International Journal of Web Services Research

October-December 2008, Vol. 5, No. 4

Table of Contents

Interoperability Among Heterogeneous Services: The Case of Integration of P2P Services with Web Services

Aphrodite Tsalgatidou, National and Kapodistrian University of Athens, Greece

George Athanasopoulos, National and Kapodistrian University of Athens, Greece

Michael Pantazoglou, National and Kapodistrian University of Athens, Greece

ABSTRACT

Service-oriented computing (SOC) has been marked as the technology trend that caters for interoperability among the components of a distributed system. However, the emergence of various incompatible instantiations of the SOC paradigm, e.g. Web or peer-to-peer services (P2P), and the divergences encountered within each of these instantiations state clearly that interoperability is still an open issue, mainly due to its multi-dimensional nature. In this paper we address the interoperability problem by first presenting its multiple dimensions and then by describing a conceptual model called generic service model (GeSMO), which can be used as a basis for the development of languages, tools and mechanisms that support interoperability. We then illustrate how GeSMO has been utilized for the provision of a P2P service description language and a P2P invocation mechanism which leverages interoperability between heterogeneous P2P services and between P2P services and Web services.

Keywords: generic service model; grid services; heterogeneous services; peer-to-peer (P2P) services; P2P service description; P2P service invocation; service-oriented computing; Web services

INTRODUCTION

The software community has been confronted with the issue of interoperability between software components during the last couple of decades. Various approaches such as object and component based approaches tackle this problem in various ways. However, they have not yet managed to provide a widespread solution that enables the interoperation of diverse components developed by different providers, in multi-vendor platforms (Medvidovic, 1999). Service oriented computing (SOC) emerged as

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

an evolutionary step of object and component based approaches, with the promise to support the loose coupling of system parts thus providing agility, flexibility and cost savings via reusability and interoperability. However, the existence of many heterogeneous types of services and the vast amount of existing and emerging standards still constitutes a major obstacle towards interoperability, as it will become apparent in the following.

The most well-known instantiations of the service-oriented computing paradigm are Web services (Christensen, 2001) and Grid services (Czajkowski, 2004). However, other types of services such as Peer-to-Peer (P2P) services, see for example the JXTA services (Gong, 2001), are currently gaining momentum. All these types of services are usually built on top of XML standards (Bray, 2004) and other proven communication protocols such as HTTP (Fielding, 1999) and TCP/IP (Tanenbaum, 2003). The establishment of a set of common characteristics for these service types including properties such as self-description, internet accessibility and message-oriented communication has been one of the main research topics in this area and existing results comprise models such as the ones that have been specified by W3C in (Booth, 2004), by Werner Vogels in (2003), and by Karl Czajkowski, et al. in (2004). These features along with the use of XML standards (Bray, 2004) provide an infrastructure that promises to leverage interoperability among the components of a service-oriented system.

Nonetheless, although existing SOC instantiations, provide for a basic infrastructure that tackles interoperability, they still do not fully address it. This is mainly due to the multidimensional nature of interoperability, as it has also been noted in (Fang, 2004; Burstein 2005). Furthermore, the multiplicity of continuously emerging service types, see for example Sensor services (Gibbons, 2003) or UPnP services (Newmarch, 2005), is further aggravating the problem, as, albeit these service types share some common characteristics, they adhere to incompatible models and standards and rely on distinct platforms and middleware to perform their basic activities. It is worth noting here that, the interoperability problem exists, not only between services of different types, but also between services of the same type. See for example the area of P2P services, where there are no common standards or reference models; existing P2P platforms, e.g. JXTA (Gong, 2001), Gnutella (Ivkovic, 2001) or Edutella (Nejdl, 2001), use proprietary advertising, discovery and invocation mechanisms, thus hindering interoperability between these P2P services. Another example is the area of Web services, where, despite the use of standard protocols, there are still interoperability problems between Web services. Efforts such as those undertaken by WS-I (Ballinger, 2004) try to tackle the latter problem; specifically, WS-I has provided a basic interoperability profile for addressing some of the interoperability problems encountered between Web services; such an effort foregrounds the need for addressing the problem in various dimensions. However, there are still a lot of open issues which remain to be solved.

This article provides a solution on Interoperability between P2P and Web services. The provided solution is based on a generic framework which provides for interoperability between any type of service. This generic framework has been developed in the SODIUM project[1], which employs a unified approach towards the interoperability of heterogeneous service types. One of the advantages of this unified interoperability approach is that it allows the various service types to retain their own traits and characteristics. In this article we present: (i) the foundation of the SODIUM work which comprises a generic service model (GeSMO) that constitutes the underlying basis for the development of appropriate software for supporting service interoperability and, (ii) a specific solution that has been provided for the integration of P2P services with Web services which has been utilized in the SODIUM and SeCSE[2] projects.

The provided interoperability solution between Web and P2P services is described in detail in the rest of the article which we have structured as follows: We first present a moti-

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

vating scenario which demonstrates the need for interoperability of heterogeneous services. Then, we examine interoperability among heterogeneous types of services from different dimensions, with an emphasis on Web and P2P services (see *section "*INTEROPERABILITY DIMENSIONS*"*); the issues that need to be addressed in order to facilitate the integration of heterogeneous services are identified in the *subsection "*Interoperability Concerns for Heterogeneous Services*"*. In the sequel, we present a generic service model (GeSMO) that takes into account the identified interoperability issues. In particular the provided constructs catering for the description of P2P services are presented in the subsection *"*GeSMO Extensions for Peer-to-Peer Services*"*. Then, in section *"*A Peer-to-peer Service Description Language*"* we describe how GeSMO and its extensions for P2P services have been utilized for the provision of the Peer-to-peer service description language (PSDL), which is an extension of WSDL (Christensen, 2001) and supports the description, discovery and invocation of P2P services. An example of a PSDL description document for the instant messaging service that is used in the motivating scenario is presented in the subsection "A PSDL Example". Then, section *"*JXTA Service Invocation Mechanism*"* illustrates the mechanism that has been built to support the invocation of JXTA P2P services. This mechanism utilizes JXTA service descriptions in PSDL so as to facilitate their seamless invocation. Next, in section "Application Case Study", we exemplify how P2P services providing location information have been integrated with Web services providing map information, for the implementation of a part of the motivating scenario. Finally, we compare our work with some other related approaches and we conclude this article with a discussion on open issues and on our future work plans.

MOTIVATING SCENARIO

Our motivating scenario is in the area of Crisis Management. This scenario was provided by LOCUS (www.locus.no), a leading supplier of mobile and other types of applications in the Scandinavian markets for transportation, emergency, security and the military. Actually, this motivating scenario was used in one of the SODIUM pilot applications which demonstrated the feasibility and usefulness of the provided solutions.

In the crisis management area a common task of paramount importance is determining how to get to a crisis location as fast as possible, as time is a critical factor. For example, in case of an accident where people are critically injured, it is imperative to reach these persons with the appropriate equipment within minutes, as if the injury causes lack of oxygen to the brain for 3-5 minutes, brain cells will start to die and after approximately 15 minutes there will be permanent damages. Therefore, it is vital that properly equipped ambulances and other rescue units are within a 15-minutes range at all times and places. In most cases this requirement is difficult to be satisfied due to the vast set of parameters such as accident/injury probability, population density and composition, accessibility, time of day/week/month/year, weather conditions, hospital locations and many others, which need to be considered.

It is thus vital that a crisis management system interoperates with various types of heterogeneous systems in order to incorporate the required information. LOCUS decided to use a service-oriented approach for the integration of all these types of systems, since there are a lot of available heterogeneous services which offer information useful to this crisis management application. Such services are:

- **Web services:** providing weather information such as temperature and precipitation, traffic conditions from roadside speed sensors and video surveillance cameras,
- **P2P services:** providing information about the locations and status of emergency vehicles or messaging facilities to the emergency vehicles with reposition message commands.
- **Other types of services** such as grid services providing driving route calculations, historical incident information and

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

"response range" calculations based on current positions and conditions.

Figure 1 presents an example of a workflow based on this scenario, the tasks of which can be satisfied by various types of services[3]. As it can be seen in this figure, first we need to get information about the person who is calling and identify the accident location; see tasks *GetCallerInfo* and *GetCallerPosition*. Then, we need to identify the closest to the accident location, properly equipped ambulance; see task *GetBestSuitedAmbulance*. Subsequently, we need to calculate the shortest route to the accident location and to draw the route from the ambulance's current location to the accident location on a map (see related tasks *DetermineShortestRoute* and *GetMap*) which needs to be sent to the ambulance (see task *InformAmbulance*).

The services satisfying these tasks, do not need to be developed from scratch; instead the LOCUS company would like to utilize existing services (mentioned before) which have either been developed in-house or are provided by trusted partners. For example, a Web service can be used to determine which ambulance is closest to the place of the accident, while a P2P instant messaging service can be used to satisfy the last task of the workflow which requires to send a map to the ambulance with the appropriate route information.

Therefore, the integration of heterogeneous services is of paramount importance, in order to satisfy the needs of service-oriented applications such as the above.

