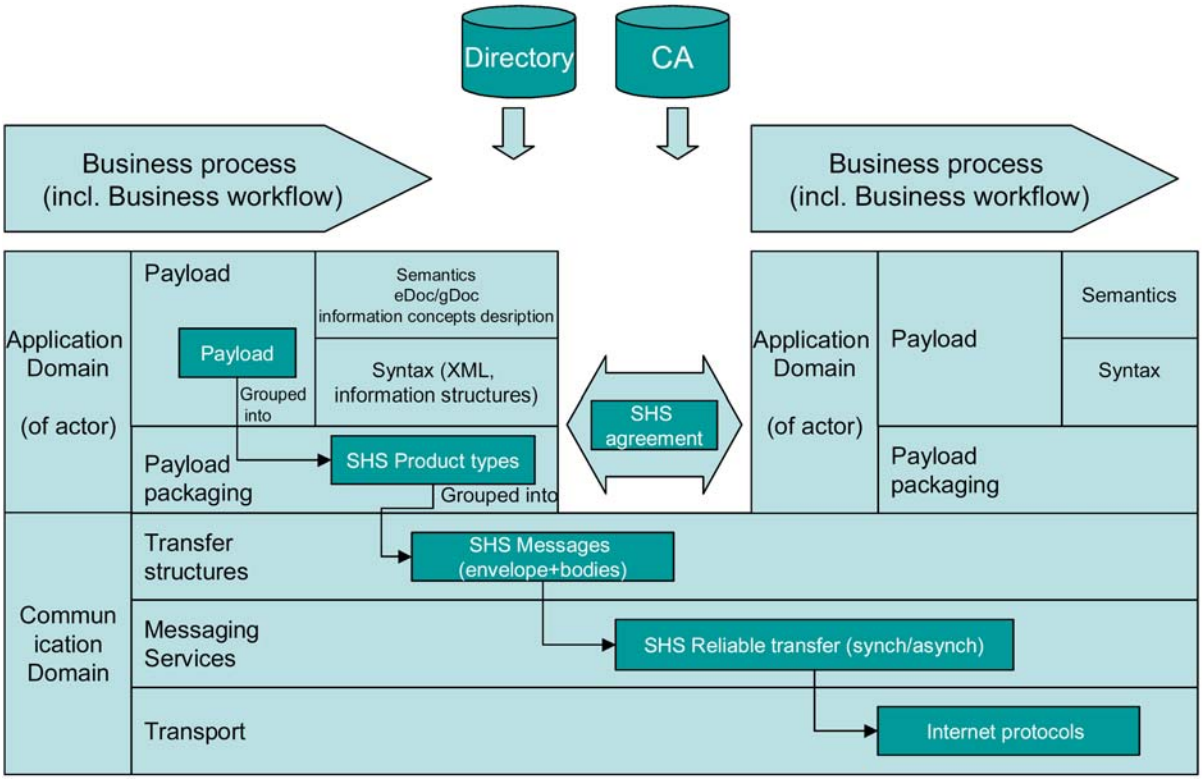

Figure 2.1 Application architecture of SHS

SHS components

The system actors (SHS Actors or Product Owners) communicate via the SHS Messaging Service, the network component that is in charge of the basic services just as routing and will also take action in case of disaster relief and exception handling. SHS knows three communication patterns:bilateral (1:1), one-to-many (1:n), many-to-one (n:1). The SHS Network is made up of so-called SHS Nodes, the server implementations for the very SHS Messaging Service itself.

In order to take part in communicating, each SHS Actor has to implement a local node at his premises or he can indirectly make use the SHS Messaging Service via Web Services offered from other neighboring nodes.

The SHS Network then assigns the Actor a logical SHS Address for further communication purposes. The Messaging Service and the SHS Network itself is transparent to the business applications (Business Systems) of the system actors, as they only indirectly communicate via SHS Nodes.

These applications make use of XML-based SHS Messages in order to exchange data with remote communication partners via the Messaging Service. These Messages have to stick to the DTD specifications as outlined by a Swedish government agencies agreement and they can be composed of one or more SHS Products. The SHS Product itself is an XML-conform business product, describing a certain pre-defined security level for its content (whether it has been signed or encrypted) and telling the recipient which action to take at reception (e.g. reply

or acknowledgment). Its document structure is pre-defined by SHS Product Types that are specific to the SHS Actor, e.g. an “Income Tax Form” or a “Report of Illness”.

The so-called compound SHS message is an SHS message containing a hierarchical structure of embedded simple SHS Messages. The functionality necessary for packing and unpacking these structures is not part of the SHS interfaces and therefore up to the business applications.

Specific syntax rules will be made available by the sending Actor as SHS Agreements, so that SHS Messages can be interpreted by the recipient according to their initial meaning. Such an Agreement also declares the transaction costs (e.g. certificate validation costs), the communication mode the message was sent in and the necessary reactions from the recipient's side when receiving this type of SHS Message[7].

These and other kinds of information are present in a SHS Repository, the central data hosting component of the overall SHS architecture. The Directory is globally available to all parts of the system, be it SHS capable business applications, Actors, as well as the SHS Messaging Service itself, since it is able to retrieve information about participating organizations, services (i.e. SHS Products), Agreements and Addresses. The repository is accessible via standardized LDAP-protocol-commands. Each SHS Node can save a local copy of the SHS Directory, to shorten response times and avoid delays for the users.

The Messaging Service makes use of this Directory when retrieving Product Types and Agreements and in case of resolving an Actor's SHS Address. In turn each Actor has to register his organization, his Node's SHS Address and the Product Types in use. Each business application has to retrieve its initial configuration from the Directory at starting time.

From a protocol perspective (see image), the SHS middleware layer takes the role of a servant for transparent data exchange between high-level business applications and basic low-level transportation services on the network (i.e. Public Internet). The SHS layer in this protocol stack is responsible for routing the XML-Messages and to ensure the exchange of signed acknowledgment messages as an officially recognized reply by the recipient.

On the technical transport layer, SHS sessions will be encrypted using standardized SSL-technology. The specifications only mention SHS interface descriptions for the high-level business applications, leaving room for implementation to the individual communication partners.

The specification whitepaper states four different types of interfaces available to make use of SHS messaging services:

- FaP – Format and Protocols

[7] http://www.statskontoret.se/shs/pdf/shs-dtd.pdf

This interface describes low-level protocols and formats (like S/MIME). These descriptions shall enable external software developers to design SHS compliant business applications for use in government agencies and private partner companies.

- External processes (stand alone programs)

  This interface layer consists of two external stand alone programs that can be directly used from business applications to communicate via SHS. The programs `shssend` and `shsfetch` incorporate functionality to generate, encrypt and sign, and in turn to receive SHS Messages and validate their authenticity.

- API – Application Programming Interface for C and JAVA

  Developers can also make use of the functionality as described above, by directly calling the respective operations from their business applications code.

- Web Services

  Each Actor is free to implement local Nodes or to indirectly use SHS Messaging Service via Web Services from other Nodes in synchronous or asynchronous mode. Their functionality is nonetheless limited compared to the one obtained from local node implementations.

Figure 2.2 shows a network perspective of the overall SHS system.

Conclusion

GeL is a set of standards defining a number of generic services, which is used by commercial partners to build interoperable "information exchange servers". When widely installed, these servers therefore constitute a "distributed middleware infrastructure", enabling seamless and secure interoperation between public sector bodies but leaving more autonomy and choices of implementation to individual organizations than centralized government-wide infrastructures will allow. GeL inspired the European Union project IDA'a – eLink to establish a trans-national e-Government middleware.

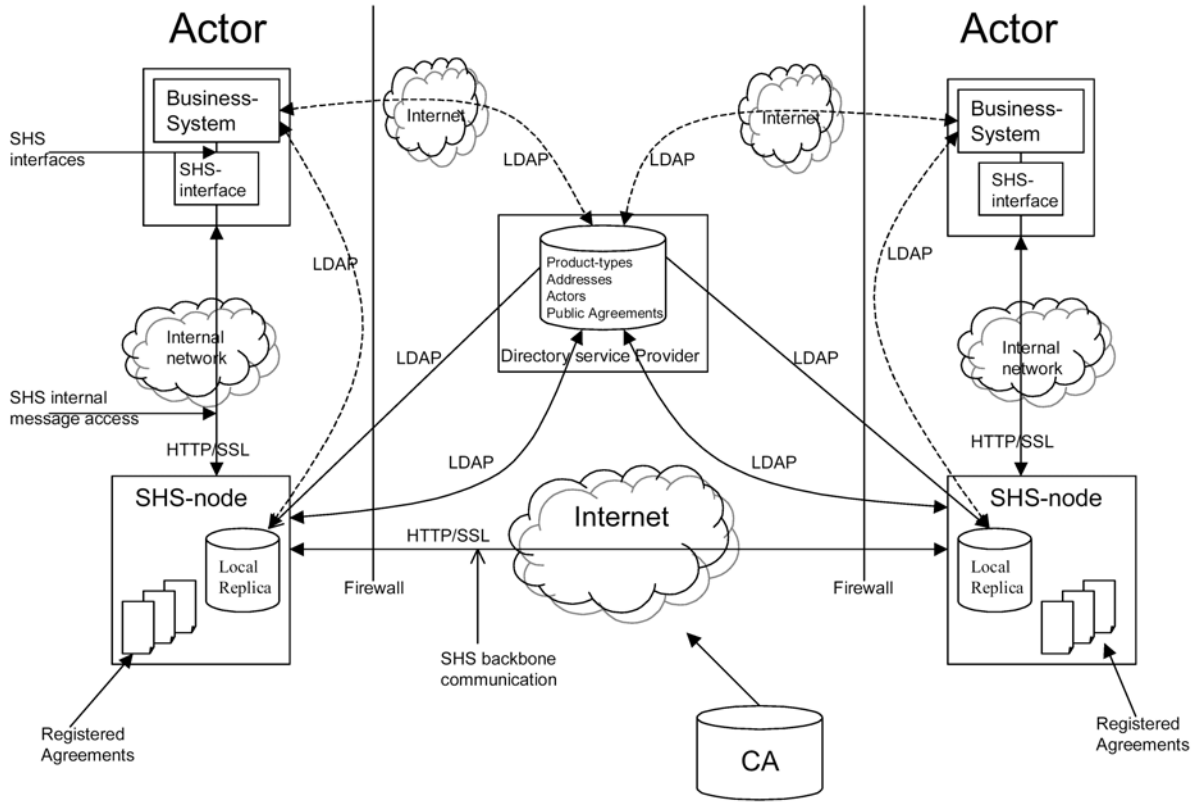

Figure 2.2 Network perspective of SHS

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• elaborate architecture and design • well-documented • based on open interfaces and standards	• ontologies not supported yet • latest documents from 2003 • integration of cascading Web Services unsolved • orchestration mechanisms not sufficiently documented
	Opportunities	**Threats**
Future	• with SHS being part of IDA eLink, it will be a largely supported platform standard throughout the EU	• expenses are assumed to be high when introducing additional ontology capability

2.2.4 United Kingdom – multi-purpose middleware

Government Gateway

The central registration and authentication component "Government Gateway" aims at empowering citizens to use government services online and is available since 2001. In order to use services offered from different government portals, citizens have to be registered centrally at the Gateway.

