SERVICE COMPOSITION APPLIED TO E-GOVERNMENT

Neil Paiva Tizzo[1], José Renato Borelli[3], Manuel de Jesus Mendes[2], Luciano Lançia Damasceno[3], Aqueo Kamada[3], Adriana Figueiredo[3], Marcos Rodrigues[3] and G. Souza[3].
PUC Minas Poços de Caldas/MG,[2] Universidade Católica de Santos/SP,[3] Centra de Pesquisas Renato Archer, Campinas, Brazil

Abstract: One of the public administration big challenges is the need to integrate services in order to offer a wide variety of new services, more suitable and better designed, that can be electronically accessed in a uniform way. Recently, the Web Service technology appeared with the promise to compose services through the Internet in a simple way. Despite the Web Services advantages, minimal technology independence is desirable at the design of such compositions in order to guarantee and preserve all the efforts invested in the development of services. In this article a Service Composition Management Framework is proposed, that focuses in a technology-independent description. The referred framework is part of a more complete platform, developed for the electronic delivery of Government services.

Key words: e-Government, Web Services, Services Composition, Collaboration, MDA, MOF.

1. INTRODUCTION

The pressure to continuously cut costs and to provide broad and efficient public access to the information has driven governments all over the world to develop electronic government (*e-Gov*) initiatives. There is a convergence among the different e-Gov strategies that points out to integrated and cross-agency services. The collaboration between the several agencies, partners of an e-Gov structure, can provide more suitable services for the potential users (citizens and enterprises).

The São Paulo State Government, Brazil, is investing a great effort in this direction. As an example, it is on the way of installing public services, directly to the society, by electronic means, mainly through the Internet, improving existing presential systems (Poupatempo's). These systems have to integrate the involved databases and applications on the back-office, independent of their locality (municipal and regional), thus saving time, displacements, simplifying processes and avoiding the presentation of paper documents. A first step has been decided with the execution of a project, named eGOIA (eGOIA, 2003), to implement a demonstration system based on the development of a software infrastructure, in order to allow the access of citizens, through the Internet, to integrated public services, at several levels.

The eGOIA project intends to develop a platform to support Internet collaborative systems in the government context (Cardoso, 2004). The authors of this paper are involved in eGOIA and the examples presented come from this real experience.

The set of Web Services (*WS*) standards is a good proposal to provide such collaborative environments. Web Services, as it will better explained in other sections of the paper, are considered as high level components, configured in order to allow the composition of basic services, if possible "on-the-fly", and thus forming new complex services. In the context of e-Gov, several situations exist for an intensive use of this kind of composition tools.

In this paper, a proposal of a Service Composition Management Framework is explained. Due to the great diversity and volatility of composition solutions and standards in the field of Web Services, a technology independent collaboration tool is proposed. In order to reach this goal, the different aspects of collaboration are developed following the concepts of MDA (Model Driven Architecture): services and related metadata are first modeled independently of technology and platform, by means of UML and EDOC. Later on, these platform independent models are mapped to the specific technologies of Web Services.

The remainder of the paper is organized as follows. In section 2 we review several concepts that are to be used in the proposal. Section 3 presents the main ideas that configure the Service Composition Management Framework and discusses some results of a prototype being developed. A realistic e-Gov service composition case study is described in section 4. Section 5 describes related works and section 6 concludes this paper.

2. OVERVIEW OF CONCEPTS

In this section we present the technical foundations that guide the development of the framework, specifically regarding aspects of multi-tier architectures, Web Services stacks of standards, MDA and MOF.

2.1 Multi-tier e-Gov Architectures

Looking at the multi-tier architecture (Frankel, 2003) used in the current generation of e-Gov systems (Figure 1), there are several opportunities for the use of service composition. Such architecture has been considered for the eGOIA project.

For simplicity, three tiers are represented: the user, the business and the persistence tier. The services will be offered through a portal in the Internet, accessed by the citizen through PC-browsers, public points of access, or other type of access channels.

The persistence tier represents the government legacy system, applications and databases *("back-office"),* and is managed by the government agencies. Future evolution, with possible legacies reengineering, brings the opportunity of data services composition, in order to allow a better integration and access by applications from other tiers.

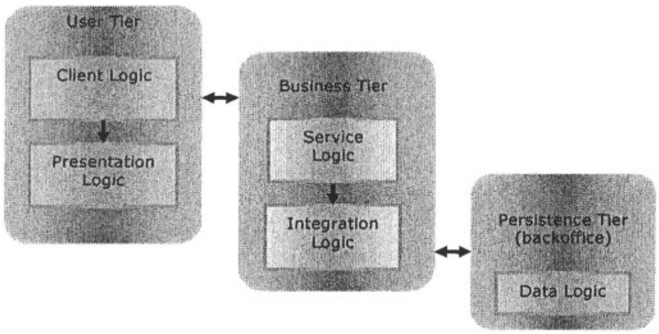

Figure 1. The multi-tiered architecture for e-Gov systems.

The business tier encloses the services and the integration logic. The integration logic is responsible for the unification of data, used by the e-Gov services, and coming from the persistence tier. Otherwise a great diversity of basic services (e.g. form server, authentication, access control, security, accountability services) will be located in it. The composition of these basic services will be used for the expansion of agency capabilities, as well as, for the support of government services.

The service logic includes a great diversity of government services (e.g. health, education services). Its deployment brings again an interesting field for the use of composition techniques, in order to offer more complex services for the users, mainly on a "one-stop" context.