INTEROPERABILITY DIMENSIONS

In this section we briefly recap the results of current research on service interoperability and then we present our approach. Thus, according to the majority of the proposed interoperability

Figure 1. Composition example for crisis management scenario

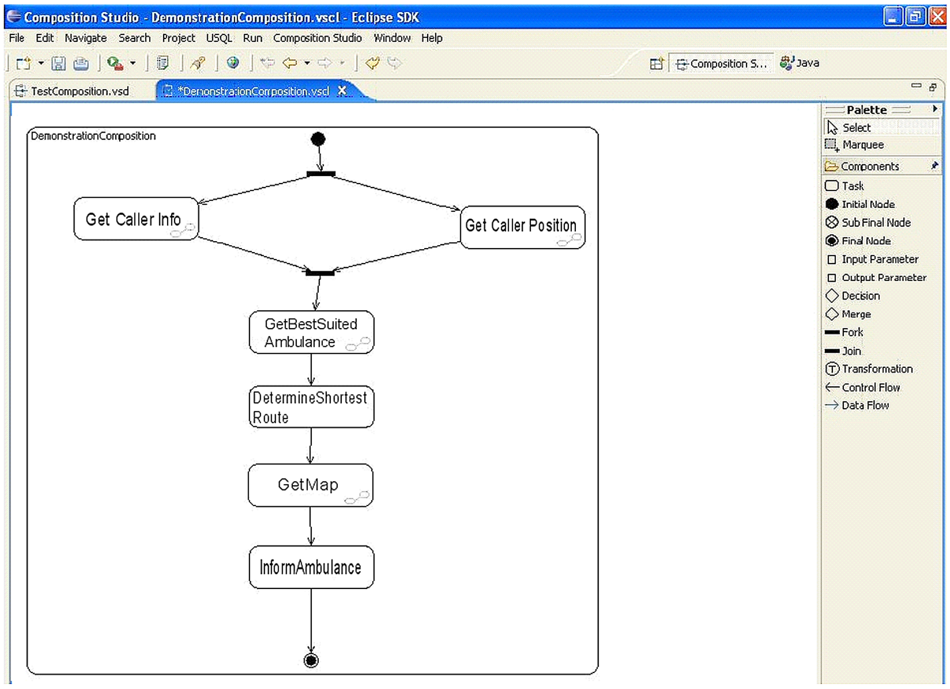


Figure 2. Service interoperability dimensions

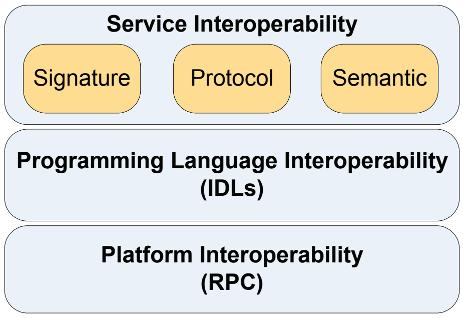
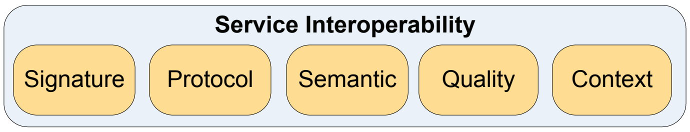

Service Interoperability
Signature
Protocol
Semantic
Programming Language Interoperability (IDLs)
Platform Interoperability (RPC)

Figure 3. Extended set of interoperability dimensions



models (Fang, 2004; Vallecillo, 2000; Murer, 1996), service interoperability has been divided into the signature, protocol and semantic levels. A classification scheme for interoperability that considers the evolution of distributed computing from the emergence of RPC or component models, such as DCOM (Microsoft, 1996) and EJBs (DeMichiel, 2006), to CORBA (Siegel, 1996), has been proposed in (Vallecillo, 2000) and it is presented in Figure 2. As we can see in this figure, the problem of interoperability is divided into three levels: service level, programming language level and platform level. The service level in particular is further decomposed in three interoperability dimensions i.e. *signature*, *protocol* and *semantic* dimensions.

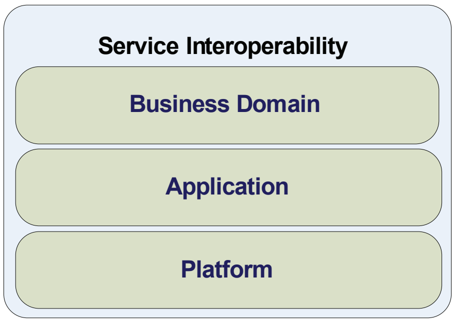
Extensions to these three dimensions of the service interoperability level have been provided in (Strang and Linnhoff-Popien, 2003) and in (Ruiz et al, 2000). In particular, Strang and Linnhoff-Popien added a context dimension to service interoperability which is of high importance to context-aware applications, where Ruiz et al proposed the addition of the *quality* dimension; see Figure 3.

Each of these dimensions describes specific interoperability concerns to be tackled when integrating two service-oriented systems. These concerns are briefly presented below:

- The *signature* dimension addresses the interface definition conformance. This includes the operations, types and order of parameters of a service interface as well as standards such as the interface definition languages (IDL).
- The *protocol* dimension addresses the order in which the methods of a service are invoked. The Web Service Choreography Description Language (WS-CDL) (Kavantzas, 2005) can be seen as an effort to resolve this interoperability problem.
- The *semantic* dimension addresses the problem of common understanding

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 4. Additional service interoperability dimensions



between service providers and service consumers. This problem can be tackled through ontologies and semantic service description frameworks such as OWL-S (Martin, 2004), WSMO (Bruijn, 2005), SAWSDL (Farrell, 2006) or WSDL-S (Akkiraju, 2005).

- The *quality* dimension addresses the conformance of the quality requirements of a service consumer and the quality properties offered by the service provider. Solutions to this issue can be provided by description frameworks such as WS-QoS (Gramm, 2004), or WS-Policy (Bajaj, 2006).
- The *context* dimension refers to the conformance between the context representations used by service providers and the context representations requested by service clients. Context-level interoperability is important when employing services in ubiquitous computing environments. This problem could be tackled through the use of common ontologies and ontology frameworks e.g. RDF (Beckett, 2004), OWL (McGuinness, 2004) or via the use of context description frameworks such as ConteXtML (Ryan, 2005)

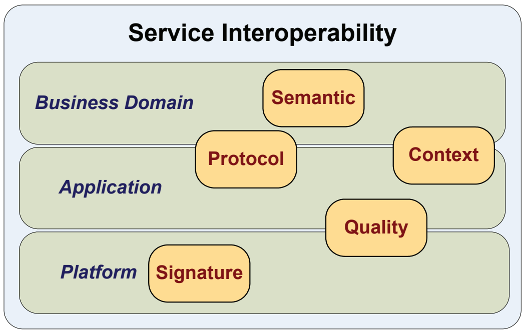
The aforementioned service interoperability dimensions (Figure 3) are the most commonly used. We argue that an additional set of interoperability dimensions needs to be defined at the service interoperability level, orthogonal to the aforementioned dimensions, in order to address interoperability from a global perspective (see Figure 4), providing in this way a more complete view of service interoperability.

The introduced orthogonal dimensions of service interoperability are: the business domain, the application and the platform dimensions. These dimensions have been layered in a top-down manner, starting from the most abstract system concepts related to business domain down to implementation specific concepts such as platform specific ones. Specifically, the introduced dimensions are as follows:

- **Business Domain:** Interoperability at this dimension represents the ability of two business systems to interoperate. This includes sharing of common domain concepts and processes which could be described using various standards and protocols such as RDF (Beckett, 2004), OWL (McGuinness, 2004), OWL-S (Martin, 2004), WS-CDL (Kavantzas, 2005), and ebXML BPSS (Dubray, 2006).
- **Application:** Applications are specific implementations of parts of business domains. Thus, application interoperability represents the ability of two specific business system implementations to interoperate. This includes the use of compatible data structures, functionality and orchestrations that may be described using standards such as WSDL (Christensen, 2001) and WS-BPEL (Alves, 2006).
- **Platform:** Platform interoperability represents the ability of the middleware underlying two or more applications to interoperate. This includes features such as the use of compatible data type representations (e.g. real numbers having the same accuracy and same format), interface specification mechanisms (e.g. interface definition languages) or architectural styles (e.g. use of message-oriented communication styles). As it can be seen, this dimension differs from the platform interoperability

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 5. An integrated view of service interoperability dimensions



layer appearing at the bottom part of Figure 2 in that it addresses both programming language and platform interoperability dimensions.

We believe that the utilization of a multi-viewpoint approach facilitates the clearer assessment of system interoperability. Thus, the dimensions depicted in Figure 3 take an internal look on the aspects that need to be considered when dealing with the integration of two systems, whereas the dimensions in Figure 4, which are orthogonal to the former ones, represent a system architect's 'coarse' point of view on the levels affected by the integration. Thus, we can integrate these two sets of dimensions into a single framework. Figure 5 illustrates this integrated view of service interoperability dimensions along with the associations between the concepts of the two individual frameworks. Such an integrated view facilitates the classification of the interoperability concerns for service interoperability from two distinct points of view.

As it can be seen in Figure 5, the *semantic* and *signature* interoperability dimensions are clearly mapped to the orthogonal dimensions of Business Domain and Platform respectively; the *protocol and context* dimensions are shared between the dimensions of Business Domain and Application and Platform, while the *quality* dimension is shared among the Application and Platform dimensions. In order to better explain these mappings, let us consider the potential interoperability problem when integrating two systems that implement two incompatible processes (i.e. processes with incompatible choreographies). Such an incompatibility is at the *protocol* level. This problem may exist either due to the fact that the two systems support different and thus incompatible business processes (in which case we have an interoperability problem at the Business Domain level), or because their developers have selected different algorithms for their implementation (in which case we have an interoperability problem at the Application level). Similar arguments may be provided for the rest of interoperability dimensions.

There are also other frameworks such as the one presented in (Nezhad, 2006) focusing on Web service related interoperability. This framework leverages a different point of view for classifying Web service interoperability problems and hence, incorporates a different set of dimensions compared to the ones presented in Figure 5. Although this classification framework has been constructed to support the categorization of Web service related interoperability issues, it can also be used for the classification of interoperability issues of other types of services as well. Thus, its dimensions can easily extend the ones of our integrated framework without introducing any inconsistencies, since they have been conceived from a different point of view. However, we believe that there is no need to extend the dimensions of our framework in Figure 5, since these are generic enough to cater for the identification

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

of all interoperability concerns that are also addressed in (Nezhad, 2006).

In the following, we present how the integrated service interoperability framework depicted in Figure 5 can be used for the classification of interoperability problems which arise when integrating heterogeneous services.