It also serves as a central (though not the only) point-of-entry to the majority of interconnected public services offered by government authorities.

This system also serves government authorities as a middleware-layer for inter-agency communication. Similar to Singapore's PSi initiative, the Gateway's reusable foundation services enable government organizations to focus on the rapid engineering of electronic services, by avoiding to rebuild common underlying components required for online services. All its interfaces are also available as Web Services, so that the responsible Cabinet Office categorizes the gateway as the implementation of a "cross-governmental SOA (service oriented architecture)".

The Government Gateway consists of:

- *Gateway User Interface*

  a multitude of centrally hosted and administered server services to generate the User Interface[8]

- *Authentication and Authorization services*

  The Government Gateway provides several foundation services with basic identity management services and technical interfaces being the most important ones.

- *Transaction Engine*

  a middleware solution built upon a central XML-based software hub to safeguard message exchange between front-end and back-end systems (i.e. between agencies and the Gateway server)

The Authentication & Authorization block works with User Id/password-credentials or digital certificates issued from Trusted Third Parties, depending on the chosen transaction mode. Currently over 6 million users are registered at the Gateway to use the more than 50 different e-services made available from 25 public authorities. The Government Gateway supports the paradigm of "cross-enrollment" so that citizens can register once for as many services as they wish (or as they're allowed to by the system rules). The linked agencies and their portals mostly use their own user credentials for each single one of their services. Therefore the

[8] http://www.gateway.gov.uk

Gateway consists of a dedicated mapping service, to allow citizens to login with their Gateway credentials when requesting a portal service. The mapping service then has to look up the corresponding agency-specific user credentials in a database ("mapping") and attaches them to the redirected portal service request. The user only needs to bear his Gateway credentials in mind and the specific agency can still use its legacy authentication mechanisms.

The Government Gateway complies with the so-called t-scheme as known in other Anglo-Saxon countries. Public services are categorized into (here) four security levels and access to them is granted according to a flat authorization scheme corresponding to the service's security level ranging from "Level 0: no authentication required" to "Level 3: authentication required to protect personal safety". The choice of whether to authorize a user or not is still up to the corresponding agency.

The software hub called "Transaction Engine" is at the heart of the overall Government Gateway. All Web Service messages are directed to and forwarded by this hub. When passing the Transaction Engine, an authenticity check is performed by default on the messages sent by citizens (via the portal) as well as by participating agencies and businesses.

The Transaction Engine works through two methods:

Citizen requests reach the back-end-systems in "Ad-hoc use over the Internet" mode. End users transparently communicate with the back-end-systems via the portal or application on which they are working. The use of this model means that portals are able to support the completion of electronic forms interactively on the Internet or locally on a PC. In both cases the Internet and the Gateway provide the mechanism for the submission of the completed forms to the appropriate organizations and the return of a corresponding receipt. This mode is used for all service requests originating from external partners and citizens using the Gateway web interface whose filled-in forms are sent via an XML-based Document Submission Protocol (DSP).

The other communication mode "Hub and Spoke" requires a dedicated connection to the Government Gateway using a locally implemented node called Departmental Interface Server (DIS) that is equivalent to Sweden's SHS Node. This set of product specifications describe a proxy server that shall be able to provide access to all the Gateway services just as secure messaging and document authentication. Communication to other DIS components in the network is handled by as special protocol also called "Hub and spoke". This communication mode is preferably used for inter-departmental data exchange.

DIS servers are conceived to provide a range of modules for rapid integration into existing legacy applications and technologies.

A secure Gateway Web Service SOAP Portal interface, available only to trusted users (such as government portals and applications) provides a range of authentication and authorization support, including the ability for a portal to register and enroll an end user in the Gateway. This SOAP-API also allows the remote administration of user accounts. Since 2004, participating organizations are able to let their Web Services communicate with each other via a secured Peer-to-Peer-protocol, routed over the Gateway's own authentication framework.

The XML-based GovTalk envelope used by the Gateway specifies the content needed to transport messages between Gateway-using organizations. It usually encompasses the sender's credentials and identifiers (including a W3C compliant signature where appropriate) and the name of the service to which the message should be routed.

The image underneath gives an overview of the Gateway's topography.

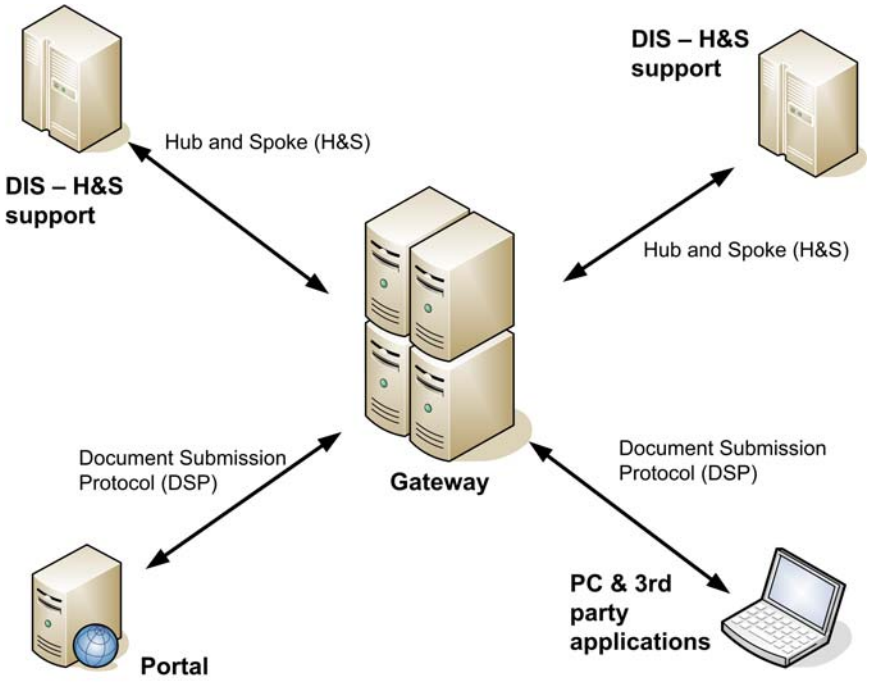

Figure 2.3 Government Gateway architecture

e-GIF

Introducing the e-Government Interoperability Framework e-GIF in the 1990s, the British government was the first European one to set up a common list of technical standards and approaches to be explicitly used throughout the whole public service sector[9].

It is a mandatory set of policies and standards to both enable and enforce public sector interoperability. This shall enable information systems from different departments to be used effectively across the public sector, since technical interoperability is the precondition for seamless logical integration of (electronic) public services.

Adherence to e-GIF specifications and policies is mandated for new systems and legacy systems involved with electronic service delivery targets and for data exchange.

[9] http://www.govtalk.gov.uk/schemasstandards/egif.asp

The document consists of high level management policy statements and implementation regimes on the one hand, and a list of technical specifications and policies on the other one.

The e-GIF defines the technical policies and specifications governing information flows across government and the public sector. They cover interconnectivity, data integration, e-services access and content management.

Public servants, domain experts and external IT suppliers are being helped in their efforts to comply with the UK's electronic-Government Interoperability Framework (e-GIF) through a website forum called GovTalk[10] and the e-GIF Compliance Assessment Service. XML schema definitions are also made available to the public to encourage the reuse for other open interface descriptions.

Specifications in e-GIF are analyzed in the fields of Interconnection, Data Integration, Content Management Metadata and Access.

E-Government Metadata Standards

The e-Government Metadata Standard e-GMS lists the elements and refinements that will be used by the public sector to create metadata for information resources. It also gives guidance on the purpose and use of each element. e-GMS supports the policy on metadata as outlined in the Interoperability Framework. It has been developed to better meet records management needs and reflect changes to related international standards. Compliance with the e-GMS is mandatory, to ensure consistency of metadata across the public sector.

Along with the e-GMS, the Integrated Public Sector Vocabulary (IPSV) has been published as an 'encoding scheme' for populating the Subject element of metadata[11]. Currently this monolingual thesaurus encompasses 756 preferred and 1334 non-preferred terms across the whole public service sector. It is a first step in semantically enriching electronic services in Great Britain.

Conclusion

With the release of a new version of its Integrated Public Sector Vocabulary in April 2006, the British government's efforts in e-Government now focus on semantic enrichment and the assigning of metadata to e-services. This monolingual thesaurus is the first step towards an integrated service delivery in terms of a real one-stop solution that the Government Gateway is going to be.

Together with the Web Services-based "Government Gateway" it will serve as the technical underpinnings of a distributed delivery platform for electronic public services. Since the British government solutions are still in evolution, a well-balanced verdict cannot be spoken yet.

[10] http://www.govtalk.gov.uk

[11] http://www.esd.org.uk/standards/ipsv/

Nonetheless, the level of proficiency can let external spectators assume that British government is about to become one of the lead users of modern, semantics-based e-Government technologies.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• well-documented interfaces • elaborate architecture design regarding 3rd party integration • already large industry-support regarding software • thesaurus-based annotation of services • growing Community of Practice	• centralized gateway approach • integration of cascading Web Services unsolved • orchestration mechanisms not sufficiently documented
	Opportunities	**Threats**
Future	• future inclusion of ontologies will be easier to manage based on existing thesaurus-annotation	• so far proprietary software implementations only