2.2 Web Service Concepts, Protocols and Languages

There are three, at least, related areas speaking about interoperability and integration of applications in the internet. The correspondent standard stacks are represented in Figure 2. All of them share basic aspects (HTTP, XML) (Turner, 2003).

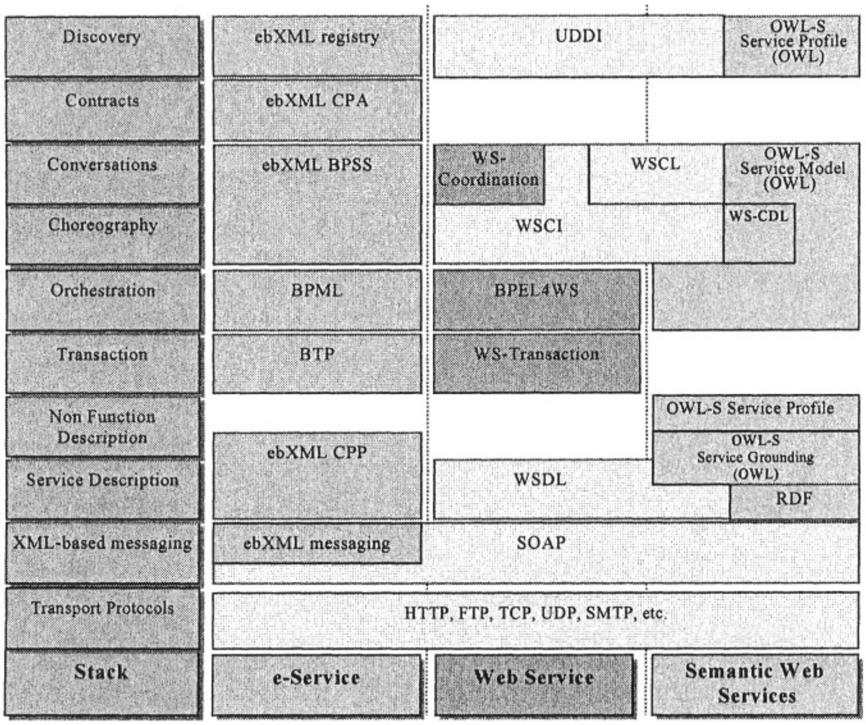

Figure 2. Related stacks of electronic services standards.

The left stack deals with *e-Services* and corresponds to the concepts being developed in the context of B2B (Medjahed, 2003a). e-Services are delivered electronically, through the Internet. Interest in e-Services has been growing continuously in the recent years. Examples of e-Services include software solutions provided by Application Service Providers, and Supply Chain information management networks. A composite e-Service is a group of several e-Services that can be seen as a workflow. A *workflow* is an automated business process that manages the sequence of work activities and the use of appropriate resources associated with the various activity steps. This kind of composition tends to be of static nature and is related to the topic of application integration (EAI). There is a large number of XML-based frameworks for B2B interactions (e.g. *eCO, cXML, RosettaNet, ebXML*). In the Figure 2 some aspects of ebXML are represented.

The right stack, *Semantic Web Services (S-WS),* is a branch of the DAML program (DARPA, 2004) and has been integrated recently in the standards stack of the Semantic Web community. The focus is more in AI agents, and th outside the scope of this paper. Services are agents acting in the Web and the aspects of autonomy are of relevance (W3C, 2002). The correspondent standards are thus not elaborated in the same degree of detail as it happens in the other stacks. The automatic composition of Web Services is slated to play a major role in enabling the envisioned Semantic Web (Medjahed, 2003b). The semantic of Web Services is crucial to enabling automatic service composition.

The stack in the middle deals with *Web Services (WS),* main subject of this paper. WS's can be defined as applications that can be published and consumed as services, by other applications, using Internet standard technologies, such as XML, SOAP, WSDL and UDDI. WS's should be easily composed dynamically with locally developed or externally available services, irrespective of the platform, development language or object model used to implement them, configuring new complex composite services. In this sense they represent an example of SOA (Service Oriented Architecture) and are high granular components, to be joined dynamically. Each composition can involve a great number of services, each one playing a specific role in the collaboration.

Composition may involve several levels of functionality of the so called Web Services stack: conversation, choreography, orchestration and transaction (Turner, 2003). Again the literature is rather confused about these topics, for example, not distinguishing exactly the meanings of choreography and orchestration. The following concepts have been adopted in this paper. *Conversation* deals with initial activities, necessary for the finding of partners (e.g. UDDI), defining roles and, if necessary, negotiating collaboration contracts. This is a field where few standards have yet been defined. *Orchestration* (related to the concepts of workflow or business process) is the description of the flow of data and control involved in the execution of a set of services. Later on in the paper, the use of modeling languages as UML and EDOC is introduced. This will help on clearly stating orchestration as the activities similar to those of defining business processes (workflows). *Choreography* specifies the coordination of some Web Services under the aspect of protocols, defining messages and the sequence of messages exchanged between the parts. This corresponds also to the EDOC concepts of choreography. *Transactions* describe the execution of composite services in an atomic form, i.e., providing rollback mechanisms in case of local failures or service abortions.