Interoperability Concerns for Heterogeneous Services

As it has been mentioned in the introduction, contemporary instantiations of the SOC paradigm, such as Web and P2P services, are ruled by different models, protocols and standards. The range of discrepancies and diversities among services spans across all of their aspects, such as description and discovery mechanisms, quality characteristics or service provision platforms. Based on the results of a thorough investigation on existing types of services that was undertaken for the purposes of the SODIUM project, we came up with a set of incompatibilities. The service types that we have investigated are that of Web, Grid and P2P services. In this article we focus on the incompatibilities between Web and P2P services which are the following:

- **Supported Models:** Web services and P2P services adhere to different models of operation; specifically Web services follow the stateless service model, although recently the concept of stateful Web services has arisen (Czajkowski, 2004) whereas P2P services are leaning towards the stateful service model, though some stateless implementations also exist (Nejdl, 2001).
- **Targeted Clients:** Although there might be specific security constraints dictating a different case, Web services in general may be invoked by any client with internet access, provided that the client has the necessary infrastructure (e.g. a SOAP engine) to exchange messages (e.g. SOAP messages) with the service provider. When it comes to P2P services, the respective client must be either a member of the P2P network or have access to the network through another peer (proxy peer).
- **Syntactic Features:** The different models supported by the investigated types of services result in a set of syntactic diversities as well. Briefly, while Web services adhere to the WSDL defined service model (i.e. a service comprises a set of distinct operations) and utilize the SOAP message format, P2P services adhere to a suppressed service model where each service comprises a single operation, and employ proprietary message formats (see for example JXTA Org, 2006). Furthermore, for the description of P2P services it is necessary to use concepts denoting the P2P network topology (e.g. peer, peer group, etc).
- **QoS Properties:** The origins of Web services are in business-oriented systems, whereas P2P services, such as instant messaging or file sharing systems, have been derived from community-oriented systems. Albeit there are certain implementations that do not adhere to the following claim, we may argue that the origins of each of these service types have dictated specific quality properties for each of them respectively. Thus, P2P services in general do not have to be reliable, secure and of high performance, whereas Web services should be.

All these incompatibilities and differences among the investigated types of services lead to a number of interoperability concerns that can be classified according to the previously identified interoperability dimensions (illustrated in Figure 5). These concerns are summarized below:

- **The incompatibility of the supported models has an impact on all interoperability dimensions but quality and context.**

The stateful service model that is accommodated by P2P services and stateful Web services, necessitates the use of additional features and

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

processes e.g. resource lifetime properties and lifetime management mechanisms or processes supporting the discovery of resources. These additions have several ramifications on the platforms that are leveraged for the provision and invocation of stateful services, on the steps that need to be followed for the integration of such services, as well as on the semantics used for their description.

- **The difference on the targeted clients has an impact on the signature and protocol dimensions as well as on the platform and application dimensions.**

As it has been mentioned before, the invocation of P2P services can only be performed by peers of the P2P network using the API, mechanisms and protocols supported by the network. It henceforth, becomes apparent that interoperability among services which have different targeted clients e.g. Web and P2P services, should take into account the APIs, mechanisms and protocols supported by each service type.

- **The incompatibility of the syntactic features has an effect on the signature and platform dimensions.**

The use of incompatible structures and elements for the description of service syntactic characteristics has an effect on the languages used for their descriptions as well as on the middleware used for the provision and usage of such services. Therefore, in order to facilitate the integration of services with incompatible syntactic features, specific middleware needs to be provided, accommodating mappings of the features of one service type to the features of another service type.

- **The incompatibility of QoS properties has an effect on the quality dimensions as well as on the platform and application dimensions.**

Considering for example the issues of security, billing or availability, appropriate middleware and processes need to be used so as to cater for the provision, monitoring or management of such quality properties. Thus, the integration of services which are either demanding or providing incompatible quality characteristics has to be supported by appropriate middleware as well as by the use of specific process steps within an application.

Based on the results of our analysis that is illustrated in Table 1, the *signature* and

Table 1. Effect of service differences on interoperability dimensions

		Differences Between Web, Grid and P2P Services			
Interoperability Dimensions		**Supported Models**	**Targeted Clients**	**Syntactic Features**	**QoS Properties**
	Business Domain	X			
	Application	X	X		X
	Platform	X	X	X	X
	Semantic	X			
	Protocol	X	X		
	Context				
	Quality				X
	Signature	X	X	X	

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

protocol interoperability dimensions as well as the *platform* and *application* interoperability dimensions are the ones that seem to be highly affected by the integration of heterogeneous services. Thus, a solution for the integration of heterogeneous services should pay special attention on these dimensions.

We would like also to mention that, as it can be noticed, none of the identified discrepancies among P2P and Web services resulted in interoperability concerns for the context dimension. This is due to the fact that our investigation was focused on the interoperability problems that emerge when integrating heterogeneous services in general, regardless of their context.

In the following section we present our approach in addressing the aforementioned interoperability dimensions which emerged by the differences between the investigated service types.

GENERIC SERVICE MODEL

Based on the findings of the previous section we developed a generic service model (GeSMO) which addresses service interoperability. GeSMO provides the basis for the development of appropriate middleware, languages and tools, which facilitate the interoperation of heterogeneous services. Its current version focuses on the syntactic aspects of a service; thus, it mainly addresses the signature and platform service interoperability dimensions. This was deemed necessary for the quick provision of input that would consequently be used in the development of the rest of the SODIUM project results i.e. tools and languages. However, as it will become apparent in the following, GeSMO is open and extensible, so it can easily support any other required interoperability dimensions.

As we have mentioned before, GeSMO was based on a thorough investigation of the current state of the art on contemporary types of services. Specifically with respect to P2P services, GeSMO has been primarily influenced by the work in JXTA (JXTA Org, 2006) as the latter is one of the very few P2P networks that inherently support the notion of a service. As regards Grid services, GeSMO was influenced by the WSRF specification (Czajkowski, et al. 2004). However, the following characteristics render GeSMO an ideal basis for supporting other types of services. Specifically, the model exhibits the following properties:

- **Generality:** The model is generic enough, as it provides a core part which contains concepts common to all investigated service types. On the other hand, it provides specific characteristics for each supported service type that are accommodated in the various modules of its extension layer. Modeling of service types beyond the Web, grid and P2P services that are currently supported, such as sensor services or UPnP services can be easily accommodated by adding extra modules in the GeSMO extension layer containing the specific characteristics of the new service types.
- **Abstraction:** The core part of the model incorporates abstractions of all common concepts of the addressed types of services, e.g. the model contains concepts such as "service" and "message" which are part of all types of services; these abstract concepts can then be instantiated to the concepts supported by each specific type of service.
- **Extensibility:** The model can be easily extended with new features and properties, as well as with new service types. There are two ways of extending the model: a) by specialization of existing concepts; b) by appending new concepts in case there are no appropriate existing ones to be used as a basis for specialization. For example, the introduction of elements for the modeling of grid services has been achieved through the utilization of these mechanisms; in particular, elements for modeling the notion of "resource" were appended to the model whereas the notion of "grid service" was introduced as a specialization of the "Web Service" concept.
- **Modularity:** GeSMO has been structured in a modular manner; thus, information

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

elements derived from specific viewpoints have been encapsulated within specific packages; for example, concepts related to the description of a service are contained within a description viewpoint package as it has been shown above. Such a modular structure allows for the easy modification and/or extension of the various parts.

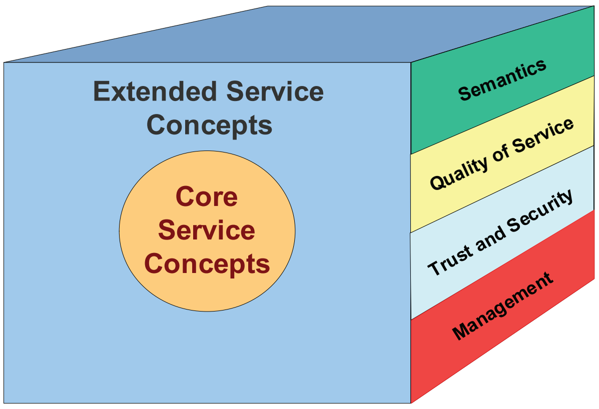
- **Expressiveness:** The model is expressive enough to accommodate several service primal activities such as description, discovery, invocation, and composition. In addition, appropriate extension points were defined so as to accommodate additional features such as Semantics and QoS descriptions, etc. This expressiveness facilitated the development of tools & mechanisms which, based on the model, were able to support the unified discovery, composition and execution of Web, Grid and P2P services (Tsalgatidou, et al. 2006)
- **Simplicity:** The model is simple enough to be easily used by a variety of users and tools. For example, within the context of the SODIUM project the model was exploited in two ways: (a) it served as the basis for the development of service modeling, service discovery, service composition and service execution languages and tools; (b) it facilitated the communication among the project stakeholders.

The architecture selected for the development of the generic service model (GeSMO) is a layered one, consisting of the following layers which are depicted in Figure 6:

- The **core service concepts** layer which models concepts that are common to all investigated types of services, i.e. Web, Grid and P2P services;
- The **extended service concept** layer, on top of the core layer, which provides for the distinct features of each service type;
- A number of layers orthogonal to the core layer and its extensions, which provide appropriate extension points for modeling features related to *semantics, quality, trust, security* and *management*; these features can be applied to all concepts of the core layer and of its extensions; this approach renders GeSMO flexible enough with respect to the adoption of constructs for the representation of all these cross-cutting aspects in which currently there are no prevailing protocols or models.

The concepts of the core part of GeSMO are presented next, followed by the extensions that we have developed for P2P services.