2.3 Tabular synopsis of important e-government solutions

	Accessibility	*Accessibility for physically impaired citizens*	*Multi channel support*	*Support of AAI functionality*	*Search facilities*	*Interoperability/Openness to external partners*	*Standardization/Uniqueness of solutions*	*Quality of service*	*User support*	*Usage Guides*	*Web Sservices / XML-based Middleware*
	Single-point-of-entry Portal (Gateway), (One-Stop-Shop-)Portal, Web-based Catalogues (Yellow Pages), Stand-alone	e.g. according to W3C-guidelines like WAI ("Web Accessibility Initiative")	(One-Stop-)Call-Center, Shop Front, Kiosk, Mobile, SMS, Email	Certificates (soft solution), Smart-cards including certificates (hard solution);	built-in (local & on-board), Portal, database (including metadata)	e.g. ID-management open to private partner organisations (banks), other agencies, other states or world-wide (even to individuals) via Internet		forms-download, online fill-in, electronic payment, (partial) shop front substitution	On-board-help (at stand-alone applications), Hotline (Call-Centre), Online-help	Availability of implementation guidelines and best practice descriptions	
Australia	Gateway, Portals, Yellow Pages	W3C-conformity envisioned	Shop Front, Call-Centers (some cross-agency ones [Centrelink and Family Assistance Office FAO])	"MediCare-Health-card" and generic smart-card-initiative	Portal, database (AGLS-metadata and TAGS-thesaurus)	Several initiatives	industry- and W3C-standards, own Open-Source-developments driven by public agencies	forms-download, online fill-in, electronic payment	Online-help	Yes	Yes
Austria	Yellow Pages/One-Stop-Shop, Portals	W3C-compliant XML-forms	Shop Front, Mobile, SMS, Call-Centers, Email	“Citizen Card“ (generic smart-card-solution in co-operation with several agencies and private partners/banks)	Portal, database	“Citizen Card“-compliant smart-cards are issued by private organisations (CAs, banks, mobile phone service	industry- and W3C-standards; some own, but open standards	forms-download, online fill-in, electronic payment, partial shop front substitution (“digitaler	Online-help	Yes	No

						providers)		Amtsweg")			
Brazil	Gateway, Portals, Yellow Pages	W3C-compliant (e-MAG-guidelines)	Shop Front, Call-Centers	Certificates envisioned	Portal, database	Tax refund via private online-banking-systems	industry- and W3C-standards	forms-download, online fill-in, electronic payment	Hotline, Online-help	Yes	Yes
Canada	Gateway, Portals, Yellow Pages	Some dedicated W3C-compliant sites	Shop Front, Mobile, SMS, Call-Centers; since 2006: One-Stop-Call-center	certificates	Portal	Cross-agency-use envisioned	Not stated	forms-download, online fill-in, electronic payment	Hotline, Online-help	Yes	Yes
Czech Republic	one-stop-shop-portal for citizens & enterprises	Not stated	Shopfront (at local Post Offices)	Certificates mandatory for some e-services	Portal, database	Direct access to all e-services from Czech Post Intranet in shopfronts	industry- and W3C-standards	forms-download, online fill-in	Online-help	No	No
Denmark	Gateway, Portals, Yellow Pages, One-Stop-Shop	Envisioned since 2003	Shop Front, One-Stop-Call-Center, Email	certificates	Portal, database	Web Services for business partners	industry- and W3C-standards; open XML-interface collection called OIOXML	forms-download, online fill-in, electronic payment	Online-help	Yes	Yes
Estonia	Integrated One-Stop-Shop for citizens, business partners and public servants	Some dedicated W3C-compliant sites	Shop Front, Call-Centers, Mobile, SMS	smart-cards	Portal, database	X-Road-Middleware open to private business customers	W3C, IETF, OASIS, some Open-Source-developments driven by public agencies	forms-download, online fill-in, electronic payment	On-board-help, hotline, Online-help	No	Yes

France	Gateway, Portals, Yellow Pages, One-Stop-Shop	W3C-compliant	One-Stop-Call-Center, Shop Front	certificates, introduction of smart-cards postponed	Portal, database	Liberty Alliance as private partner for ID-verification	industry- and W3C-standards	forms-download, online fill-in, electronic payment	Hotline, Online-help	Yes	Yes
Germany	Federal Gateway, Portals, Yellow Pages	own W3C-like guidelines("barrier free Internet")	Shop Front, Call-Centers, Mobile	certificates, smart-cards envisioned	Portal, database	So far for federal agencies only	industry- and W3C-standards; some own, but open standards	forms-download, online fill-in, electronic payment (limited)	Online-help	Yes	Yes
Singapore	Gateway, Yellow Pages	Some dedicated W3C-compliant sites	Call-Centers, Shop Front, Mobile	certificates, Smart-cards (optional: generic "SingPass"-PIN)	Portal, database	PSi makes use of private banking databases; Web Services-initiative for private business customers	Some proprietary tools, based on industry- and W3C-standards	forms-download, online fill-in, electronic payment, shop front substitution	Hotline, Online-help	No	Yes
South Korea	Gateway, Portals, Yellow Pages ("Service Guide")	Envisioned since 2005	Shop Front, Mobile, Call-Centers	certificates	Portal, database	G4C to be accessible to banks by 2007	industry- and W3C-standards, proprietary cryptographic approach	forms-download, online fill-in, electronic payment	Online-help	No	No

Sweden	Gateway, Portals, Yellow Pages, One-Stop-Shop	Not stated	Not stated	certificates, smart-cards optional	Portal, database	Official e-ID card compliant to private "BankID"	industry- and W3C-standards	forms-download, online fill-in, electronic payment	Hotline, Online-help	No	Yes
United Kingdom	Gateway, Portals, Yellow Pages	envisioned in e-GIF	Shop Front, Kiosk, Call-Centers	certificates	Portal, database (based on IPSV thesauri)	Gateway accessible to private partners via DSP	Based on industry- and W3C-standards, open to the public	forms-download, online fill-in, electronic payment	Hotline, Online-help	Yes	Yes
United States	Gateway, Portals, Yellow Pages, One-Stop-Shop	W3C-conformity envisioned; dedicated portal: disability.gov	Shop Front, One-Stop-Call-Center	envisioned	Portal, database	Openness in G2B only	industry- and W3C-standards	forms-download, online fill-in, electronic payment	Hotline, Online-help	Yes	Yes

2.4 EU-projects

The evolution towards integrated IT-based public services, as described above, shows the necessity to adopt new ways of interacting with and between public service institutions. As member-countries of the European Union are confronted with the challenge of providing even border-crossing public services, the way needs to be paved for new approaches suited for them.

Some academic projects already envision practicable solutions to problems that are originating from interoperability issues. These kinds of problems are encountered in many countries of the Western Hemisphere that want to further automate their public service sector.

Most of these EU-funded initiatives mainly focus upon semantic enrichment of electronic services and their aggregation and orchestration towards combined “complex e-services”. The following chapter will give an overview of their current state of development and shall help the reader to assess the different approaches taken.

The majority of these projects focuses on the technical level and thus still lacks a citizen-centred point of view that could be taken by implementing software components tailored to assist the citizen when applying for a public service. In these projects, citizens' needs take a background position compared to technical aspects that are very often predominating.

Therefore, new approaches in e-Government have to put the emphasis on easy service accessibility for customers.

2.4.1 TERREGOV

Since its founding in 2002, the TERREGOV project group sets out the goal to empower local community authorities (Local Agencies) to make all their services available to the public in the form of Web Services as well as to use other agencies' services in the same manner[12]. This service usage between Local Agencies and Service Providers will be monitored and coordinated at central, TERREGOV-own Regional Interoperability Centres.

The communication between all of the components is based on a Web Services architecture. Available services, called eProcedures, can be aggregated to abstract, more elaborate service clusters:

- Atomic eProcedures thereby depict simple “YES/NO” business processes (e.g. family allowances) that can be triggered by system actors. They can nonetheless consist of several Web Service-calls.
- Composite eProcedures depict sophisticated business processes that are assembled using more than one Atomic eProcedure.

[12] http://www.terregov.eupm.net

TERREGOV makes use of current state-of-the-art W3C technologies, such as OWL-S for describing ontologies. Semantic registries register Web Services and BPEL files will be used to orchestrate and compose eProcedures.

The core-projects of TERREGOV follow different approaches and application purposes, so-called "streams". "Stream 1" projects implement basic Web Service architectures to make e-Government processes electronically available. "Stream 2" tries to semantically enrich Web Services for easy discovery and execution, whereas "Stream 3" focuses upon supporting public servants with reference and implementation guides.

TERREGOV-subsystem

A common infrastructure is made available to all project partners that is consisting of several subsystems. They are in turn made up of various software modules.

- Ontology Management: This set of modules consists of tools to generate and extend the central reference ontology.
- Community of Practice: These centrally hosted tools encompass a document-repository for index-search and an online-forum to let public servants share their experiences in daily use of the TERREGOV-system.
- Workflow Management: These programs are used to develop eProcedures and to monitor their runtime behavior.
- Semantic access to Web Services: This subsystem offers real-time search facilities and storage operations for semantic descriptions.
- The Intra-Agency Interoperability Layer empowers authorities to publish their eProcedures as an integrated chain of Web Services.
- Categorization of Citizen Cases: According to the current life situation of the service requester (i.e. the citizen), these modules select the public services that the requester is eligible for.
- Security module is used to encrypt and sign Web Services messages.

Some of the components have to be installed at the premises of the Local Agency whereas others are only available centrally at the Regional Interoperability Centres.

Prototypes

In order to test the different modules of the subsystems for their collaboration in daily use, four regional pilot trials are held all over Europe. They implement several distinct use cases. Legacy systems are also integrated into the TERREGOV-environment to simulate a more realistic scenario using wrapper applications.

- *United Kingdom*

  The intention behind the UK-pilot and its use case “Data Exchange” is to demonstrate how TERREGOV technology can be instrumented for simple, automated data exchange using Web Services. Performance data shall be read out of a data source (Service Provider) via a central data handling service-hub and be forwarded to a data sink (Local Agency). Information about these services is semantically described using OWL-format. Users of the system have to provide the necessary data that shall be read out, using a special “Form Filling” module.

- *France*

  The French prototype implements an Atomic eProcedure to ask for an official authorization. Some TERREGOV modules are installed at the premises (Data Access, SMTP Access and Management System Information) to control ten Web Services that shall electronically replace a previously paper-bound process. The community staff has been provided with a manual and installation guide featuring detailed descriptions on how to set up and administer the modules.

- *Italy*

  The prototype of the integrated “Regional Online Net of Social Services” of the Veneto region shall facilitate the application process for Social services and electronically replace the previous process. These services are based on WSMO (resp. WSML) technology for service description. Using unique citizen identifiers, public servants are able to read out all relevant data relating to an applicant from legacy systems via a TERREGOV-web-front-end. They can then judge the applicant's eligibility for the service. Local legacy systems and the TERREGOV-pilot will be communicating using an already existing interoperability model from the Italian government that had been designed to let regional and community administrations cooperate electronically. Its interfaces were not WSDL-compliant and therefore had to be wrapped to integrate them into the TERREGOV-environment.

- *Poland*

  The e-services of the Polish pilot implementation consist of a number of rudimentary Web Services that are to substitute manual processes in the local public service authorities. Some legacy databases in the Social Service sector are the technical underpinnings of this prototype's implementation. The relevant basic modules alongside with the Semantic Web Services Registry and the TERREGOV Interoperability Layer are to be installed soon at the premises.

By 2006, all parts of the project shall be tested in productive environments.

Conclusion

This project can be considered a step towards the right direction with regards to system interoperability on a regional and community level. Nonetheless, the TERREGOV solution still lacks an inter-, if not supra-national point of view needed in today's European Union

member states, as the project only operates on a regional level of administration. Services can so far be detected and requested by public servants only (apart from the French pilot). Additionally, single agencies appear to be too rigidly bound up to the Regional Integration Centres: some functionality that is needed on the local level is only offered at the Centre.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• modular design approach • shared functionality with Regional Integration Centres • several trials are implementing different use cases • uses more than one state-of-the-art ontology (OWL-S and WSMO)	• complex allocation of functionality separated between Local Agencies and Regional Integration Centres • unclear decision making process for choosing the modules to be installed at the premises • focus on regional e-services only • implementation manuals only provided in some cases • no multi-channel support
	Opportunities	**Threats**
Future	• the elaborate platform design allows easy tailoring of components	• reduced autonomy of the Local Agencies

2.4.2 OntoGov

The OntoGov project group is working on a generic software engineering environment in order to better accompany the complex life-cycle of electronic e-Government services on the whole[13].