In order to better understand this subject a brief description of existing standards is now given. SOAP (Simple Access Object Protocol) handles the transport layer of XML messages among nodes, e.g. over HTTP, SMTP, XML-RPC. WSDL (Web Service Description Language) handles the interface layer by defining the syntax of the I/O, the names of the operations, the error messages. UDDI (Universal Description, Discovery and Integration) handles the registration of services offered by businesses. WSCI (Web Service Choreography Interface) describes the flow of messages exchanged by a WS participating in choreographed interactions with other services. BPEL4WS (Business Process Execution Language for Web Services) provides a language for the formal specification of business processes and business interaction protocols. It supersedes WSFL and XLANG. CS-WS provides the conversation support framework, executing "conversations" that require a more loosely coupled, peer-to-peer, dynamic, proactive model of interaction between the messaging system and business processes. WS-Coordination describes an extensible framework for providing protocols that coordinate the actions of distributed applications. WS-Transaction describes coordination types that are used with the coordination framework in order to implement distributed (e.g. atomic) transactions.

In this paper the set of Web Services standards is considered as the right proposal to provide a collaborative environment in the context of eGovernment. There are several problems related to this architecture as, for example, the volatility of the standards stack. The current WS architecture proposes several layers of functionality, each one containing different solutions (protocols, languages) that strive to become standards. However, the adoption of a possible universal solution for each layer will never be accomplished. Furthermore, new technologies (or even layers) can appear to fill new requirements for specific WS contexts.

Finally, the three approaches (e-Services, WS and S-WS) are not necessarily mutually excluding, and situations may be visualized where they may cooperate. The basic WS standards (e.g. SOAP and WSDL) have been adopted also in the context of e-Services and S-WS.

2.3 MDA and MOF

MDA, Model Driven Architecture, (OMG, 2004) has been adopted as the modeling framework in our research project. MDA provides an open, technology independent, approach to address the problems of productivity, portability, interoperability, maintenance and documentation in software development process, building upon and leveraging the value of established modeling and metamodeling standards. It separates business logic from specific platform technology (e.g. WS standards) creating a new way of developing software in which the platform independent Models (PIM's) turn into the long-term maintainable artifacts. A PIM can be transformed in platform specific Models (PSM's) and code applications, deployed on a variety of platforms (e.g. J2EE, CORBA, .NET, Web Services and future new ones).

In the core of the MDA approach is the Meta-Object Facility (MOF), which is used to manage the models generated during modeling activities. MOF provides a model repository and an abstract language, the MOF Model, which can be used to specify and manipulate metamodels, thus providing support consistency in manipulating models in all phases of the use of MDA. Models and metamodels are *metadata,* information about data. The MOF main objective is to facilitate the access, the management, the processing and the sharing of a great collection of structured and/or non-structured metadata (Kerhervé, 1997). Using MOF is possible to define different languages for modeling different aspects of systems and to integrate models expressed in different languages.

The MOF highlight characteristics are: (i) it is self-defining, i.e., the MOF Model is used to self-describing; (ii) the MOF Model is object oriented; (iii) it uses models to define other models; and (iv) it is based on the four-layer metamodeling architecture. The lowest layer of the architecture, the Information Layer or M0, holds objects and data. The layer above, called the Model Layer (Metadata Layer) or M1 that describes the M0 data, consists of instances of M2 metamodel constructs. The M2 is the metamodel layer consisting of MOF-compliant metamodels, standardized or not. And finally, the M3 or meta-metamodel layer conceptually consists of only one model, called the "MOF Model".

As MOF is platform independent, it will allow mappings to different technologies. At the present time, there are several MOF technology mappings, like: the Java Metadata Interchange (JMI) (JMI, 2002) that defines a set of rules for mapping the elements of a MOF compliant model or metamodel to Java interfaces, allowing the metadata in the repository, represented as Java objects, to be manipulated through the use of them; the MOF-CORBA mapping that defines mapping rules to CORBA IDL enabling the manipulation of the stored models and metamodels as CORBA objects; and the MOF-XML mapping called XML Metadata Interchange (XMI) (OMG, 2004). These mappings produce XMI documents, and Java or CORBA interfaces, to manipulate the models and metamodels stored in the repository.

However, the physical format of how the metadata should be stored is not predefined.

XMI is a mechanism to be used by various tools, repositories and middleware to interchange models and metamodels serialized into XML documents and XML Document Type Definitions (DTDs)/XML Schemas respectively. The import (or export) of XMI documents is a means for exchanging metadata among repositories, living in different business domain.

Currently, OMG is working on a MOF-WSDL mapping that will make possible to expose MOF repositories for Web Services.

3. PROPOSALS FOR SERVICE COMPOSITION

The Service Composition Management Framework *(SCMF),* proposed in this paper, has as main modules the Composition Planning (CP) module and the Composite Service Enactment (CSE) module. These two modules cooperate with two other systems that are being developed by our group, the Modeling System (MoS) and the Metadata Management System (MMS). The functionalities of the SCMF and their interaction with the others are shown in Figure 3.

The Composition Planning (CP) receives the initial request for a service composition and elaborates a plan for it. The Modeling System (MoS) generates models and metamodels and their transformations. The Metadata Management System (MMS) is a MOF based system to support the definition and management of different types of metadata (models and metamodels). The Composite Service Enactment (CSE) is responsible for running the composite service.

CP receives the service request from a client and proceeds to its identification, fetching the necessary information from MMS (step 1). If the composite service is not existent, CP, based on the MMS contents, prepares the master plan to guide the service composition and activates MoS (step 2). MoS consults MMS for the metadata of each service involved (step 3), generates the new composite service metadata and stores it (step 4) in the MMS. Finally, CP requests the service execution (step 5) by the CSE that receives the necessary information from the MMS (step 6).