Figure 6. GeSMO's layered architecture



Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 7. Generic service model viewpoints

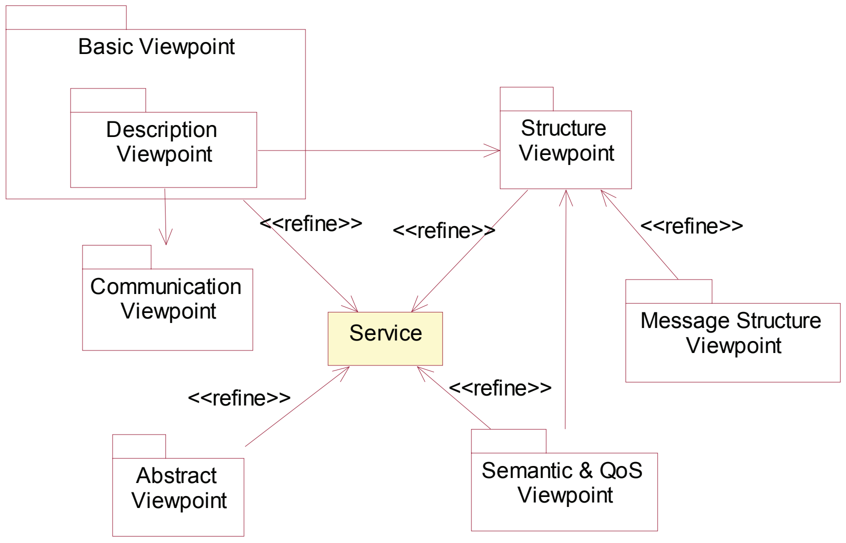


Figure 8. A service from an abstract point of view

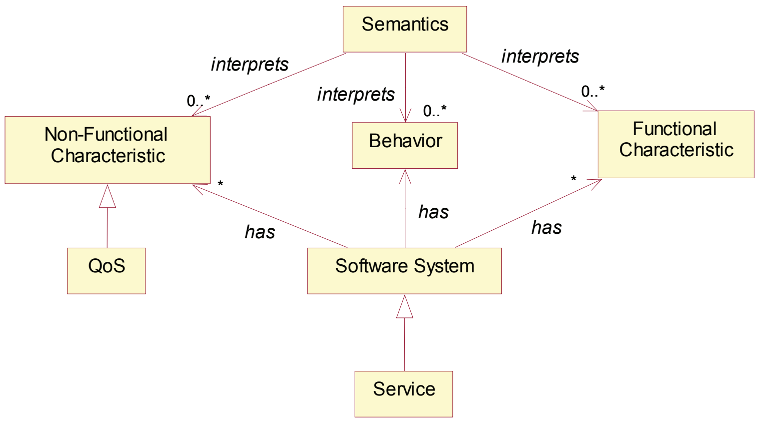
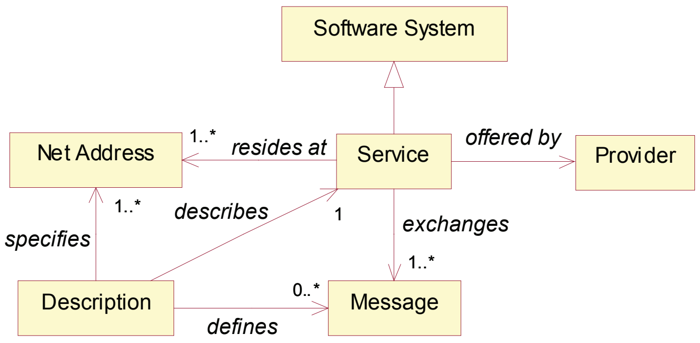


Figure 9. Basic service model



Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 10. Description point of view of a service

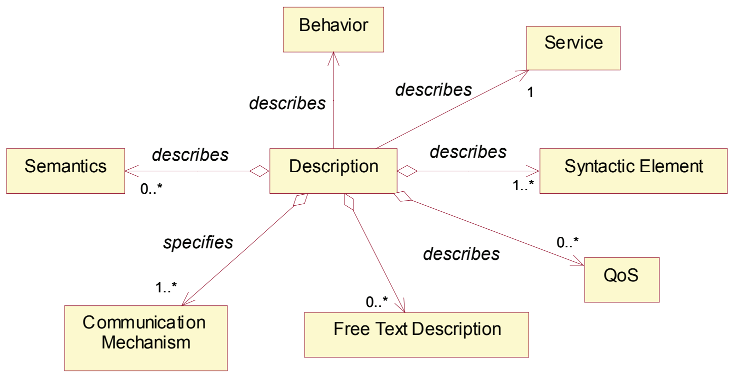


Figure 11. Structural point of view of a service

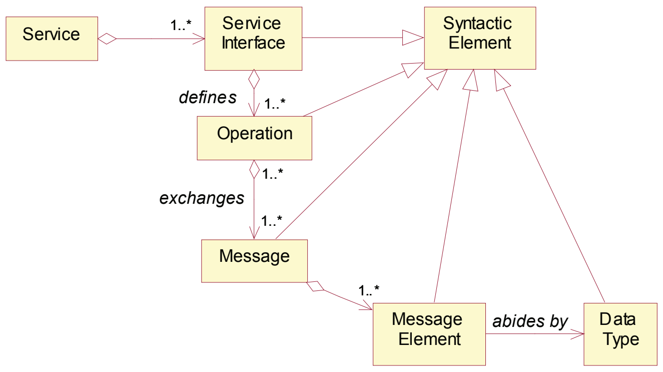
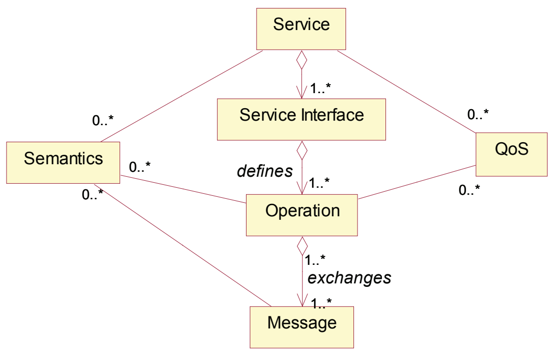


Figure 12. Semantic and QoS point of view for a service



The Core part of GeSMO

The notion of service is the fundamental element in the core part of GeSMO model and it is modeled through various points of view. Figure 7 depicts the service concept along with the viewpoints that were used for its refinement.

As we can see in Figure 7, the viewpoints that were used for refining the concept of service are: the abstract point of view, the basic point of view, the description point of view, the structural point of view and the semantic and QoS point of view. These views are further detailed in the following figures.

Figure 8 depicts a service from an abstract view as a software system which has a set of functional and non-functional characteristics and exhibits a specific behavior. Its behavioral, functional and non-functional properties may have a semantic interpretation, whilst some of its non-functional properties can be considered as QoS properties.

Figure 9 depicts the basic service model where a service is a self-described software system that resides at a network address, is offered by a service provider and exchanges messages which are defined in its description.

The description of a service conveys a lot of important information, therefore we have developed a distinct description point of view which is presented in Figure 10. As we see in this figure, a service description may convey information about the behavioral, semantic and quality features of a service. In addition, it provides information about the communication mechanisms that may be used for accessing the service.

As regards the syntactic elements of a service, more information is provided by the structural point of view, which is depicted in Figure 11. According to this view, a service provides one or more interfaces consisting of operations that group a set of messages. Each message comprises a set of elements which adhere to specific data types. Messages are used for the communication between a service and its respective clients.

An additional point of view provided by the core part of GeSMO for a service is the semantic and quality-of-service (QoS) point of view depicted in Figure 12. As it can be seen, a service, its operations and the exchanged messages may have semantic interpretation. In addition, a service and its operations may also be associated with specific QoS properties.

As we mentioned in the beginning of this section, the aforementioned concepts along with the additional ones which have been defined in (Tsalgatidou, 2005) establish the core part of the GeSMO model. In the following paragraphs, we present how these concepts were extended in order to support the rendering of P2P services in general as well as the description of JXTA P2P services offered by the JXTA platform. These concepts served as the basis for the specification of a language used for the description of P2P and JXTA services, as well as for the development of a P2P service invocation mechanism. Both the language and mechanism are described later on in this article.

GeSMO Extensions for Peer-to-Peer Services

The GeSMO peer-to-peer service model extends the GeSMO core concepts with additional elements needed for the description of P2P services. As it can be seen in Figure 13, the additional concepts catering for the description of P2P services are:

- **P2P service** which represents a service provided by a peer or a group of peers in a P2P network
- **Peer** which represents a node in a P2P network, capable of communicating with other peers and provide them with services
- **Peer group** that represents the logical grouping of peers within a P2P network
- **PSDL** (P2P Service Definition Language) description which is a document defined according to the PSDL language that is described in the following section. This document conveys information on what a P2P service does and how it can be invoked.

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 13. P2P service model

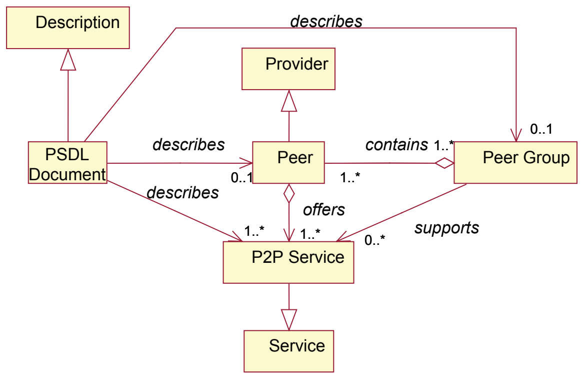
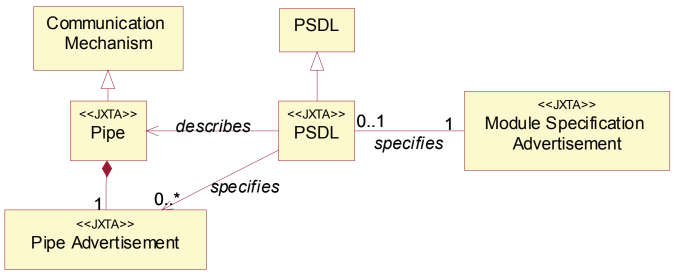


Figure 14. JXTA service model



The PSDL concept was specifically added in order to facilitate the description of P2P services with additional information elements. Hence, a PSDL document does not aim at replacing the existing description constructs provided by the underlying P2P platform, but rather to enhance them with information that these constructs lack. Furthermore, PSDL descriptions provide the basis upon which mechanisms and middleware can be built so as to facilitate the more efficient discovery and utilization of P2P services.