A set of ontologies will be used to describe data and its functional use to build Web Services. They serve as unified methodologies to semantically enrich certain steps in the software engineering process.

Architecture

It allows the development of even more sophisticated Composite Services with the help of a generator tool for process modeling. They are described utilizing an orchestrated chain of Atomic Services. The decision paths or workflows of these combined Atomic Services (i.e. the Composite Services) are already defined at configuration time, even before their first execution and stored in BPEL file format.

[13] http://www.ontogov.com

As common service oriented architecture is not capable of process modeling, OntoGov extends the standard W3C specifications for implementing SOA. The approach taken by the OntoGov team is much more efficient and less susceptible to execution errors at runtime.

The logical architecture model is made up of three layers:

- Business Modeling Layer

  On this level, abstract services can be designed using a graphical generator tool. This task has to be fulfilled by domain experts with extensive knowledge of the public services in question or public servants who have been trained on this matter. The chain of Atomic Services called "Service Ontology / Service Model" will be generated with an ontology editor, the "Service Modeller". Each e-service is represented by one Service Ontology, with each one having its own profile, process model and life-cycle.

- Configuration Layer

  The business logic of the Service Model is realized through a set of software components. OntoGov-application-experts have to describe the corresponding Web Services interfaces. Atomic Services will be mapped to their technical Web Services implementations using ontologies that are stored in a Web Services Orchestration Registry (WSOR). These mappings register the WSDL-operations to be called and the input and output formats of the corresponding Atomic Service.

- Runtime Layer

  Modules on this technical level orchestrate all Atomic Services and guarantee the correct execution order of their corresponding Web Services as predefined in the Configuration Layer. Each so-called Broker Machine has to dispose of a unified runtime environment at its premises. An orchestration component called "Process Engine" selects the Service Ontology in question, interprets it and forwards the output to the next Atomic Service in the Process Model. Synchronisation with other Web Services will be done by a Synchronisation Manager component.

Ontologies

OntoGov is aiming at describing the whole life-cycle of an e-service, using an abstract high-level ontology with a collection of several lower-level ones to be deducted from it. They are especially tailored to fit the specific service structures of the public agency in question. This set of specific ontologies is represented using semantic Web Services standard like OWL-S and WSMO. They are separated into the following three groups:

- *Meta-ontologies* define language schemes to allow modeling e-government services via more basic ontologies.

- *Domain-oriented ontologies* serve the agency to autonomously model their concrete e-services and all corresponding data. These ontologies are specializations of those belonging to the cluster of meta-ontologies (e.g. Legal-Municipality o., Legal-State o., Legal-Federal o.) and have to be defined for each single governmental institution. The

Service Ontology of the Business Modelling Layer is the main element of this category.

- *Administration Ontologies* have been explicitly created to describe the maintenance process of government e-services.

The documentation available online also mentions the following three examples of ontologies without categorizing them into the three types stated above.

- *Life-Event Ontology* is used for classification of the e-government services according to the kind of service they deliver. It includes concepts such as residential affairs, permissions, moving, education, etc. This ontology is based upon the Swiss Standard eCH-001 which aims at giving an overview of all available government services, listing 1200 of them. This Life-Event ontology is common for all the users.

- *Lifecycle Ontologies* are used to describe the information flow and the agencies' internal decision-making processes. Utilized to justify decisions in design and to outline interdependencies, they can be consulted when changes are necessary. Four different types of decisions are taken into account: Design Goals (the goals of the change), Design Resources (necessary to implement the changes), Design Techniques (the strategies used to reach the goals) and Design Constraints (of external kind). Additionally, Lifecycle Ontologies encompass decision-supporting information just as change costs, priority and impact. They are intended to support the transition from knowledge acquisition to implementation and provide an automated, user-centered approach to the entire process life-cycle (including analysis and execution phase).

- With the help of *Web Service Orchestration Ontologies*, particular services can be integrated at runtime into the workflow of another service. They describe all configuration information necessary to call a Web Service. These ontologies consist of an unequivocal, direct link between an Atomic Service and the WSDL description of a Web Service. The mappings are stored in the WSOR repository.

  Three distinct kinds of mappings exist: syntactical mappings (with lexicographic pattern-matching), structural mappings (with inputs and outputs being taken into consideration) and context mappings (regarding the context out of which an operation has been called).

The development of new services follows to the scheme as described underneath: After its modeling, a machine-readable version of the service description will be generated and stored in OWL-S file format. OntoGov-experts utilize the WSOR repository to configure the basic Web Services. Each Atomic Service will be assigned a WSDL description of an already existing Web Service prior to its first use and this mapping will be stored.

OntoGov is currently tested in three pilot trials under realistic conditions

Conclusion

OntoGov encompasses a multitude of needful ontologies to describe and support the life-cycle of e-Government. This elaborate approach mainly focuses on the software engineering side rather than on detection and orchestration of e-services and thus leaves room for interpretation on how these ontologies can be used in practical scenarios. In addition, the process modeling is the only part that can be done by other than application-experts, so OntoGov lacks a certain degree of transparency to public servants using the system. Its ontology work is heavily bound to OWL-S and can therefore not easily be converted to a newer, more flexible semantic technology like WSMO.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• layered architecture • generic software engineering environment covers the whole service life-cycle • multitude of ontologies supporting the Web Service life-cycle	• ontology documentation insufficient • focusing on OWL-S only • not all tasks can be fulfilled by domain experts/public servants
	Opportunities	**Threats**
Future	• ontologies will constitute reusable and condensed public service domain knowledge	• high expenses when redesigning to fit other basic techniques

2.4.3 EU-PUBLI.com

The Italian project EU-PUBLI.com (pursued by various universities between 2002 and 2005) aimed at further automation of inter-agency communication through the structured use of integrating Web Services architectures and ontologies. The approach taken resembled the one of OntoGov: complex public macro-processes (e.g. applying for a tax benefit) shall be broken up into atomic micro-processes that are implemented as Web Services. Europe-wide cooperation of public service agencies shall be achieved through a common framework architecture[14].

The goal was to show whether and how Web Services can be used efficiently by public service agencies on a large scale with direct interfaces to end users.

[14] http://www.eu-publi.com

Architecture

EU-PUBLI.com extends the common service oriented architecture of W3C that has been regarded as being too generic in terms of directly meeting public service needs. New components were added to manage the life-cycle of macro-processes. Each authority has to fully implement a generic tool collection called “Cooperation Layer” on the premises in order to participate in the network. This set consists of the following three groups of modules:

- Cooperative Gateway

  The back-end of the architecture provides all agency-services that are implemented as Web Services. It also consists of an orchestration engine on peer-to-peer-basis to coordinate and delegate service execution. Dedicated “System Wrappers” allow the inclusion of legacy systems at the premises by enabling them to deliver their services as Web Services. The orchestration engine subsystem uses BPEL4WS technology to coordinate the workflow of composite macro-processes and the execution of its atomic components. Services can be dynamically looked up at runtime using orchestration schemes (“process definitions” in BPEL file-format) and bound up during the workflow's execution.

  While executing a particular business process, orchestration engines of different authorities can communicate with each other to negotiate the shift of the process control to the other party. The “Transaction Engine” therefore guarantees the channel's reliability between two e-services regardless of the actual business process in question. An additional “Security Engine” takes care of encryption, access control and data protection.

- Information Manager

  A so-called “Semantic Repository” is used to store service- and orchestration-schemes of the currently active e-services in an enhanced UDDI-registry. The “Compatibility Engine” has to guarantee a certain quality of service level and is able to replace single atomic e-services with equivalent ones during execution of an active macro-process. This module also improves availability in the overall system with its load balancing capabilities and allows hot-updates to be deployed at runtime of a macro-process.

- Presentation Layer

  The front-end systems will have to generate the graphical user interfaces using XForms technology. Depending on the language preferred by the communication partner, a “Semantic Engine” translates all displayable content based on OWL-S and UDDI registry entries and further harmonizes data representation on a technical level.

The only centrally hosted component of the overall EU-PUBLI.com network system is the “Global Information Manager Registry”, a single-point-of-access for all the complex e-services that cannot be executed (or found) locally. It is charged with registering and lookup of all publicly available services in its internal UDDI registry.

Orchestration

As soon as a public service agency makes internal services that are part of macro-processes, available to the outside world using Web Services, an orchestration engine is needed to coordinate the workflow of the overall macro-process. A BPEL engine instance as part of the orchestration engine subsystem is responsible for executing the scripts of a process, coordinating the single Web Services and forwarding its results to the recipient in the given order.

To delegate or shift responsibility for the process control to the next (other peer's) orchestration engine, a "send/accept"-protocol had been developed based on BPEL4WS technology. This protocol will be interpreted by the "Distributed Orchestration Engine" and is implemented using simple Web Services. During execution of a distributed macro-process,

1. the corresponding BPEL file describing the process will be modified and
2. transferred to the new peer's orchestration engine.
3. That executes the following atomic process,
4. restores the state of the computation (explicitly sent by the old peer) and
5. continues the activity from the point at which the old peer has interrupted its activities.

From that particular point of time on, the initiator of the shifting of the workflow-process (the "old peer") is no longer involved into the macro-process-workflow (unless the workflow again passes over the responsibility for the orchestration).

The I/O-information flow will be analyzed by a network component called "Interceptor". In case of a macro-process shift due to a workflow instruction, it shall send asynchronous messages to other peer instances.

Conclusion

EU-PUBLI.com's main challenge has been described as proving whether or not Web Services orchestration can be applied in practical e-government scenarios under realistic service load. According to the publications available, this claim has been proved successfully in prototypical simulations. The recommendations are that Web Services can efficiently be used for electronic government services on a large scale with their interfaces directly accessible to service requesters (i.e. the citizens).

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• modular, but integrated software suite that needs to be fully installed at the premises • elaborate orchestration mechanisms	• theoretical approach • simulations only • annotation of services insufficiently documented • unknown functionality of central

		registry component
	Opportunities	**Threats**
Future	• platform services can easily be integrated by other platforms due to clear architecture design	• high expenses when implementing in the field

3 Semantic Web Services Formalisms

3.1 OWL Web Ontology Language for Services (OWL-S)

Name	**OWL-S** (Web Ontology Language for Services)
Actors	Authors: OWL-S (formerly DAML-S) Coalition
Current version	OWL-S 1.1 http://www.daml.org/services/owl-s/1.1/
History	2006-03 OWL-S 1.2 prerelease 2004-11 OWL-S 1.1 current version 2003-11 OWL-S 1.0 2001-05 DAML-S 0.5

Description

OWL-S is OWL ontology for semantic description of the Web Services. The structure of the OWL-S consists of a service profile for service discovering, a process model which supports composition of services, and a service grounding, which associate profile and process concepts with the underlying service interfaces.

Service profile has functional and nonfunctional properties. Functional properties describe the inputs, outputs, preconditions and effects of the service (IOPEs). The nonfunctional properties describe the semi-structured information intended for human users for service discovery, e.g. service name, description and parameters which incorporates further requirements on the service capabilities (e.g. security, quality-of-service, geographical scope, etc.).