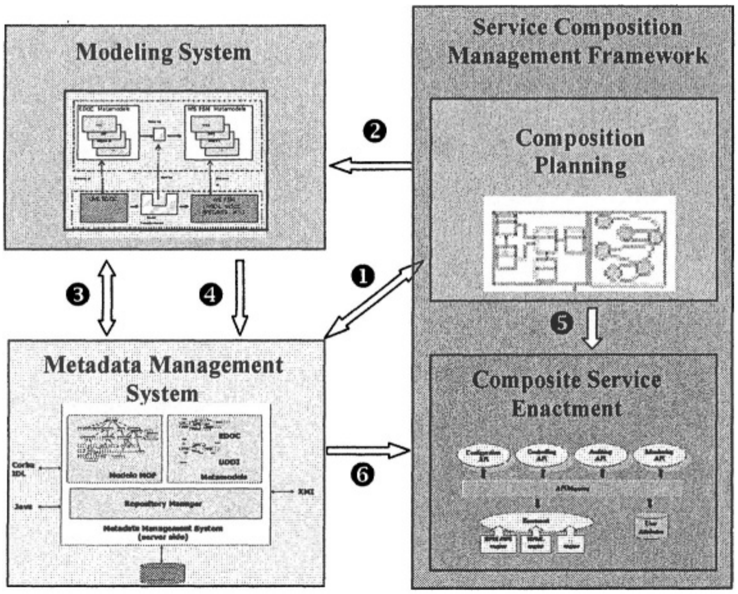
Modeling System
Service Composition Management Framework
Composition Planning
Metadata Management System
Composite Service Enactment
Repository Manager
XMI
Java
Corba IDL
EDOC
UDDI
Modelo MOF
Metamodelo
Metadata Management System (server side)
❶
❷
❸
❹
❺
❻

Figure 3. Interaction between the Composition components.

3.1 Composition Planning (CP)

The CP module owns the main external interface of the proposed SCMF. It is responsible to receive the initial request for a service composition, which shall describe the general characteristics and desired results. Based on the platform resources, rules and constraints, CP analyzes the viability to attend it, and elaborates a composition plan.

Initially, the analysis consists on the existence and availability verification, of the necessary basic services in the platform or elsewhere in the system. This step must obey some concepts of modularity and reusability of the services and there descriptions. Therefore, the CP identifies the units of constituent services in the MMS. Determined the existence and availability of the basic services, the CP analyzes the models, from which the basic services types are instanced, and compares their syntactic and semantic features to determine whether they may me composed. Some restrictions and legal constraints, such as service provider domains and contracts, must be verified. Aspects such as sequence of service execution and services dependencies must be evaluated. Services characteristics such as demand, time of execution, run time resources, priorities, are also considered in the analysis. Finally, as a last criterion, each service utilization weight cost may be used to decide the services selection.

Once the service composition viability is confirmed, the CP generates the contracts between the services providers, which shall define the composition and documents, and a master plan, that will guide the composition implementation by the MoS. Afterwards, the CP must generate all the necessary support and management information to the other modules, in order to promote the composition results as stated in the master plan.

3.2 Modeling System (MoS)

The independence, in respect to the technology used to compose services, is obtained through the development of PIM's (e.g. with EDOC) and their posterior mapping to the different technologies (Web Service PSM).

PIM, in this context, means that the composition will not be defined in terms of specific current languages used to do the Web Service composition (e.g. BPEL4WS, WSCI, BPML, WSTrans). As cited before, there is no settled standard in this area. For example, the functionalities expressed with BPEL4WS can be alternatively reached with the junction of the WSCI and BPML. In this context OMG has issued a RFP for UML Extensions for Workflow Process Definition (OMG, 2004) that requests, in particular, proposals for one metamodel and/or profile, which extends the UML to define workflow processes. An example of a unified metamodel proposal for workflow processes creation, expressed as a UML profile, and its supported software was developed by Soto (Soto, 2002).

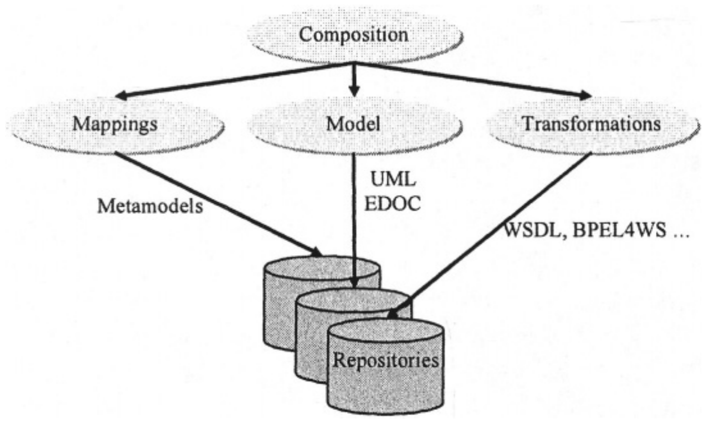

Figure 4. Service Composition Modelling

The different aspects of service composition are first modeled by means of EDOC. Different MDA tools allow, later on, the transformation of these models to technology specific models. In order to produce the model transformations it is first necessary to produce metamodels of WS standards (e.g. WSDL, SOAP, UDDI, BPEL4WS) and to map them to the correspondent EDOC Metamodels (Figure 4).