The aforementioned concepts represent a basic set of elements that may be used for the description of any kind of P2P service. However, when it comes to modeling P2P services offered by specific platforms such as JXTA (JXTA Org, 2006) or Edutella (Nejdl, 2001), additional elements or specializations of the aforementioned ones might be needed.

JXTA is one of the most well known and mature platforms that can be used for the development of P2P services. Also, as we mentioned before, it is one of the very few P2P platforms that inherently support the notion of service, therefore it is the most prominent candidate to be modeled by GeSMO. With respect to JXTA services (Gong, 2001), JXTA offers constructs such as pipes, module specification advertisements and pipe advertisements. Pipes are communication mechanisms used for handling the exchange of messages among peers, thus facilitating the interactions between service requestors and service providers. Pipe advertisements and module specification advertisements are specializations of the advertisement construct and are used for publishing information about pipes and services, respectively.

Figure 14 illustrates the specializations of the P2P service model, which are provided by GeSMO for the representation of the afore-

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

mentioned JXTA concepts. An extended PSDL description for the JXTA platform describes the pipes that can be used for exchanging messages with a service and provides references to their respective Pipe Advertisements. A P2P service module specification advertisement provides a reference to the PSDL description document of that service. This association is accomplished through the use of an extension element (i.e. SURI element (Duigou, 2003, pp. 23-24)) that is inherently supported by the JXTA Module Specification Advertisement.

The specification of the PSDL language which utilizes the aforementioned GeSMO extensions for the description of P2P service is presented next.

A PEER-TO-PEER SERVICE DESCRIPTION LANGUAGE

The *peer-to-peer service description language (PSDL)* was introduced as a mechanism that leverages interoperability between P2P services as well as between P2P and other types of services. Specifically, it incorporates a set of extensions upon the WSDL specification (Christensen, 2001) which were developed with the goal to support the description, discovery and invocation of P2P services. The design of PSDL was influenced by:

- the concepts and properties of the P2P service model as they have been perceived in GeSMO,
- the existing mechanisms (i.e. middleware, languages and tools) supporting the description, discovery and invocation of services, and
- the need to be extensible and customizable so as to enable the description of P2P services offered by various types of P2P platforms.

Instead of developing PSDL from scratch, we decided to develop it as an extension to WSDL. The main reason for this decision was our intention to reuse existing standards rather than introducting another totally new language and thus complicate further the interoperability problems. Thus, the intrinsic similarities between the concepts and the structure that were required to be supported by the PSDL and those of the WSDL language (i.e. the use of the syntactic elements modeled in the core layer of GeSMO (see Figure 11)), the availability of ample tools and middleware supporting the construction and utilization of WSDL documents and the WSDL support for extensibility, led us to use WSDL as a basis for the development of PSDL.

Although W3C (www.w3c.org) has lately released the specification of WSDL 2.0 (Chinnici, 2006) most of the existing tools and middleware have not yet completely adopted this new version of the specification. Therefore, the version of WSDL that we used as a basis for the development of the PSDL language is the WSDL 1.1 (Christensen, 2001).

As far as the elements of WSDL 1.1 (Christensen, 2001) are concerned, the ones catering for the description of a Web service abstract interface (e.g. PortType, Operation, Message and Types elements) can be reused in the description of a P2P service abstract interface as well, since both PSDL and WSDL utilize the same concepts. However, with respect to the concrete information part of a service, the elements provided by WSDL can not cater for the description of the concrete details of a P2P service. Thus, in order to render PSDL able to deliver its intended functionality, i.e. description, discovery and invocation of P2P services, we had to introduce appropriate extensions which are presented in the following.

According to (Christensen, 2001), WSDL provides the necessary extension points which can easily accommodate our requirements. The binding and port elements of the WSDL language provide the basis upon which P2P related concepts can be constructed. The introduced elements were devised in a way that enables the description of various types of communication/interaction patterns between a P2P service provider and its respective clients. Moreover, these elements were defined with extensibility and customizability in mind, in order to easily

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Table 2. Imported Namespaces and associated prefixes

xsd	http://www.w3.org/2001/XMLSchema
wsdl	http://schemas.xmlsoap.org/wsdl/

Table 3. Namespaces for introduced concepts

gesmo	http://metis.di.uoa.gr/Sodium/gesmo/services/P2P/
jxta	http://metis.di.uoa.gr/Sodium/gesmo/services/P2P/jxta/

provide for the properties and characteristics of various P2P service provision platforms.

Apart from the PSDL language which facilitates the description of P2P services, one may argue that extensions and modification might be needed in other widespread protocols and standards of service oriented computing such as for example SOAP in order to further support interoperability among service types. There are projects striving towards the adoption of the SOAP protocol for exchanging messages among peers of a P2P network (e.g. http://soap.dev.java.net/), however, we decided not to ensue such an approach because our intention was to retain the autonomy of the existing P2P platforms and middleware without affecting their primal features, such as their communication mechanisms. Therefore, we focused on the provision of the PSDL language.

In the following, we will illustrate and elaborate on the extensions that we have developed, to support the representation of P2P services and their customizations for the description of JXTA services. Subsequently, we are going to exemplify how the PSDL language is used for the description of an Instant Messaging service.

The namespaces that were used as the basis for the development of appropriate extensions along with their associated prefixes are presented in Table 2.

The namespaces and assigned prefixes of the introduced elements are presented in Table 3.

The notation and the conventions that will be used hereafter for the description of the provided extensions are as follows:

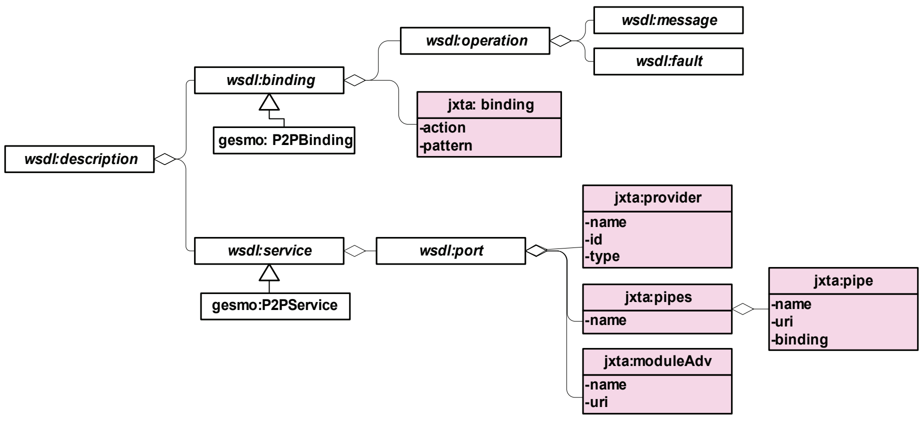
- The name of each element is going to be preceded by its associated prefix that is specified in the tables above. Moreover the names of the WSDL elements that are going to be used in the following are in italics.
- The definition of each of the introduced elements (i.e. elements whose prefix is going to be 'gesmo' or 'jxta') adheres to an informal XML grammar, which uses the same notational conventions as the ones specified in (Christensen, 2001, section 1.2). In addition to the specified notational conventions the '|' character is used as separator among a list of optional values that can be assigned to an attribute or as a separator among a list of sub-elements that may be assigned to a specific element.
- The elements whose names start with a capital letter are abstract elements[4] whilst the elements whose name doesn't start with a capital letter can be instantiated within a document.

PSDL Language Elements

Since PSDL is based on the WSDL 1.1 specification (Christensen, 2001), it inherits and retains the structure of the WSDL elements. In addition, it extends WSDL with the information components that have been presented above. We need to stress here that only some

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 15. PSDL extensions to WSDL



of the elements of WSDL 1.1 are extended in PSDL, whilst the rest of the WSDL concepts, elements and mechanisms (e.g. the import and include WSDL mechanisms) are reused without any modification. An illustration of the WSDL information components that have been utilized to accommodate the concepts of the PSDL language and the provided extensions are presented in Figure 15. Please note that the provided extensions appear in the colored boxes, while the WSDL elements are white.

More details on those elements along with their definitions are presented in the following. First we present some general concepts that can be used in the description of various P2P services and then we illustrate the concepts that can be used in the description of JXTA services.

General P2P Service Related PSDL Elements

The gesmo:P2PBinding and the gesmo:P2PService components are abstract constructs that provide the foundation for the development of P2P platform related elements. These elements serve as placeholders where concrete information supporting the binding and invocation of P2P services may be conveyed.

Specifically, the gesmo:P2PBinding component is an extension of the *wsdl:binding* component, which provides information related to i) the interaction patterns supported by a service, ii) the operations associated with these patterns, and iii) the protocols used for the message exchange. On the other hand, the gesmo:P2PService component is an extension of the wsdl:service element, which conveys information related to i) the *wsdl:ports* where a service resides and ii) the *wsdl:bindings* that one may use to access each *wsdl:port*.

The formal definition of those elements is presented next.

gesmo:P2PBinding: This element denotes that the underlying communication mechanisms and the associated protocols will be related to P2P technology. An illustration of the attributes and sub-elements of this element is presented in Figure 16. According to this description, the P2PBinding element has a name and an interface attribute. The name attribute is used for the element identification and the interface attribute holds a reference to the associated interface (i.e. related wsdl:portType element) element. A P2PBinding element may include a documentation sub-element as well as a set of feature, property or operation sub-elements.