Service model specifies how to interoperate with the service. The service is viewed as a process which defines the functional properties of the service (IOPEs), together with details of its constituent processes (if the service is a composite service). The service model functional properties can be shared with the service profile.

OWL-S distinguishes between atomic, simple, and composite processes. OWL-S atomic processes can be invoked, have no sub-processes, and are executed in a single step from the requester's point of view. The simple processes are used as elements of abstraction, they are viewed as executed in a single step, but they are not invocable. Composite processes consist of the simple processes and define their workflows using control constructs, such a sequence, split, if-then-else or iterate.

Service grounding enables the execution of the Web Service by binding the abstract concepts of the OWL-S profile and process model to concrete messages and protocols. Although

different message specifications are supported by OWL-S, the widely accepted WSDL is preferred.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• covers all aspects of the Semantic Web Services • based on the standardized OWL language (many application developed around OWL and OWL-S)	• OWL-S uses single modeling element (Service Profile) for requester and provider views • OWL-S recommends the combination of various rule languages (i.e. SWRL, DRS or KIF) with OWL to specify conditions. Combinations with SWRL or DRS lead to undecidability or leave semantics open. Furthermore, it is not clear how OWL and KIF descriptions interact (KIF syntax is treated as a string.
	Opportunities	**Threats**
Future	• structured process model (easy to visualize)	• has to be extended for traditional services

3.2 Web Service Modeling Ontology (WSMO)

Name	**WSMO** (Web Service Modeling Ontology)
Actors	Authors: ESSI WSMO working group, part of the European Semantic System Initiative (ESSI) Cluster http://www.essi-cluster.org/
Current version	WSMO 1.2 http://www.wsmo.org/TR/d2/v1.1/
History	2005-04 WSMO 1.2 current version 2005-02 WSMO 1.1 2004-09 WSMO 1.0 2004-02 WSMO 0.2 first version

Description

The Web Service Modeling Ontology (WSMO) is a conceptual model for describing semantic Web Services. WSMO consists of four major components: ontologies, goals, Web Services and mediators.

Ontologies

Ontologies provide the formal semantics to the information used by all other components. WSMO specifies the following constituents as part of the description of ontology: concepts, relations, functions, axioms, and instances of concepts and relations, as well as non-functional properties, imported ontologies, and used mediators. The latter allows the interconnection of different ontologies by using mediators that solve terminology mismatches.

Goals

A goal specifies objectives that a client might have when consulting a Web Service, i.e. functionalities that a Web Service should provide from the user perspective. In WSMO a goal is characterized by a set of non-functional properties, imported ontologies, used mediators, the *requested capability* and the *requested interface* (see the Web Services description).

Web Services

A Web Service description in WSMO consists of five sub-components: non-functional properties, imported ontologies, used mediators, a capability and interfaces.

The *capability* of a Web Service defines its functionality in terms of preconditions, postconditions, assumptions and effects. A capability (therefore a Web Service) may be linked to certain goals that are solved by the Web Service via mediators. Preconditions, assumptions, postconditions and effects are expressed through a set of axioms and a set of shared all-quantified variables.

The *interface* of a Web Service provides further information on how the functionality of the Web Service is achieved. It describes the behavior of the service for the client's point of view (service choreography) and how the overall functionality of the service is achieved in terms of cooperation with the other services (service orchestration).

A choreography description consists of the states represented by ontology, and the if-then rules that specify (guarded) transitions between states. The ontology that represents the states provides the vocabulary of the transition rules ant contains the set of instances that change their values from one state to the other. The concepts of an ontology used for representing a state may have specified the grounding mechanism which binds service description to the concrete message specification (e.g. WSDL).

Like for the choreography, an orchestration description consists of the sates and guarded transitions. In extension to the choreography, in an orchestration can also appear transition rules that have as postcondition the invocation of a mediator that links the orchestration with the choreography of a required Web Service.

Mediators

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

- *OOMediators* - import the target ontology into the source ontology by resolving all the representation mismatches between the source and the target;
- *GGMediators* - connect goals that are in a relation of refinement allowing the definition of sub-goal hierarchies and resolve mismatches between those;
- *WGMediators* - links a goal to a Web Service via its choreography interface meaning that the Web Service fulfills the goal; or links a Web Service to a goal via its orchestration interface meaning that the Web Service needs this goal to be resolved in order to fulfill the functionality;
- *WWMediators* - connect several Web Services for collaboration.

Web Services Modeling Language (WSML)

WSMO is formalized using the Web Service Modeling Language (WSML) which is based on description logic, first-order logic and logic programming formalisms. WSML consists of a number of variants based on these different logical formalisms, namely:

- *WSML-Core* is based on the intersection of Description logic and Horn logic;
- *WSML-DL* extends WSML-Core to an expressive Description logic and offers similar expressiveness to OWL-DL;
- *WSML-Flight* is an extension in the direction of Logic programming and incorporates a rule language, while still allowing efficient reasoning;
- *WSML-Rule* extends WSML-Flight to Logic programming language, which does not require rule safety and allows to use function symbols; and
- *WSML-Full* unifies all WSML variants under a common first-order umbrella with non-monotonic extensions.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• WSMO relies on loose coupling with strong mediation. • WSMO provides a family of layered logical languages which combines conceptual modeling with rules. • non-functional core properties, based on the Dublin Core Metadata Standard, can be used to describe	• some parts of the specification are new or are not finished

	all WSMO elements • provide opportunity for sophisticated goal-oriented discovery	
	Opportunities	**Threats**
Future	• defines mapping of OWL ontologies with mediation • it is trying to be W3C compliant technology, possibly it will become the standard in the Web Services ontologies	• orchestration and choreography is based on the abstract state machine where workflow is encoded in transition rules against OWL-S where the workflow is structured with control constructs. • new semantic language • has to be extended for traditional services

3.3 Web Service Semantics - WSDL-S

Name	**WSDL-S** (Web Service Semantics)
Actors	Authors: Authors: University of Georgia and IBM
Current version	WSDL-S 1.0 http://www.w3.org/Submission/WSDL-S/
History	2005-11 WSDL-S 1.0 current version

Description

WSDL-S is a small set of proposed extensions to Web Service Description Language (WSDL) by which semantic annotations may be associated with WSDL elements.

WSDL-S defines URI reference mechanisms to the interface, operation and message WSDL constructs to point to the semantic annotations defined in the externalized domain models. WSDL-S defines following extensibility elements and attributes:

- *modelReference* element - allows for one-to-one associations of WSDL input and output type schema elements to the concepts in a semantic model;
- *schemaMapping* attribute - allows for many-to-many associations of WSDL input and output complex type schema elements to the concepts in a semantic model. It can point to a transformation (for example XSLT), from XML data to the external ontological data in RDF/OWL or in WSML;
- *precondition* and *effect* elements - are used on WSDL interface operations to specify conditions that must hold before and after the operation is invoked. The conditions can be specified directly as a expression with format defined by the semantic language or by reference to the semantic model;

- *category* element - provides a pointer to some taxonomy category. It can be used on a WSDL interface and is intended to be used for taxonomy-based discovery.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• strong development group behind WSDL-S. • WSDL-S is built on WSDL standards. • WSDL-S does not depend on the semantic model language.	• new approach. Still in research, authors suggest to be careful. • there is a small set of real examples. • does not explicitly support orchestration and choreography.
	Opportunities	**Threats**
Future	• semantic model language independency. • possible further research and extension by the development group.	• orchestration and choreography is not covered. • further (our) examples can show some weaknesses of this approach • extends WSDL, which is not oriented to traditional services.

3.4 Business Process Execution Language for Web Services (BPEL4WS)

Name	**BPEL4WS** (Business Process Execution Language for Web Services)
Actors	Authors: IBM, Microsoft, BEA (version 1.1 also SAP and Siebel Systems)
Current version	BPEL4WS 1.1 http://www-128.ibm.com/developerworks/library/specification/ws-bpel/
History	2002-07 BPEL4WS 1.0

Description

BPEL4WS (Business Process Execution Language for Web Services) is a specification that models the behavior of Web Services in a business process interaction. It represents a convergence of the ideas in the XLANG and WSFL specifications. Both XLANG and WSFL are superseded by the BPEL4WS specification and therefore they are not presented separately in this report. It is based on the XML grammar which describes the control logic required to coordinate Web Services participating in a process flow. An orchestration engine can interpret this grammar so it can coordinate activities in the process.

BPEL4WS and WSDL

BPEL4WS is a layer on the top of WSDL (Web Services Description Language). WSDL defines the specific operations and BPEL4WS defines how the operations can be sequenced. Every BPEL4WS process can be considered as a Web Service using WSDL. WSDL describes the public entry and exit points for the process. WSDL (XML) data types are used within a BPEL4WS process to describe the information that passes between requests. WSDL might be used to reference external services required by the BPEL4WS process.

Orchestration and Choreography

BPEL4WS provides support for both executable and abstract business processes. The executable process models a private workflow. The abstract process specifies the public message exchanges between parties. The executable processes provide orchestration support while the business protocols (abstract processes) focus more on the choreography of the services.

BPEL activities

BPEL4WS includes support for basic and structured activities. The basic activities might be receiving or replying to message requests as well as invoking external services (tags <receive>, <reply> and <invoke>). The structured activities specify what activities should run in what order – the whole process flow. These activities also provide support for conditional looping and dynamic branching. The structured activities might specify that certain activities should run sequentially or in parallel.

Containers and partners

Containers and partners are two important elements within BPEL4WS. A container is a variable for exchange in the message flow. A partner could be any service that the process invokes or any service that invokes the process. Each partner is mapped to a specific role that it fills within the business process. This is managed by containers.

Transactions and exception handling

In BPEL4WS, a set of activities can be grouped in a single transaction - tag <scope>. This tag signifies that the steps enclosed in the scope should either all complete or all fail. Within this scope, the developer can then specify compensation handlers that should be invoked if there is an error. BPEL4WS provides a robust exception handling mechanism through the use of throw and catch clauses, similar to the Java programming language.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• strong development group behind BPEL4WS • BPEL4WS suitable for representing workflow	• need to employ semantic into WSDL – new approach
	Opportunities	**Threats**
Future	• support and further improvement and development • representation of workflows	• coping with implementation of semantics

4 Frameworks and Tools for Semantic Web Services

4.1 OWL-S Tools

Name	**OWL-S Tools**
Actors	Authors: various authors
Current version	http://www.daml.org/services/owl-s/tools.html
License	(L)GPL

Description

A set of disparate OWL-S tools exists, but not a complete execution environment based on OWL-S concepts. The following list describes some of the tools:

- *OWL-S editor* - is divided into three main parts: creator, validator and visualiser. The creator enables to create an empty OWL-S description either from a template or through a wizard called "OwlsWiz" which accepts an input WSDL file and extracts partial information from it to create a basic OWL-S description. The validator part provides for validing of the URIs used in the OWL-S descriptions and also validate the syntax of the ontologies. The Visualiser part enables the user to visualise the descriptions and service compositions in a graphical manner by exploiting UML activity diagrams.