EDOC is composed of several parts assembled under the so called Enterprise Collaboration Architecture (ECA), each one well defined by a particular metamodel: the Component Collaboration Architecture (CCA), the Entities profile (ER), the events profile (EP), the Business Process Profile (BPP), and the Relationships Profile (RP) (Figure 5). A last one, the Pattern Profile, may also be used for the composition through collaboration patterns previously defined and stored in a library.

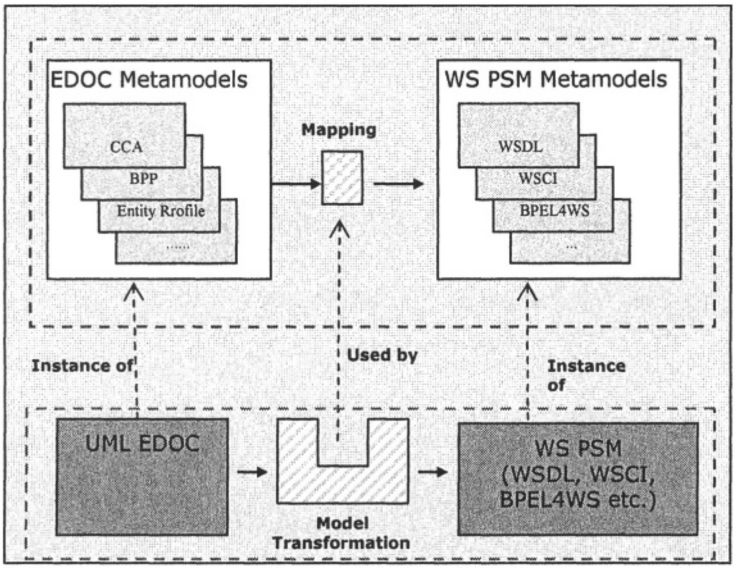

Figure 5. Mappings and transformations.

CCA represents the basic concepts of a service, including activities, ports, artifacts and process and is closely related to WSDL. Their similarities allow the mapping of their metamodels and the transformation from an CCA-PIM to WSDL documents.

CCA goes much further, explicitly defining richer collaboration metamodels, including control mechanisms of choreography and transaction. These mechanisms may be mapped to correspondent WS metamodels of SOAP, WSCI (W3C, 2002) and BPEL4WS. As stated in these specifications, WS choreography concerns the observable interactions of services with their users. A choreography description is a multi-party contract that describes the behavior across multiple WS, through the presence of messages that are exchanged and their sequence. EDOC contains a recursive component model, which in turn can express the choreography of the component's data exchange with the outside world and through recursion, with its inside world.

The Entity Profile of EDOC is used to generate new PIM Data Models, for example models of messages being exchanged in the choreographies. There is yet no similar standard proposed in the context of WS's. This could be relevant, for example, to allow the integration of existent Data Base accessed by specific basic services, which participate in the composition. Based on the concepts of CWM (CWM, 2003), proposals are discussed by the authors in another paper (Figueiredo, 2004)

The Business Process Profile (BPP) provides modelling concepts that allow the description of business processes in terms of a composition of business activities, selection criteria for the entities that carry out these activities, and their communication and coordination. These are the concepts needed for the orchestration modelling of Web Services. For example, developing a metamodel of BPEL4WS will allow later on the mapping to the BPP metamodel and though transformations of BPP models to BPEL4WS descriptions.

With these proposals we are, in reality, constructing a WS–Metamodel, formed by different metamodels, as is already the case in EDOC. Using MOF (see section below) based rules we assure a coherent construction and storage. But EDOC is very rich in its semantic and includes several modelling mechanisms not yet included in the WS standards.

3.3 Metadata Management System

A Metadata Management System (MMS), also referred as a repository system, is a key system to the support of the composition framework proposed in this work. We took the approach to develop the MMS based on the modeling framework as defined by OMG, i.e., using MOF and XMI technologies.

The motivation for defining a MOF-compliant system is to facilitate the implementation of a repository system for models and metamodels from different domains. Furthermore, XMI offers a standard way for exchanging models between our and other tools environments.

The metadata management system is a central component of an e-Gov system (Figueiredo, 2004). In that work, an extensive use of standard metamodels is proposed to promote the integration of distributed and heterogeneous legacy data.

In the context of this work we foresee the MMS being used to support the definition and management of different types of metadata:

- EDOC metamodels:
- EDOC models, instances of the EDOC metamodels;
- Transformation rules metamodel: rules models are used to map for example, a PIM to a PSM;
- WS-Metamodels, for example
  - UDDI metamodel: the UDDI metamodel will supply details of the offered services. In order to support details of the composed services, the UDDI metamodel will be extended;
  - WSDL metamodel: instances of the WSDL metamodel will describe simple services and composed ones.
  - BPEL, WSCI ,etc metamodels
- WS- Models (WSDL, BPEL, WSCI,etc)

The server-side of MMS is presented in Figure 6. Its main components are the repository manager and a database mechanism to persistent metamodels and models. The repository manager provides services to modeling, retrieving and managing the artifacts in the repository. Additionally, it provides functions specific to a repository system, such as checkout/checking, version control, configuration control, notification and workflow control. A set of tools, such as a graphical editor, a compiler and a modeler viewer, in the client-side complete the MMS system.