Please note that, the P2PBinding element is abstract, thus it can't be instantiated and included in a P2P service description document. Appropriate extensions of this element need to

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 16. P2PBinding abstract element

```
<gesmo:P2PBinding name= “xs:NCName”
interface=”xs:QName” abstract= “true”>
<documentation />?
  [<feature /> | <property /> | <operation />]*
</gesmo:P2PBinding>
```

Figure 17. P2PService abstract element

```
<gesmo:P2PService name= “xs:QName” abstract= “true”>
<documentation/>?
<wsdl:port/>*

</gesmo:P2PService>
```

be created in order to convey information related to specific P2P service provision platforms.

gesmo:P2PService: The gesmo:P2PService element contains information about the endpoints where the service resides. Each P2PService element should have a name which uniquely identifies this element within a service description document. An illustration of the structure of this element is presented in Figure 17. As it can be seen, a P2PService element may contain additional documentation information as well as a set of *wsdl:ports* that its clients should use in order to access the service.

Note that, the P2PService element is also abstract, therefore appropriate extensions need to be provided in order to convey additional information related to specific P2P service provision platforms.

We would like to note here again that the gesmo:P2PBinding and the gesmo:P2PService abstract elements were introduced so as to serve as placeholders for future extensions of the PSDL language catering for specific types of P2P services e.g. Gnutella P2P services. Thus, these elements merely stand as boundary holders that support the structure of the PSDL language.

JXTA specific PSDL Elements

JXTA provides a set of protocols and tools which leverage the development of P2P services and applications. According to (Sun Microsystems, 2005), JXTA services may be provided to their clients via various mechanisms, i.e. through pipes directly or via proxy modules which have to be installed by clients prior to accessing a service. In the work presented here, we describe the support that we have provided for JXTA services accessible through pipes.

Although our model can be extended to support JXTA services accessible through proxy modules, we have chosen not to address such services, mainly due to the following reasons: the first one being the deviation from the service model (Vogels,2003) that this approach would introduce, i.e. the resulting model would resemble more to a distributed component model rather than a service model; and the second reason being the complexity and source of inconsistencies that this would result in, e.g.

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 18. Binding type declaration

```
<jxta:binding action= "send|receive|sendreceive"?
pattern= "synchronous|asynchronous"?/>
```

Figure 19. Provider element

```
<jxta:provider name= "xs:NCName" type= "Peer|Group"/>
```

the discovery and invocation of such services would also involve the discovery and invocation of appropriate proxy components.

The elements that have been introduced for the description of a JXTA service are the jxta:binding, jxta:provider, jxta:pipes, jxta:pipe and jxta:moduleAdv, as it can be seen in Figure 15. More details on each of these elements are presented next.

jxta:binding: In order to provide for the specification of JXTA binding schemes that may be used by a JXTA service, we have introduced the jxta:binding concept. This element is used for extending the *wsdl:binding* element in order to declare that this is a JXTA binding and in order to specify the action and the communication pattern that will be supported by this binding.

As it is illustrated in Figure 18, an element of this type has two optional attributes; the "action" attribute which is used for the specification of the type of actions that are supported i.e. send-receive, send or receive, and the "pattern" attribute which is used for the specification of the interaction type, i.e. synchronous or asynchronous.

The type of binding that is supported by a P2P service depends on the type of the pipe that will be used for communicating with that service. Hence, if a service uses a bi-directional pipe for receiving and replying to messages this service supports send-receive actions. Depending on if this is a blocking or a non-blocking communication the interaction pattern for this binding could be either synchronous or asynchronous.

jxta:provider: The jxta:provider element has been introduced to support the specification of the JXTA service provider and the type of the service. The PSDL specification dictates that only one instance of this element must be present within an enclosing port element and its properties should correspond to the ones of the service at hand.

The attributes of this element, which can be seen in Figure 19, provide for the distinction among the types of services that may be provided in a JXTA network (i.e. either Peer or Group services according to (Sun Microsystems, 2005)) and for the specification of the name of the service provider. Therefore the "name" attribute is used for describing the name of either the peer or the group which offers a specific service and the "type" attribute specifies whether this is a peer or a group service (see (Sun Microsystems, 2005).

jxta:pipes: The jxta:pipes element provides for the description of the list of pipes that are supported by a specific endpoint in order for someone to able to exchange messages with a service. Specifically the jxta:pipes element conveys a listing of the pipe advertisements that one should use in order to establish pipe connections with a JXTA service.

As it can be seen in Figure 20, a jxta:pipes element should contain at least one "pipeAd" element. Although, in most of the cases the use of a single pipe is adequate for the establishment

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 20. Pipes element

```
<jxta:pipes name="xs:NCName"? >
 <jxta:pipeAd/>+
</jxta:pipes>
```

Figure 21. PipeAdv element

```
<jxta:pipeAdv name= "xs:NCName" uri= "xs:URI"
binding="xs:QName"?/>
```

Figure 22. ModuleAdv element

```
<jxta:moduleAdv name= "xs:NCName" uri= "xs:URI"/>
```

of a communication channel among a JXTA service and its respective client the jxta:pipes element may contain additional jxta:pipeAd elements in order to be able to support more complex communication schemes - which are not excluded by the JXTA framework (Sun Microsystems, 2005) - that require the use of more than one pipes for the same endpoint. Furthermore, the jxta:pipes element may have a "name" that is used for referring to that element within a specific *wsdl:port*.

jxta:pipeAdv: The jxta:pipeAdv element is used for providing information regarding the pipe advertisement that a client may use in order to open a message exchange channel with a service. In addition, a jxta:pipeAd element provides information related to the types of interactions that are supported via the respective pipe.

As it is illustrated in Figure 21 each instantiation of the pipeAdv element should have a "name" which uniquely identifies this element within the list of pipeAdv elements that may be contained in an enclosing jxta:pipes element. Furthermore, an instantiation of the jxta:pipeAdv element should have a reference to the pipe advertisement file (by using the "uri" attribute) that a client should use in order to establish a pipe connection with the service. Finally, an instantiation of this element could have a description of the binding that this pipe supports. In case this is omitted, the binding of the enclosing Port element is used instead.

jxta:moduleAdv: The jxta:moduleAdv element was introduced so as to cater for the specification of the module specification advertisement that a JXTA service is using for its description (see Figure 22). A module specification advertisement is an advertisement document that is leveraged by the JXTA network so as to provide for the specification of JXTA services. More details on the Module Specification Advertisement can be found at (Duigou, 2006).

An instantiation of this element (see Figure 22) has a "name" and a "uri" attribute. The "name" attribute is used for the identification of the moduleAdv element within the enclosing

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

wsdl:port element. The “uri” attribute on the other hand provides for the specification of a reference to the module advertisement file of the service at hand.

This concludes the definition of PSDL and the description of its elements.

A PSDL Example

The elements of the PSDL language can be used to describe any JXTA service that adheres to the language constraints, as they were specified above. Taking for example an Instant Messaging (IM) service that reliably exchanges messages with its clients through a bi-directional pipe (Sun Microsystems, 2005, pp 14-17), its service description document will be as the one illustrated in Figure 23. The service described is a JXTA peer service called *“IMService”*, which comprises a single operation called *“sendReliableMessage”*. This method accepts messages of type *“IMessage”* and returns messages of type *“IMResponse”*. As it can be easily seen, the description of the abstract part of the *“IMService”* service is the same as that of a Web service. The elements that have been introduced for the specification of a JXTA service can be identified within the binding and service WSDL elements.

The jxta:binding element included in the *“JxtaBidiBind”* element denotes that the *“IMServiceIF”* port type, that is referenced by the binding’s type attribute, is an interface

Figure 23. Instant messaging service description in PSDL

```
<message name=”IMessage”>
  <part name=”MessageContent” type=”xsd:string”/>
  <part name=”MessageSender” type=”xsd:string”/>
</message>
<message name=”IMResponse”>
  <part name=”MessageAknow” type=”xsd:string”/>
</message>
<portType name=”IMServiceIF”>
  <operation name=”sendReliableMessage” parameterOrder=”MessageContent Mes-
sageSender”>
    <input name=”msg” message=”tns:IMessage”/>
    <output name=”resp” message=”tns:IMResponse”/>
  </operation>
</portType>
<binding name=”JxtaBidiBind” type=”tns:IMServiceIF”>
   <jxta:binding action=”sendreceive”/>
   <operation name=”sendReliableMessage”>
     <input/>
     <output/>
   </operation>
</binding>
<service name=”IMService”>
   <port name=”JxtaBidiPort” binding=”tns:JxtaBidiBind”>
     <jxta:provider name=”Jemini” type=”Peer”/>
     <jxta:pipes name=”IMServicePipes”>
     <jxta:pipeAd name=”BidiPipe” uri=”http://metis.di.uoa.gr/.../IMS_BidiPipeAd.xml” />
     </jxta:pipes>
<jxta:moduleAdv name=”IMServiceModuleAd” uri=”http://metis.di.uoa.gr/.../IMSpecAdx-
ml”/>
  </port>
</service>
```

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

of a JXTA service. This binding supports send-receive message exchanges using the default synchronous communication pattern. According to the information that is conveyed in the *"JXTABidiPort"* element, we can see that the *"IMService"* service, which adheres to the *"IMServiceIF"* port type, is a peer service. This service provided by a peer of a JXTA network whose name is *"Jemini"*. Clients of that service may use the *"BidiPipe"* pipe advertisement in order to access this service, whilst its module specification advertisement that is called *"IMServiceModuleAd"* can be retrieved at a specific location that is specified by the uri attribute of the jxta:moduleAdv element.

This PSDL description document can be fed as input to the JXTA service invocation mechanism that is described next, so as to facilitate the flexible invocation of the described service, without the developer caring for the underlying JXTA network details.

JXTA SERVICE INVOCATION MECHANISM

Based on (Sun Microsystems, 2005), a JXTA service, that is made available to its clients through pipes, can be invoked via messages that are a priori known to the service and its clients. The service advertisement mechanism provided by the JXTA framework does not convey appropriate information elements that could facilitate the documentation of the exchanged messages. Thus, the description as well as the interpretation of messages is normally embedded within the service and the client code.

Therefore, for someone to utilize JXTA P2P services in the same way as other types of services, e.g. Web services, and to compose them with other services, appropriate invocation mechanisms need to be put in place. These mechanisms should enable the dynamic invocation of JXTA P2P services based on annotated service description documents. Such a mechanism was developed by the authors of this article so as to enable the invocation of JXTA P2P services via the SODIUM execution engine based on the use of PSDL description documents.