- *DAML-S Matchmaker* - is a Web Service that helps make connections between service requesters and service providers. The Matchmaker allows users and/or software agents to find each other by providing a mechanism for registering service capabilities. Registration information is stored as advertisements. DAML-S Matchmaker employs techniques from information retrieval, AI, and software engineering to compute the syntactical and semantic similarity among service capability descriptions. The matching engine of the matchmaking system contains five different filters for namespace comparison, word frequency comparison, ontology similarity matching, ontology subsumption matching, and constraint matching. The user configures these filters to achieve the desired tradeoff between performance and matching quality.

- *ASSAM (Automated Semantic Service Annotation with Machine learning) WSDL Annotator* - is an application that assists the user in annotating Web Services. Annotations can be exported in OWL-S. WSDL files can be annotated with an OWL ontology with a point-and-click-interface, but the key feature of ASSAM is machine learning assisted annotation: After a training phase, ASSAM can make recommendations on how to annotate datatypes in the WSDL. ASSAM is still under development and should be seen as a "technology preview", not an industrial-strength application.

Interest for Access-eGov

	Strengths	Weaknesses
Present	• together cover all aspects of the Semantic Web Services • support for semi-automatic annotation.	• various independent tools which have to be integrated • some methods are still under development and should not be used as an industrial application
	Opportunities	**Threats**
Future		• possible integration problems

4.2 WSMX

Name	**WSMX** (Web Service Execution Environment)
Actors	Authors: WSMX working group (DERI, NIWA, The Open University, SAP, BT, Ontotext lab (Sirma), EPFL Global Computing Center (GCC))
Current version	0.3 http://www.wsmx.org
License	LGPL
History	WSMX Core 0.3 release 2006-03-24 WSMX Core 0.2 release 2005-07-01 WSMX Core 0.01 release 2004-07-26 - initial version on SourceForge

Description

Web Service Execution Environment (WSMX) is an execution environment which enables discovery, selection, mediation, and invocation of Semantic Web Services. WSMX is based on the conceptual model provided by WSMO, being at the same time a reference implementation of it. It is the scope of WSMX to provide a test bed for WSMO and to prove its viability as a mean to achieve dynamic interoperability of Semantic Web Services.

WSMX functionalities can be classified in two main categories: first is the functionality that should be part of any environment for Semantic Web Services and second, the additional functionality coming from the enterprise system features of the framework. In the first case, the overall WSMX functionality can be seen as an aggregation of the components' functionalities, which are part of the WSMX architecture. In the second case, WSMX offers features such as plugging mechanism that allows the integration of various distributed components, an internal workflow engine capable of executing formal descriptions of the components behavior or a resource manager that enables the persistency of WSMO and non-WSMO data produced during run-time.

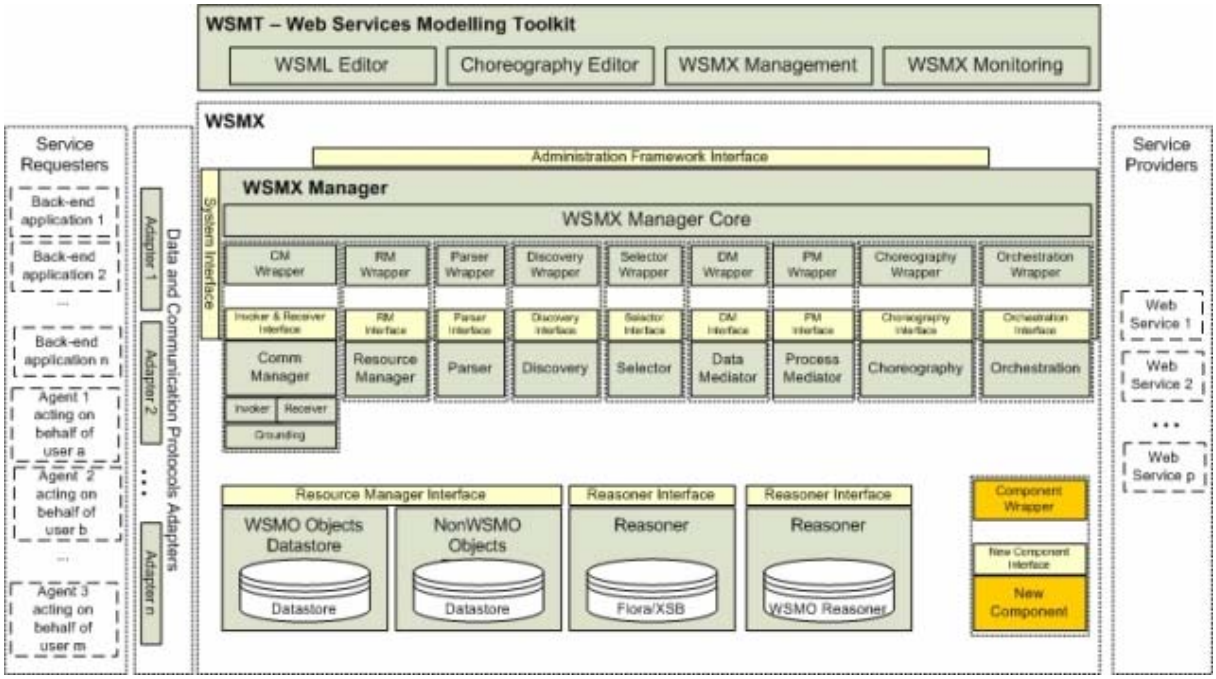

Figure 4.1 WSMX Architecture.

The main components that have been already designed and implemented in WSMX are: Core Component, Resource Manager, Discovery, Selection, Data and Process Mediator, Communication Manager, Choreography Engine, Web Service Modeling Toolkit and Reasoner (Figure 4.1).

- *Core Component* is the central component of the system connecting all the other components and managing the business logic of the system
- *Resource Manger* manages the set of repositories responsible for the persistency of the WSMO and non-WSMO related data flowing through the system. It is offering an uniform and at the same time the only (in the framework) point of access to potentially heterogeneous implementation of such repositories;
- *Discovery component* has the role of locating the services that fulfill a specific user request. This task is based on the WSMO conceptual framework for discovery which envisions three main steps in this process: Goal Discovery, Web Service Discovery, and Service Discovery. Currently in WSMX, the Service Discovery covers only the matching of user's goal against service descriptions based on syntactical consideration. In case that more than one suitable services are found, WSMX offers support for choosing only one of them; this operation is performed by the Selection component by applying different techniques ranging from simple "always the first" to multi-criteria selection of variants (e.g., Web Services non-functional properties as reliability, security, etc.) and interactions with the requester.
- Two types of mediators are provided by WSMX to resolve the heterogeneity problems on data and process level. Data mediation is based on paradigms of ontologies, ontologies mappings and alignment with direct application on instance transformation. The Process mediation offers functionality for runtime analysis of two given patterns

(i.e., WSMO choreographies) and compensates the possible mismatches that may appear.

- *Communication Manager* through its two subcomponents, the Receiver and the Invoker, enables the communication between the requester and the provider of the services. The invocation of services is based on the underlying communication protocol used by the service provider and it is the responsibility of an adapter framework to implement the interactions that require a different communication protocol than SOAP.
- *Choreography Engine* has to provide a means to store and retrieve choreography interface definitions, to initiate the communication between the requester and the provider in direct correlation with the results returned by the Process Mediator, and to keep track of the communication state on both the provider and the requester sides. In addition, it has to provide grounding information to the communication manager to enable any ordinary Web Service invocation.

Even if the reasoner is not a part of the WSMX development effort, a WSML compliant reasoner is required by various components such as Data Mediator, Process Mediator and Discovery. The Web Services Modeling Toolkit (WSMT) is a framework for rapid creation and deployment of homogenous tools for Semantic Web Services. An initial set of tools includes a WSML Editor for editing WSML and publishing it to WSMO repositories, a WSMX Monitor for monitoring the state of the WSMX environment, a WSMX Data Mediation tool for creating mappings between ontologies, and a WSMX Management tool for managing the WSMX environment.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• research group investigates possibilities for integration with other technologies (OWL-S, WSDLS) • modular architecture	• some modules are not implemented or have limited functionality.
	Opportunities	**Threats**
Future	• in the scope of our project (3 years), WSMX is going to be a complex and ready to use framework • we have an opportunity to participate in WSMX development and thus we are able to change some specifications	• investigate which modules have to be modified for traditional services • there is possibility, that WSMX community will not be able to finish needed parts of the framework, before we use it, thus we will be forced to develop them ourselves • no messages in the SourceForge discussion forum on WSMX

4.3 IRS III

Name	**IRS III** (Internet Reasoning Service)
Actors	Authors: The Open University
Current version	IRS III http://kmi.open.ac.uk/projects/irs/
License	GNU Lesser General Public License

Description

The Internet Reasoning Service (IRS) is a framework for Semantic Web Services that supports the publication, location composition and execution of Web Services based on their semantic descriptions. IRS supports the conceptual model defined by WSMO and also provides mappings for service descriptions provided in OWL-S.

The next paragraphs provide a brief description of the main functionality offered by IRS:

- *Publication* - in IRS has two roles. The first is where a Web Service represented by a URI endpoint is associated with a semantic service description known to IRS. The second is where standalone Java or Lisp code is wrapped to make it appear as a Web Service and then, as in the first case, the service is associated with a semantic service description known to IRS. Once a service has been published to IRS it is available to be used in the achievement of a user goal.
- *Discovery* - IRS has its foundation in an earlier project called IBROW which made the distinction between tasks that need to be solved and problem solving methods that "provide abstract, implementation-independent descriptions of reasoning processes which can be applied to solve tasks in specific domains". Bridges were then defined to provide mappings to counter problems of heterogeneity. Adopting the WSMO conceptual model, tasks in IRS are modeled as goals while problem solving methods are modeled as services. Discovery in IRS is based on matching the pre- and post-conditions defined in the semantic descriptions of goals and services known to the IRS server.
- *Composition* - in IRS is goal based. Where a goal can not be achieved by a single service, it is decomposed into smaller goals each of which can either be mapped directly to a specific service description or can be resolved by the IRS server at runtime. The approach uses mediators to handle the problems of data heterogeneity.
- *Execution* - IRS server handles the invocation of Web Services based on mapping the semantic description of the service choreography to the actual implementation of the service. This part of the IRS functionality is hidden from clients of IRS.

The main components of IRS are the IRS Server, the IRS Publisher and the IRS Client. The IRS Server stores the descriptions of goals, mediators and Web Services along with domain ontologies. Discovery, composition, mediation, reasoning and invocation are all controlled by the IRS Server. The IRS Publisher carries out the tasks required for publication as described

above. Finally, the IRS Client provides a user-interface for goal-based Web Service invocation.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• integrated platform • good integration with the semantic web community	• competitive to WSMX • choreography and orchestration do not follow WSMO specification and they are implemented in a non standard way
	Opportunities	**Threats**
Future	• better integration with the WSMO community • although competitor to WSMX, it would benefit from extended usage of WSM* family products	• DIP ends 12/06, will IRS III be mature enough to survive on its own by then?