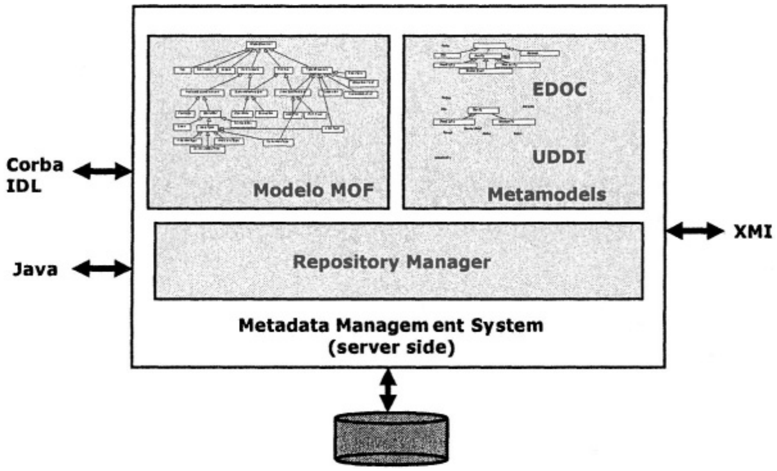

Figure 6. Metadata Management System Architecture

3.4 Composite Service Enactment

The Composite Service Enactment (CSE) is responsible to run the composed service. Depending on the language/protocol used to describe the composition, the CSE connects one specific engine for its execution. New engines can be connected to the module. This flexible architecture allows the adoption of new technologies or standards (Figure 7).The CSE is build by the following principal components: User's API, API Mapping, Enactment and User's Attributs data base.

The User's API functionality is aligned to that defined in the Workflow Reference Model described by the Workflow Management Coalition (WfMC, 2004). Below, a brief description:

- **Configuration:** defines how the Enactment component will run, i. e., the deployment configuration, as well as the service priorities and security access control;
- **Controlling:** supplies the API for administration and interaction of the work list managed by the Enactment. It allows, for example, search for work in progress, the examination of the composition execution status, initiation, pause, cancellation or stop;
- **Auditing:** the date, time, person steps etc, are recorded in a history repository. The auditing interface provides access to this data;
- **Monitoring:** reporting and analysis such as the total work accomplished.

The API Mapping receives the request from the User's API and maps it to the specific engine that is being used. So, several different engines are accessed by the same API. The transformation rules that permit the mapping are stored in the MMS. When a new engine is linked to the module, its associated transformation rules need to be processed and stored at the MMS.

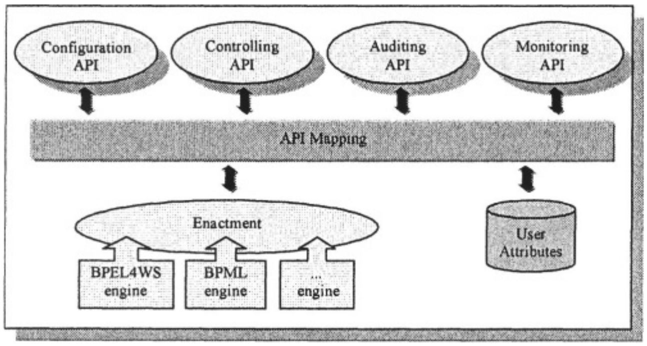

Figure 7. Composite Service Enactment Module

A User's Attribute Data Base stores the configuration for each user. When a request arrives, the API Mapping consults it to do the proper mapping for each one. The data base also stores the access policies. Depending on these policies, the user (client application) may be able to access the entire or only part set the functionalities correspondent to the controlling, auditing and monitoring API's.

The Enactment is responsible to connect and control the specific engine for each application. It does the communication between the engine and the API Mapping.

When an event is generated by one engine, it is captured by the Enactment that resends it to the API Mapping. At this time, the mapping needs to be done at reverse order, i.e., the specific engine event is mapped to the User's API. Again, the API Mapping accesses the MMS to do it.

3.5 Implementation Aspects

Figure 8 illustrates an e-Gov platform prototype being implemented in our project. The users will access the services through a portal site that will ensure an uninterrupted and easy communication with the service provider.

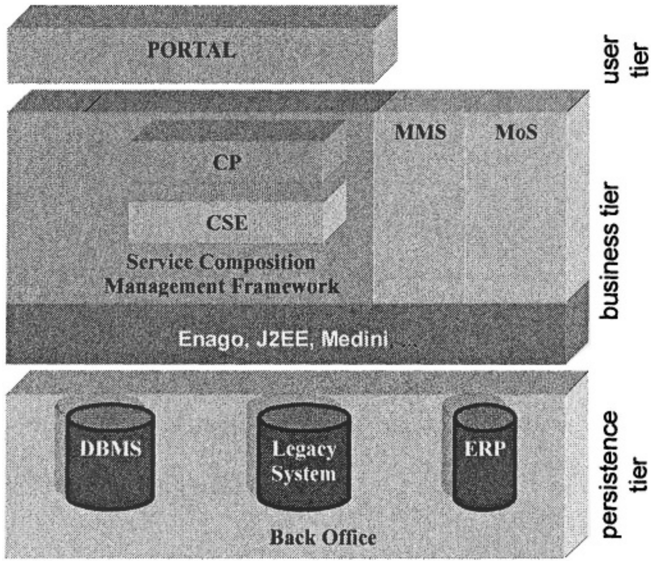

Figure 8. Implementation aspects.

In the first version of the prototype, the framework proposed will reside on top of Enago (Figure 8), an open service platform developed by the Fokus Institute (Fokus, 2004). Enago provides a secure and uniform service access and management environment, across different administrative and technological services. The main focus of Enago is to enable the integration, composition and management of existing and emerging application services based on different access technologies and different service technologies.