The architecture of the provided JXTA P2P service invocation mechanism has been based on the use of the web service invocation framework (WSIF) (Duftler, 2001). This design decision was driven by the similarities of the PSDL and the WSDL languages and by the fact that WSIF is offered as open source.

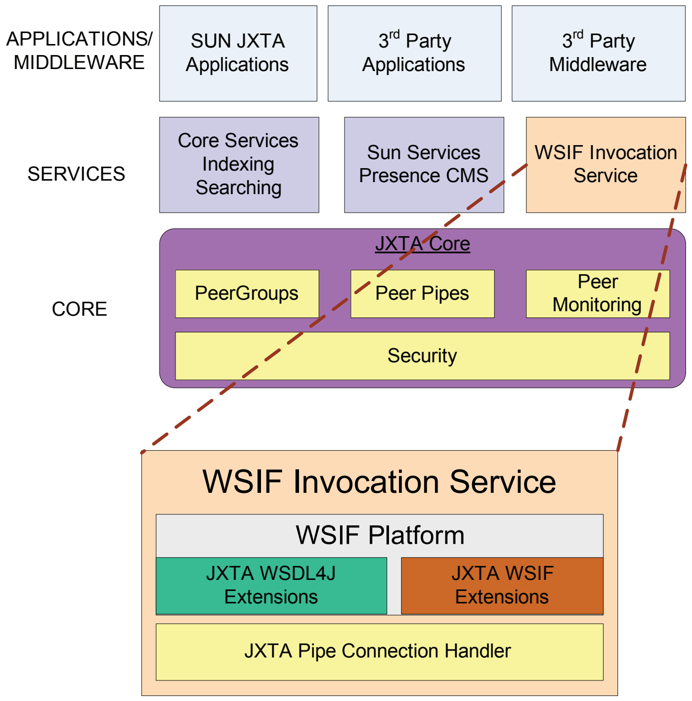
WSIF accommodates the necessary programming abstractions that enable the invocation of WSDL compliant services over various communication protocols and message formats. It enables either the dynamic or the static (i.e. via appropriate stubs that are created at design time) invocation of services irrespectively of the protocol bindings and the implementation details that they use. In order to facilitate the invocation of any type of code that may be available at the network, the framework provides appropriate extension points that can be utilized by developers so as to accommodate the peculiarities of specific implementations and communication protocols.

Currently, the WSIF framework provides extensions that enable the utilization of implementations such as Java code, EJBs and J2EE Connectors in a service-oriented manner. Nonetheless, developers are able to create 'WSIF Providers' (i.e. extensions to the WSIF framework) along with appropriate extensions to the WSDL[5] language that enable the utilization of additional types of services. Such a 'WSIF Provider' was developed by the authors of this article in order to support the invocation of JXTA services in a more flexible and service-oriented way. An illustration of the provided component and its placement within the JXTA reference architecture is presented in Figure 24.

The provided mechanism (*'WSIF Invocation Service'* component illustrated in Figure 24) is an extension to the JXTA platform reference architecture (Gong, 2001) that leverages other JXTA supported services and core components so as to enable the flexible invocation of JXTA services. Specifically the *'WSIF Invocation Service'* component utilizes the PeerGroup component and the pipe service to support the establishment of pipe connections between a

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Figure 24. "WSIF Invocation Service" component integration with the JXTA Reference Architecture



service and a client. Consequently the *'WSIF Invocation Service'* component can be utilized by other JXTA applications so as to facilitate the dynamic invocation of JXTA services that are described using the PSDL language.

As it can be seen in Figure 24, the components that have been added to the WSIF Platform in order to support the invocation of JXTA services are the *"JXTA WSDL4J Extensions"* and the *"JXTA WSIF Extensions"*. The *"JXTA WSDL4J Extensions"* contains all the necessary extensions to support the parsing of a PSDL description document and its mapping to in-memory objects, whereas the "JXTA WSIF Extensions" comprises the extensions that are mandated by the WSIF framework (WSIF Org, 2006) to enable the dynamic invocation of JXTA services. The "JXTA Pipe Connection Handler" on the other hand leverages services and components of the JXTA platform so as to enable the establishment of pipe channels and the exchange of messages.

Although JXTA supports the exchange of either XML or Binary messages (Sun Microsystems, 2005), our current implementation of the *'WSIF invocation service'* facilitates only the exchange of XML formatted messages. This decision was influenced by our need to have a clear mapping between the message representation conveyed in a PSDL description document and the message that is exchanged through a pipe connection. Nonetheless, the invocation mechanism may be extended in order to facilitate the transformation of messages to a Binary format that will be used over a pipe connection between a service and its clients. However, the description of such a transformation mechanism falls outside the scope of this article.

In spite of the placement of the *"WSIF Invocation Service"* component within the JXTA Reference Architecture (Sun Microsys-

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

tems, 2005) (see Figure 24) we do not imply a tight bond between the JXTA platform and the service invocation mechanism. Provided that appropriate extensions are developed for the PSDL language as well as for the *'WSIF Invocation Service'* so as to cater for the description and invocation of other types of P2P services e.g. Gnutella services (www.gnutella.org) or Edutella services (Nejdl, 2001), this mechanism can be easily plugged in to those platforms so as to facilitate the interoperation of their respective services. This is possible because, service consumers may invoke any type of service via the WSIF framework in a way that is independent of the underlying bindings and protocols. Thus, a client which might be a node of a Gnutella network is able to invoke services provided by peers of a JXTA network and vice-versa.

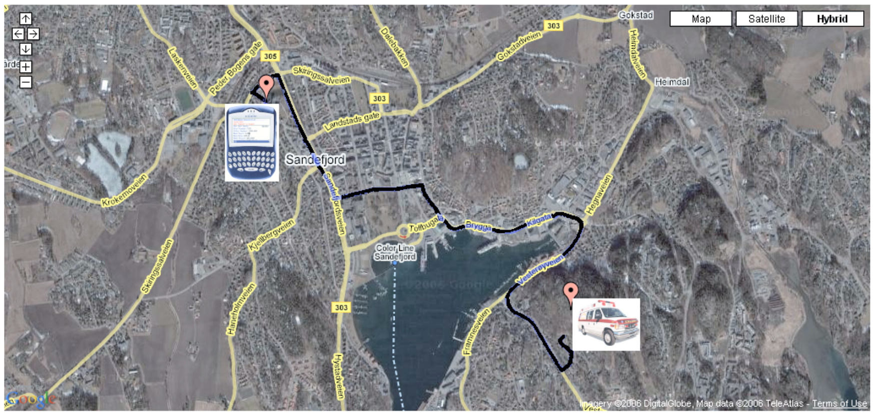
The level of interoperability that is achieved by the *'WSIF Service Invocation'* component is sustained by the extensibility traits of the WSIF framework (WSIF Org, 2006) and of the PSDL language. This level of interoperability has also facilitated the integration of the *'WSIF Service Invocation'* mechanism with the Execution Engine component of the SODIUM platform which supports the integration of heterogeneous types of services i.e. Web, grid, and P2P services. We would like also to note that this invocation mechanism can be easily used by any engine supporting execution of service compositions which contain P2P services described by PSDL documents.

We believe that it has now become apparent that the work described above (i.e. the GeSMO model and its extensions for P2P and JXTA services, the PSDL Language and the provided JXTA service invocation mechanism) provides for the interoperability between P2P services and between P2P and Web services. In order to better support our argument we illustrate next an example from an application which leverages P2P and Web services for the implementation of the motivating scenario that was presented in the beginning.

APPLICATION CASE STUDY

An important part of the motivating scenario described earlier in this article is related to the identification of the location of an accident, the identification of the nearest rescue unit and the calculation of the shortest route between them. According to this scenario, when someone calls to report an accident, his/her location can be

Figure 25. Rescue route calculation



Caller Phone: **91331952**
Caller Name: **Tone Hansen** - Address: **Lønnegloveveien 23B, 3230 SANDEFJORD**
Caller Position:**10.242499999999998, 59.123333333333342**
Closest Ambulance Location:**10.211637096117364, 59.134121265639223**

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

easily spotted either if he/she is calling from a fixed-line or from a mobile phone. Appropriate Web services providing the location of the caller already exist and can be used for this task.

Rescue units on the other hand are already equipped with communication devices which leverage P2P services in order to exchange location or other types of information (e.g. commands and other types of messages). Based on the caller location the closest and most properly equipped rescue unit is selected and dispatched to the accident place.

Maps with appropriate directions are transferred to rescue units in order to reach the accident location in time. Such maps and routing directions can be calculated using already existing Web services providing map and other related info (e.g. Google Maps) along with other services performing route calculations.

The pilot application that was developed within the context of the SODIUM project integrates the existing Web and P2P services (mentioned before) using the models and the mechanisms that have been described above in order to support this scenario. The pilot was deployed in Norway at LOCUS, the company which provided the motivating scenario. Figure 25 depicts the outcome of the developed application which is a satellite map, provided by the Google Map service, with the calculated route drawn upon it. This map along with the calculated route is sent to the ambulance that is closest to the accident with the use of a P2P service; all ambulances are peers of the same p2p network.

We believe that through this case study, it becomes apparent that the integration of P2P services with Web services is of great importance and facilitates a lot the development of service-oriented applications which need to integrate such heterogeneous services.

RELATED WORK

The main contribution of this article is in service interoperability and service modeling in general, via the proposed interoperability framework and the generic service model (GeSMO), and in the interoperability among P2P services and between P2P and Web services in particular, via the P2P service description language (PSDL) and the JXTA service invocation mechanism. Therefore, the related work that follows refers to any of these results.