4.4 METEOR-S

Name	**METEOR-S**
Actors	Authors: LSDIS Lab, University of Georgia
Current version	METEOR-S 0.8 http://lsdis.cs.uga.edu/projects/meteor-s/
License	Commons Public License Version 1.0

Description

METEOR-S project proposes the application of semantics to existing Web Service technologies. In particular the project endeavors to define and support the complete lifecycle of Semantic Web Service processes. The project extends WSDL to support the development of Semantic Web Services using semantic annotation from additional type systems such as WSMO and OWL ontologies.

METEOR-S proposes an enhancement of UDDI to facilitate semantic discovery as well as a framework for SWS composition. METEOR-S is not based on an overall SWS conceptual model and is rather a collection of related discrete tools than a single, encapsulated architecture.

The current implementation of METEOR-S allows for the (i) the creation of WSDL-S descriptions from annotated source code, (ii) the automatic publishing of WSDL-S descriptions in enhanced UDDI registries, and (iii) the generation of OWL-S descriptions, from WSDL-S, for grounding, profile and service. The architecture is depicted in Fig. 4.2.

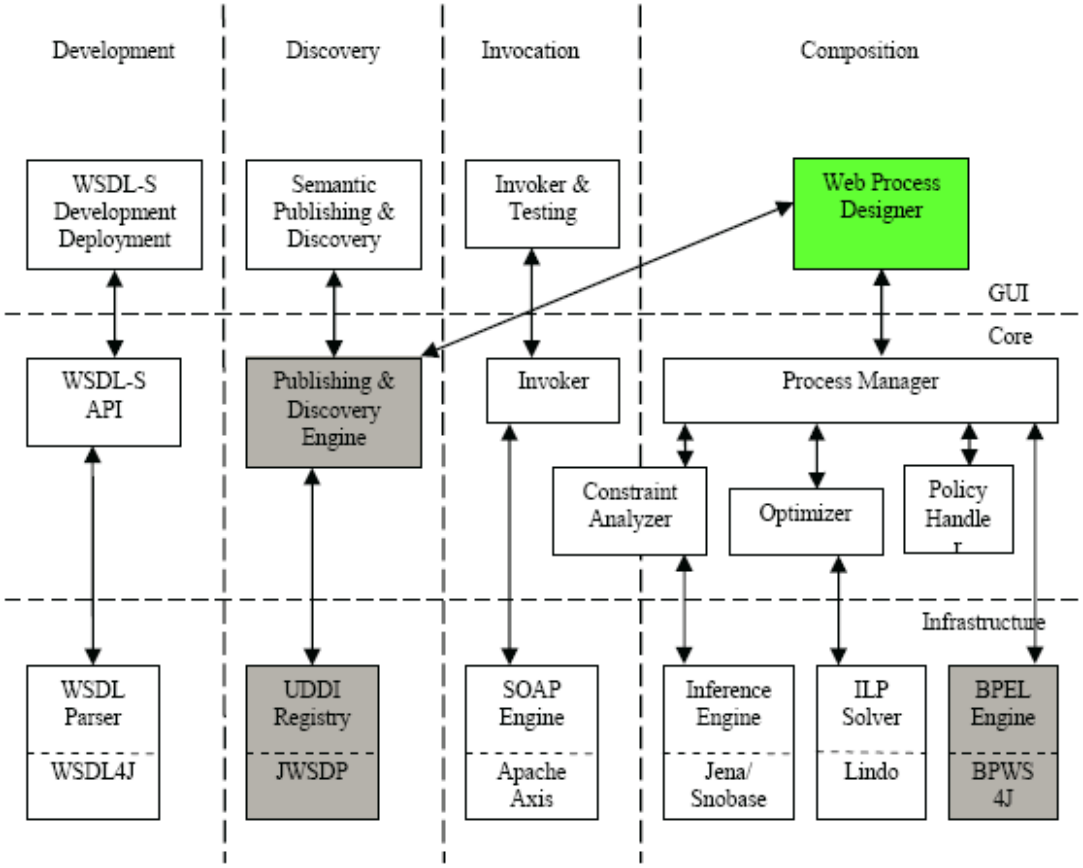

Figure 4.2 METEOR-S Architecture.

The development module provides a GUI based tool for creating Semantic Web Services representing using WSDL-S. The tool provides support for semi-automatic and manual annotation of existing Web Services or source code with domain ontologies. The publication and discovery (MWSDI) module provides support for semantic publication and discovery of Web Services. It provides support for discovery in a federation of registries as well as a semantic publication and discovery layer over UDDI. The composition module consists of two main sub-modules – the constraint analysis and optimization sub-module and the execution environment. The constraint analysis and optimization sub-modules deal with correctness and optimization of the process on the basis of quality service constraints. The execution environment provides proxy based dynamic binding support to BPWS4J execution engine for BPEL4WS.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• tools for annotation of WSDL and BPEL • possibility to publish and discover	• fairly new tools

	Web Services	
	Opportunities	**Threats**
Future	• possibility to implement semantic into WSDL and BPEL by using WSDL-S	• further use can show some weaknesses of these tools

4.5 CSAP

Name	**CSAP**
Actors	Authors: Webocrat, EU Project IST-1999-20364
Current version	0.9
License	modified openssl license

Description

This module originated in the EU project Webocrat which was aimed at supporting direct participation of web users in democratic processes. Its main goals included:

- Discussion management
- Web publishing
- Opinion polling
- Intelligent information retrieval

The very nature of these tasks implies the use of a sophisticated security framework, which was implemented in the scope of this project as the so-called CSAP: Communication, Security, Authentication and Privacy module. CSAP provides the following services:

- Identification and authentication
- Access control and authorization
- Auditing
- Session management

Figure 4.3 shows the basic design approach. This three layered architecture is highly flexible and extensible because the core component, the Kernel, uses plugins as service providers.

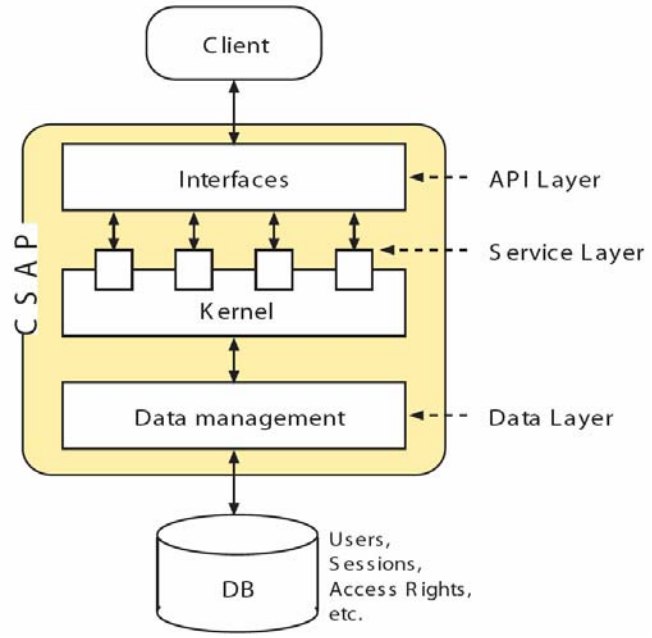

Figure 4.3 CSAP Architecture

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• proven technology • joint development of members of this consortium • fits well to the security requirements in Access-eGov • well documented	• no large user base • only partial Web Service integration • no support for Peer-to-Peer yet
	Opportunities	**Threats**
Future	• easily extensible • some original developers now in Access-eGov • possibility to turn CSAP into an open source project with all the benefits	• no uptake in open source community

4.6 OBE (Open Business Engine)

Name	**OBE** (Open Business Engine)
Actors	Authors: Sourceforge.net members (open source community project)
Current version	OBE 1.0 RC1 (2006-01) http://sourceforge.net/projects/obe
License	modified Apache License

Description

The Open Business Engine is a Java workflow engine implementing Workflow Management Coalition Open Standards (WfMC: XPDL, WAPI, Auditing). OBE will provide support for XPDL for process definition with the ability to plug in parsers for other definition languages, WAPI for client access, using RMI (Remote Method Invocation) initially with XML-RPC and SOAP implementations to follow, a complete implementation of the WfMC audit specification and support for the Workflow Interoperability specification. OBE is J2EE compliant, executing on top of any J2EE application server. OBE will also include an embedded version of the engine for use directly in Java applications.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• based on open standards • easily extensible through plugins	• small number of developers • odd user interface

	• some high profile use-cases • high development activity	
	Opportunities	**Threats**
Future	• development partially funded by MetricStream • incorporated into commercial products while still staying open source	• further involvement of MetricStream unsure • many comparable projects forked that had a private company as main contributor

4.7 Enhydra Shark/JaWE

Name	Enhydra Shark/Enhydra JaWE editor
Actors	Authors: Enhydra open source community
Current version	Enhydra Shark 2.0beta1 (2006-05) Enhydra JaWE 2.0-3 (2006-05) http://shark.objectweb.org/
License	GNU Lesser General Public License (LGPL)

Description

Enhydra Shark is an extendable workflow engine framework including a standard implementation completely based on WfMC specifications using XPDL (without any proprietary extensions) as its native workflow process definition format and the WfMC "ToolAgents" API for server side execution of system activities.

Storage of processes and activity instances is done via a configurable persistence API. The standard persistence layer implementation uses the lightweight Enhydra DODS O/R mapping. A more heavyweight J2EE EJB persistence layer alternative will follow. Every single component (persistence layer, transaction manager, scripting engines, process repository,...) can be used with its standard implementation or extended/replaced by project specific modules. This way, Enhydra Shark can be used as a simple "Java library" in servlet or swing applications or running in a J2EE container (supporting a session beans API and maybe using EJBs for persistence), Corba ORB or be accessed as a web service.

WfMC WDF API specifications will be used to attach the JAWe editor or self-written programs to runtime instance information and even to modify instances while they are running. Using this approach Enhydra Shark supports dynamic workflows which can modify themselves to support more complex workflow scenarios or organizational exception handling.