The Medini tool suite, also developed by Fokus Institute, is the MoS adopted in this first version. Medini provides a modeling infrastructure based on the EDOC and Enago metamodels. PIM models instantiated from the EDOC metamodel are stored in a MOF-compliant repository and, subsequently, transformed to Enago PSM models and Java code. The Medini tool suite (Kath, 2003) provides transformations tools between a PIM written in EDOC to PSMs for Enago Platform and J2EE. The transformations are supported by MoS.

The development of the Composition Planning module is still in the phase of conception. The format for the service composition request, that shall describe the characteristics and the desired results of the service composition, is still undefined and it is conceived to attend as well the general users' requisitions, through a web portal, as also the planning of e-government internal services, through an e-Gov framework.

The MoS is in an advanced stage of implementation and now the activities are focusing the metamodels development and needed mappings, in the WS context described.

The Composite Enactment Service module is under implementation. The BPWS4J (IBM, 2002) engine has been integrated in it and the next step is to connect other engines. Further, we intend that the Enactment will be able to support collaborations between WS's that use different choreography languages.

4. CASE STUDY: VEHICLE EXTRACT

As an example to show the problem, we propose a real scenario to modify a vehicle ownership. The already existing basic services in the example are the following (VID is a vehicle ID and CID is a citizen ID):

1. *Vehicle service:* supplies vehicle data;
   Interface 1:*search (in: VID, out: ownerID);*
   Interface 2: *setOwner (in: VID, CID, out: transfer certification);*
2. *Citizen service:* supplies citizen data;
   Interface 1: *search (in: CID, out: citizen data);*
3. *Tax service:* government taxes;
   Interface 1: *search (in: VID, out: tax data);*
4. *Fine service:* government penalties;
   Interface 1: *search (in: VID, out:fine data).*

In order to present to the user more suitable services, the *vehicle extract* is created. It returns an extract and is composed by the basic services above mentioned. It has the following interface: *extract (in: VID, out: report).* It starts when the user supplies the VID. The tax, fine and vehicle basic services are accessed in parallel; the vehicle service returns the "owner ID" that is the equal to the CID code used by the citizen service. The Extract Assembler is a new service that was created to supply the need of this composition. It assembles the data returned form the other services and sends it back to the user. In this example the vehicle service interface 2 was not used (Figure 9).

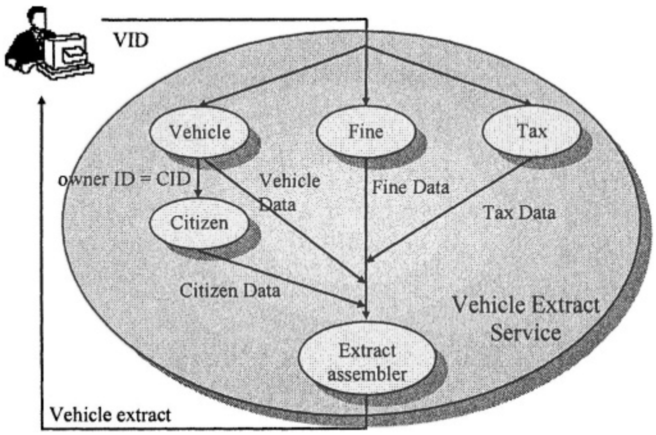

Figure 9. Case study: vehicle extract.

5. RELATED WORK

In this section, we discuss some initiatives for composition of B2B e-Services, Web services and Semantic Web Services, which related to our research.

Many software industries, such as IBM, Sun Microsystems, Microsoft, BEA systems, HP and Oracle are currently working on implementation of B2B e-Services platforms. The common key technologies for almost all of them are J2EE components, XML and the use of a workflow engine. The IBM WebSphere integrates support for Web service standards such as SOAP, UDDI, and WSDL. Additionally, it provides distributed transaction support for major database systems. The Sun ONE iPlanet is the core of Sun ONE platform. It includes a stack of products that allow the creation, deployment, and execution of B2B e-Services.. Microsoft .NET embraces the concept of Web services to enable B2B e-Services interaction. It consists of key elements, which provides the standard based tools for SOAP, WSDL, and UDDI.

In (Orriëns, 2003) a tool for service composition specification, construction and execution is presented. This tool intends to support the development and delivery of composite services in a coordinated and effectively reusable manner, turning aspects related to business process modeling languages, such as BPEL4WS, more flexible and less complicated. It also focuses on the modularity and reusability of the existent services in the platform. A Service Composition Specification Language (SCSL) is developed, where activity, binding, condition and composition type constructs are defined, among others, in order to support the web service composition.

In the Semantic Web Services area the major initiatives use ontologies and software agents to deal with Web services composition. The autonomous Semantic Web Services (Paolucci, 2003) is a mechanism that tries to bridge the gap between the Web Services infrastructure and the Semantic Web. It is based on the DARPA Agent Markup Language for Services (DAML-S), which promises capabilities for discovering, invoking, composing and monitoring Web services. The ontology-based framework for the automatic composition of Web services (Medjahed, 2003b) starts from high-level declarative descriptions and generate composite services by defining formal composition safeguards through the use of composability rules. These rules compare the syntactic and semantic features of Web services to determine whether two services are composable or not. In the agent-based multi-domain architecture (Maamar, 2003) a software agent-based approach is presented, that supports the interleaving of Web Services composition and execution.