The Enhydra JaWE graphical XPDL editor can be used to produce XPDL process definitions for Enhydra Shark. Currently a Swing based administration GUI can be used to do administrative work. JMX extensions and an HTML based administration client will follow. Additional APIs will be available for repository access, logging, repository persistence, event notification and scripting engine adapters for transition evaluations.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• very open architecture • adheres to standards • good choice of service wrappers	• tightly integrated into a company portfolio • might have very small number of independent developers
	Opportunities	**Threats**
Future	• growing Enhydra application server usage patterns may provide momentum to the workflow tools	• company losing interest in the open source versions

4.8 BPWS4J

Name	**BPWS4J** (Open Business Engine)
Actors	Authors: IBM
Current version	BPWS4J engine 2.1 http://www.alphaworks.ibm.com/tech/bpws4j
License	IBM International License Agreement for Early Release of Programs

Description

The IBM Business Process Execution Language for Web Services JavaTM Run Time (BPWS4J) provides a platform for the execution of business processes written in the BPEL4WS (as described above). It also provides a set of sample applications demonstrating the use of BPEL4WS and a tool that validates BPEL4WS documents. In detail, it provides tools for analysis (verification), validation (simulation) and execution (exception handling) of the process models and also tools supporting State Machines, Petri nets, activity diagrams etc.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• strong development group at IBM for this engine (Mangala Gowri -	• does not support appropriate level of semantics

	part of IBM's India Research Lab.)	
	Opportunities	**Threats**
Future	• further support and maintenance • possible help in the official public forum	• coping with implementation of semantics

4.9 JBoss jBPM

Name	**JBoss jBPM**
Actors	Authors: SourceForge open source community
Current version	JBoss jBPM 3.1.1 (2006-05) http://www.jboss.com/products/jbpm
License	GNU General Public License (GPL)

Description

jBPM is a WorkFlow Management System. jBPM defines process definitions within files using the JBoss process definition language. jPDL is a graphic-oriented programming (GOP) language based on a model of nodes, transitions, and actions. In this model, nodes are commands executed as they are encountered during the flow of a process definition. Transitions direct the flow of execution of a process definition, and actions perform specific logic as a node or transition event occurs.

JBoss jBPM is encapsulated within the following components:

- Process engine: This component executes defined process actions, maintains process state, and logs all processes
- Process monitor: This module tracks, audits, and reports the state of all processes during execution
- Process language: The process definition language (jPDL) is based on GOP
- Interaction services: These services expose legacy applications as functions or data to be used in process executions

In jBPM, process definitions are packaged as process archives. A process archive is passed to the jPDL process engine for execution. The jPDL process engine traverses a process graph, executes defined actions, maintains process state, and logs all process events.

JBoss jBPM 3.0 delivers the capability of developing new automated business processes and workflows with industry-standard orchestration using Business Process Execution Language (BPEL), a flexible and pluggable API, a native process definition language, and a graphical modeling tool.

Interest for Access-eGov

	Strengths	**Weaknesses**
Present	• currently, this solution seems to be the most mature in an open source environment. • documentation (users guide, architecture, developers manual, etc.). • help forums on the JBPM site.	• the modelling language JBPL is not a standard.
	Opportunities	**Threats**
Future	• jboss has a large support in the open source community. • jBPM could one day become a de-facto standard.	• failure to build a specific jBPM community

5 Conclusions

While researching e-Government implementations in 14 countries from all over the planet, it turned out that most nations are currently using XML-based middleware systems to loosely integrate their various back-end systems. In the long run, one-stop solutions are envisioned to combine all kinds of e-services into a single bundle under a consistent web-interface for citizens and businesses.

Sweden's middleware solution, SHS is exemplary in terms of an openly standardized and well-documented message-brokering suite. The Danish OIOXML-program tries to encourage all public service agencies to implement Web Services with a publicly accessible online-repository containing all XML-interfaces in use. Singapore is accepting the challenge of tool-aided service-development with its e-service-Generator. The United Kingdom's Government Gateway is taking a similar approach whereas its multi-purpose character as both web-front-end and message-broker targets at leaving the beaten track. Its usage of a thesaurus for annotating services also seems to be a promising way.

Nonetheless all of these solutions are still limited to a nation-wide service delivery at the utmost and their services largely lack a sufficient annotation with retrievable meta-data. The central challenge will be the enhancement of cross-governmental and trans-national connectivity.

Some EU-sponsored research projects try to live up to these kinds of requirements. Most of them follow a semantics-based approach using multiple ontologies. The outcomes of these projects differ a lot from each other and most of them are still in progress. A final estimation would be premature.

Semantic web services formalisms and supporting tools presented in chapters 3 and 4 were selected based on the criteria of the compliance to the expected Access-eGov functionality, mainly service discovery, composition and invocation. Proposed formalisms and tools lead to two different approaches of implementing the required functionalities. These approaches are summarized in the following table.

	Process oriented	**Service oriented**
Description	Based on workflow for process modelling (i.e. BPEL) used for service choreography and orchestration. Workflow is extended with "light weight" semantic description of services indented for service discovery and composition.	Based on one semantically rich conceptual model which covers all aspects of semantic web services including formalism for process modelling.
Opportunities	• better support for user interaction • rich graphical notation for process model supported by tools - this	• covers all aspects of semantic web services in one conceptual model • strong semantics based on languages which

	can be used for both modelling and user interface • not strictly oriented to web services	combines conceptual modeling with rules • integrated execution environment
Threats	• coping with implementation of semantics (for service discovery and composition) • many tools and formalisms have to be integrated to provide all required functionalities	• has to be extended for traditional services
Formalisms	BPEL4WS, workflow standards + WSDL-S for semantic discovery and composition	OWL-S, WSMO
Tools	METEOR-S, workflow engines, BPWS4J	WSMX, IRS III, OWL-S tools

Although we plan to explore both the approaches, we can conclude that service oriented approach based on common conceptual model, implemented in an integrated execution environment is a more viable approach. From this point of view WSMO and WSMX seem most promising for the Access-eGov Project.

References

[DKOIOEA05]	OIO, "Architecture for e-Government in Denmark" 05/Dec/2005 http://www.oio.dk/arkitektur/eng
[DKOIOWP05]	OIO, "Whitepaper on Architecture for e-Government in Denmark" 05/Dec/2005 http://www.oio.dk/arkitektur/publikationer/whitepaper
[DKOIOWI05]	OIO ISB, "What is OIOXML?" 05/Dec/2005 http://isb.oio.dk/Info/Standardization/OIOXML%20Classes.htm
[DKOISB05]	OIO ISB, "Overview of the Infostructurebase" 05/Dec/2005 http://isb.oio.dk/Info/Welcome+to+the+Infostructurebase.htm?wbc_purpose=Basic&WBCMODE=PresentationUnpublished
[SIAPPSi05]	Government Chief Information Office, "Government to Citizens: The Public Service Infrastructure (PSi)" 01/Sep/2005 http://www.egov.gov.sg/AwardsandAchievements/AchievementsinG2C/Government+to+Citizens+The+Public+Service+Infrastructure+%28PSi%29.htm
[SIPSi05]	Government Chief Information Office, "Factsheet of Public Services Infrastructure (PSi)" 01/Mar/2005 http://www.egov.gov.sg/NR/rdonlyres/1EB34CDA-624E-4F50-97EC-D71521289595/0/PSI___JUNE_02.PDF
[SISWTA05]	Government Chief Information Office, "Networked Government: Common Architectures and Infrastructures" 01/Sep/2005 http://www.egov.gov.sg/PlansandStrategies/e-GovernmentPlans/eGovernmentActionPlanII/NetworkedGov/Common+A

	rchitectures+and+Infrastructures.htm
[UKGG05]	Cabinet Office – e-Government Unit, "Government Gateway FAQ's Version 2.0" 05/Apr/2005 http://www.cabinetoffice.gov.uk/e-government/docs/responsibilities/document_library/pdf/gateway_faqs_v2.pdf
[UKEDT05]	Cabinet Office – e-Government Unit, "EDT (e-delivery) - Government Gateway" 26/Jan/2006 http://www.cabinetoffice.gov.uk/e-government/responsibilities/edt-intro.asp
[UKEGIF05]	Cabinet Office – e-Government Unit, "e-Government Interoperability Framework v6.1" 18/Mar/2005 http://www.govtalk.gov.uk/schemasstandards/egif_document.asp?docnum=949
[UKIPSV05]	Cabinet Office – e-Government Unit, "IPSV-Integrated Public Sector Vocabulary Version 1.00" 10/Feb/2006 http://www.esd.org.uk/standards/ipsv/index.html
[SWSHAP03]	Statskontoret, "SHS Version 1.2 API" 09/Oct/2003 http://www.statskontoret.se/shs/pdf/shs-api.pdf
[SWSHAR03]	Statskontoret, "SHS Version 1.2 Architecture" 09/Oct/2003 http://www.statskontoret.se/shs/pdf/shs-architecture.pdf
[SWSHDO03]	Statskontoret, "SHS Version 1.2 Documentation Overview" 09/Oct/2003 http://www.statskontoret.se/shs/pdf/shs-documentation.pdf
[SWSHPR03]	Statskontoret, "SHS Version 1.2 Protocols"

	09/Oct/2003 http://www.statskontoret.se/shs/pdf/shs-protocols.pdf
[SWSHDI03]	Statskontoret, "SHS Version 1.2 Directory" 09/Oct/2003 http://www.statskontoret.se/shs/pdf/shs-directory.pdf
[SWIFS04]	Statskontoret, "Infra Services – a Swedish Way to Facilitate Public E-services Development" 30/Jul/2004 http://www.statskontoret.se/upload/Publikationer/2004/2004120.pdf
[TEREGV05]	OCC, AIP, Regione Veneto, CoI, HP, Airial, Epektasis, "TERREGOV - Impact of eGovernment on Territorial Government Services" 06/Jun/2005 http://www.terregov.eupm.net/Documents/Deliverables/WorkPackage5:%20Pilot%20activities%20in%204%20Regions/Deliverable_D5.5%20-%20Pilot%20application%20prototypes%20(all%204%20pilots)%20-%20v1/TGV_D5_5_Pilot_Prototypes_v2.zip
[TEREGV06]	HP, MIP, "TERREGOV - Pilot Application Prototypes" 11/Jan/2006 http://www.terregov.eupm.net/Documents/Deliverables/WorkPackage5:%20Pilot%20activities%20in%204%20Regions/Deliverable_D5.8%20-%20Pilot%20application%20prototypes%20(all%204%20pilots)/D5_8_-_Pilot_Application_Prototypes.zip
[ONTOG05]	Apostolou, Stojanovic, Lobo, Thoenssen, "Towards a Semantically-Driven Software Engineering Environment for eGovernment" 2005 http://www.hsw.fhso.ch/ontogov/documents/34160156%20TED.pdf
[ONTOTA05]	OntoGov, "Technical Approach" Feb/2005 http://www.hsw.fhso.ch/ontogov/Index.asp?lang=en&option=techApproach
[ONTOER05]	Stojanovic, Hatziathanasiades, "OntoGov Editor and Report"

	Feb/2005 http://www.hsw.fhso.ch/ontogov/documents/D6-OntoGov%20Ontology%20Management%20System.pdf
[EUPBC05]	Contenti, Mecella, Termini, Baldoni, "A Distributed Architecture for Supporting e-Government Cooperative Processes" 08/Jan/2005 http://springerlink.metapress.com/(aeft4fyq2ow205znhdn5oer1)/app/home/content.asp?referrer=contribution&format=2&page=1&pagecount=12
[EUPBC04]	Contenti, Termini, Mecella, Baldoni, "Supporting Inter-administration Cooperation: The EU-PUBLI.com Approach" 05/Nov/2004 http://springerlink.metapress.com/(z3exav55u5sj2zvntqgoso45)/app/home/content.asp?referrer=contribution&format=2&page=1&pagecount=8