6. CONCLUSIONS AND FUTURE WORK

In this work, which is concerned with the requirements on WS dynamic composition, we present a framework to discovery and compose WS's and to invoke and execute the new composite service. This emphasizes the coexistence and seamless interoperation on varieties of software components, which have been deployed based on legacy applications or on emerging service standards.

We propose a Framework for Web Services composition that is part of an e-Gov platform Prototype. The system uses a metamodeling infrastructure and the MDA concepts in order to be independent of composition technologies as conversation, choreography, orchestration and transaction.

The WS area is still very volatile. The great advantage of the proposed system is that it provides platform independence, avoiding the obsolescence caused by the appearance of new technologies. Otherwise, it contributes to solve the problem of applications heterogeneity and improves their portability among different platforms.

The proposed Framework is under development. Besides the proof of concepts, the prototype will be used to test performance aspects, in particular, the practical use of the framework in the context of dynamic composition. Nowadays, the more promising solutions for dynamic composition are being proposed by the community of Semantic Web Services, with strong focus on "automatic composition tools". In this context the semantic aspects are overwhelming. Our proposal goes in a similar direction for Web Services, with the use of metadata concepts.

A further issue, that needs to be addressed, is the incorporation of Business Rules, due to the need to express logical statements, e.g. preconditions and post conditions, or rules to describe dependencies between service elements. In this sense, one of the challenges is to integrate concepts, technologies and tools coming from both Semantic Web Services and Web Services worlds. There are many aspects and problems that have to be considered, such as the mapping between ontology concepts and metadata software engineering concepts, that are not always straightforward.

REFERENCES

Cardosol J. L., et al., 2004, Implementing Electronic Government: the eGoia project, EULAT Workshop on eGovernment and eDemocracy, May 2004, Chile.

CWM, 2003, Common Warehouse Metamodel (CWM) Specification - Version 1.1, Volume 1, OMG Document Number: formal/03-03-02; 2003.

DARPA, 2004, The DARPA Agent Markup Language Homepage, (February, 2004); http://www.daml.org/index.html.

EDOC, 2001, A UML Profile for Enterprise Distributed Object Computing, Joint Final Submission, Part I, Version 0.29, 18 June 2001; http://www.omg.org/.

eGOIA, 2004, Electronic Government Innovation and Access, EU-@LIS Project (September, 2003); http://www.egoia.info/.

Figueiredo, A. C., et al., 2004, Metadata Repository Support for Legacy Knowledge Discovery in Public Administrations, In: 5th Working Conf. on Knowledge Management in eGov, May 2004, Austria.

Figueiredo, A. C., et al., 2003, Using Metamodels to Promote Data Integration in an e-Government Application Scenario, In: Third I3E IFIP Conference, September 2003, Guarujá, SP, Brazil.

Fokus, 2004, Fraunhofer Institute for Open Communication Systems (February, 2004); http://www.fokus.fraunhofer.de/.

Frankel, David S., 2003, Model Driven Architecture - Applying MDA to Enterprise Computing, Wiley Publishing, Inc, 2003.

IBM, 2002, IBM Business Process Execution Language for Web Services Java Runtime (April, 2004); http://www.alphaworks.ibm.com/aw.nsf/reqs/bpws4j.

JMI, 2002, Java Metadata Interface (JMI) Specification, version 1.0, JSR40 – Java Community Process (June/2002); http://jcp.org/aboutJava/communityprocess/final/jsr040/ index.html

Kath, O., Born, M., 2003, An Open Modeling Infrastructure, Fokus Institute. (January, 2004); http://modeldrivenarchitecture.esi.es/pdf/04-2_Born_Kath.pdf.

Kerhervé B., Gerbé O., 1997, Models for Metadata or Metamodels for Data?, Proceedings of 2nd IEEE Metadata Conference.

Medjahed, B., et al., 2003a, Business-to-business interactions: issues and enabling technologies, The VLDB Journal (2003) 12: 59–85 / Digital Object Identifier (DOI) 10.1007/s00778-003-0087.

Medjahed, B., et al., 2003b, Composing Web services on the Semantic Web, The VLDB Journal (2003) 12: 333–351 / Digital Object Identifier (DOI) 10.1007/s00778-003-0101-5.

OMG, 2000, UML Extensions for Workflow Process Definition – Request for Proposal, OMG Document Number: bom/2000-12-11; 2000.

OMG, 2004, Object Management Group, (March, 2004); http://www.omg.org/.

Orriëns, B., Yang J., Papazoloug, M. P., 2003, SerciveCom: A Tool for Service Composition Reuse and Specialization (April, 2004); http://maximus.uvt.nl/sigsoc/pub/Orriens %20et%20al%20- %20ServiceCom%20-%20A%20tool%20for% 20service% 20compo sition%20reuse%20and%20specialization.pdf

Paolucci, M., Sycara, K., 2003, Autonomous Semantic Web Services, IEEE Internet Computing (September, 2003); http://www-2.cs.cmu.edu/~softagents/papers/Intemet Computing.pdf

Soto, J. A., 2002, Uma Abordagem Unificada para Modelar Processos de Workflow e seu Software de Suporte. Unicamp, Brazil, 2002.

Turner, M., Budgen, D., Brereton, P., 2003, Turning Software into a Service, In: Computer, Edited by IEEE Computer Society, October 2003, pp. 38-44.

W3C, 2002, Web Services Architecture Requirements, W3C Working Draft 29 April 2002 (April, 2004); http://www.w3.org/TR/2002/WD-wsa-reqs-20020429.

WfMC, 2004, Workflow Management Coalition (February, 2004); http://www.wfmc.org.