The convergence of Web service and P2P computing models is an issue that has been tackled by many researchers (Papazoglou et al, 2004; Benatallah et al, 2003). A plethora of approaches have been proposed on how this can be achieved. In (Papazoglou et al 2004) a P2P architecture is used for the organization of federated UDDI registries. The introduced federated model enables the formation of syndicates where services of the same domain can be advertised. This approach departs from the one that we have presented within this article as we are considering P2P services as another service-oriented computing paradigm that can be utilized in the same way as Web services. Regarding the P2P service publication and discovery, we reuse the mechanisms supported by the underlying P2P network infrastructures.

The SELF-SERV platform that is proposed by (Benatallah et al, 2003) is also using a P2P infrastructure as the enabling technology which leverages the utilization of Web services, i.e. the discovery and composition of Web services, in a P2P organized manner. This approach also does not regard P2P services as another instantiation of the service-oriented computing paradigm, as we do. Therefore our approach is different from the one followed within the SELF-SERV platform.

Integration of the P2P and Web services worlds was also attempted in (Qu, 2004), by means of two interaction scenarios: i) exposing the existing Edutella/JXTA services as Web services, and ii) integrating Web service-enabled content providers into Edutella/JXTA. The ultimate goal of the proposed approach was to establish integration of Web and P2P services at the service layer through the use of proxies. Although sharing some common ideas, our approach is distinguished by the fact that the P2P and Web service technologies retain their autonomy. No additional components or infrastructure modifications are required in

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

our approach, besides the description of P2P services with the use of PSDL, or some other WSDL-based language.

P2P service description and invocation was also tackled in (Oriol, 2002). Therein, a number of alternative, distributed implementation infrastructures were proposed, which served the purpose of automating service discovery and invocation. Still, the approach was described from a different point of view than ours, addressing service description and invocation from a relatively higher level. In fact, our P2P service description language along with the WSIF-based invocation mechanism could be used complementary with respect to that approach. Combining our results with other similar efforts is one of our future research plans.

In (Elenius, 2004), a framework named Oden was proposed, which utilizes the OWL-S specification to annotate JXTA service descriptions with semantics. The results of this effort included automated P2P service discovery and invocation. However, the issue of P2P and Web services integration and interoperability goes beyond the scope of the Oden framework. As opposed to that, in our approach, we mainly focus on establishing a unified manner of describing and invoking heterogeneous services, and here we describe how this unified solution has been applied in the integration between Web and P2P services. To this end, we would like to note that, PSDL descriptions could be further annotated with semantic and QoS properties, so as to facilitate P2P service discovery without affecting the developed invocation mechanism.

Triana (Churches, 2006) is a platform that enables the distributed execution of scientific workflows. It leverages GAT (Allen, 2003) to support the provision of the necessary abstractions that facilitate the distribution and monitoring of workflows which is based on the Triana Group unit concept. GAT acts as an overlay above the JXTA infrastructure which enables the distribution of Group units among the peers of a JXTA network. GAT leverages the underlying JXTA Pipe and ID mechanisms to support the coordination and communication among the distributed Group units. Although, there are certain similarities between our service invocation mechanism and the GAT toolkit, our approach is different in that it has a direct support for the notion of JXTA service whilst GAT utilizes underlying JXTA communication and naming mechanisms to support the exchange of information among Triana's Group units.

CONCLUSION

Service oriented computing (SOC) has emerged as the computing paradigm for developing large-scale, distributed applications, by integrating existing pieces of software exposed as services. Although the primary objective of SOC was to leverage interoperability among the various heterogeneous software platforms and systems, its various diverse instantiations (e.g. Web and P2P services) introduced new interoperability issues. Thus, despite that the integration of the various heterogeneous types of services has been called for in many vital application domains, the incompatible underlying models, protocols and standards of such heterogeneous services render this an arduous task. Given this situation along with the absence of a set of common, widely-accepted standards governing all these service-oriented technologies, we argue that interoperability can be supported through the establishment of a unified approach in terms of models, languages and tools which are based on a common set of conceptual primitives.

In this article, we went through the foundations of our unified approach towards the integration and interoperability of heterogeneous services, and exemplified how such an approach was applied for the integration of Web and P2P services. In summary, we presented the dimensions influencing interoperability among the various service-oriented technologies and presented an integrated classification scheme for these dimensions. Then, we examined the impact of the discrepancies among the investigated types of services on the identified interoperability dimensions. The generic service model (GeSMO), which was developed in order to address some these interoperability dimensions was then described. Finally, we focused on the description of the extensions provided

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

in GeSMO for P2P services, and specialized in the case of JXTA services.

As it was discussed in this article, the layered architecture of GeSMO renders it extensible and independent from the various service types and their related standards, thus, a perfect basis for the development of unified languages and tools. We defended this argument by presenting PSDL, a WSDL-based language which uses appropriate GeSMO concepts and is used for the description, discovery and invocation of P2P services, enabling in this way P2P services to be integrated and interoperate with Web services. Along with PSDL, we presented an effective mechanism for the invocation of JXTA P2P services. We believe that the combination of this invocation mechanism (being based on the WSIF framework) with the PSDL language (being based on WSDL) provides a novel solution for the integration of Web and P2P services.

It should be noted that the extensibility of the approach presented in this article was exemplified by the extensions of GeSMO within the context of the SODIUM project, so as to address grid services as well. Therefore, all SODIUM tools and languages support Grid services, too, via appropriate extensions on GeSMO; hence they provide a platform which supports the unified discovery, composition and execution of Web, grid and P2P services. Furthermore, the presented solution was integrated with the results of another European project, namely SeCSE, which supports the development of service-oriented applications. We would like also to note that GeSMO and its extensions have been included in a proposal submitted to OMG as a response to a Request for Proposal for a UML Profile and Metamodel for Services[6] .

Finally, we would like to mention that, GeSMO served as a basis for handling interoperability at the signature or platform level, therefore it needs to be further extended in order to fully address the interoperability problem. In this regard, issues related to other interoperability dimensions such as protocol, quality or application, as well as semantic or business domain are going to be addressed in our future work. Furthermore, future plans regarding GeSMO include the provision of extensions to accommodate other types of services, such as sensor services, and the incorporation of additional P2P networks and platforms that provide for other P2P services, besides the JXTA platform.

ACKNOWLEDGMENT

This work has been partially supported by the Special Account of Research Funds of the National and Kapodistrian University of Athens under contract 70/4/5829 and by the European Commission (under contract IST-FP6 004559 for the SODIUM project and IST-FP6 511680 for the SeCSE project). Finally, the authors would like to thank all the SODIUM partners which contributed in this work and in particular LOCUS for their input in the motivating scenario and in the case study.

REFERENCES

Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.-T., Sheth, A., Verma, K., (2005) Web Service Semantics, WSDL-S, W3C Submission, http://www.w3.org/Submission/WSDL-S/

Allen, G., Davis, K., Dolkas, K., Doulamis, N., Goodale, T., Kielmann, T., Merzky, A., Nabrzyski, J., Pukacki, J., Radke, T., Russell, M., Seidel, E., Shalf, J., and Taylor, I., (2003), Enabling Applications on the Grid: A GridLab Overview, International Journal of High Performance Computing Applications: Special issue on Grid Computing: Infrastructure and Applications, August 2003

Alves, A., Arkin, A., Askary, A., Bloch, B., Curbera, F., Ford, M., Goland, Y., Guízar, A., Kartha, N., Liu, C. K., Khalaf, R., König, D., Marin, M., Mehta, V., Thatte, S., Rijn, D., Yendluri, P., Yiu, A., (2006), Web Services Business Process Execution Language Version 2.0 Public Review Draft, 23rd August 2006, available at http://docs.oasis-open.org/wsbpel/2.0/

Bajaj, S., Box, D., Chappell, D., Curbera, F., Daniels, G., Hallam-Baker, P., Hondo, M., Kaler, C., Langworthy, D., Nadalin, A., Nagaratnam, N., Prafullchandra, H., Riegen, C., Roth, D., Schlimmer, J., Sharp, C., Shewchuk, J., Vedamuthu, A., Yalçinalp, Ü., Orchard, D., (2006), Web Services Policy 1.2 - Framework (WS-Policy), W3C Member Submission April

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

2006, available at http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/

Ballinger, K., Ehnebuske, D., Ferris, C., Gudgin, M., Nottingham, M., Yendluri, P., (2004) Basic Profile Version 1.1, WS-I specification, 8 August 2004, http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-07-21.html

Beckett, D., McBride, B., (2004), RDF/XML Syntax Specification (Revised), W3C Recommendation, February 2004, available at: http://www.w3.org/TR/rdf-syntax-grammar/

Benatallah, B., Dumas, M., and Sheng, Q. Z.,(2003), The SELF-SERV Environment for Web Services Composition, IEEE Internet Computing Jan/Feb Issue, Vol. 7 (1). IEEE Society 2003

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F., (2004), Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation 04 February 2004, http://www.w3.org/TR/2004/REC-xml-20040204/

Burstein, M., Bussler, C., Zaremba, M., Finin, T., Huhns, N. M., Paolucci, M., Sheth, P. A., Williams, S., (2005), A semantic web service architecture, In IEEE Internet Computing, vol 9 issue 5, Sept.-Oct. 2005, 72-81

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D., (2004), Web Services Architecture, W3C Working Group Note, February 2004, available at http://www.w3.org/TR/ws-arch/

Bruijn, J., Bussler, C., Domingue, J., Fensel, D., Kifer, M., Kopecky, J., Lara, R., Oren, E., Polleres, A., Stollberg, M., (2005), Web Service Modeling Ontology (WSMO), WSMO Final Draft February 2005, available at http://www.wsmo.org/TR/d2/v1.1/

Christensen, E., Curbera, F., Meredith, G., Weerawarana, S., (2001) Web Services Description Language (WSDL) 1.1 W3C Note 15 March 2001, available at: http://www.w3.org/TR/wsdl

Chinnici, R., Gudgin, M., Moreau, J.-J., Weerawarana, S., (2003), Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Candidate Recommendation, 27 March 2006, available at http://www.w3.org/TR/wsdl20/

Churches, D., Gombas, G., Harrison, A., Maassen, J., Robinson, C., Shields, M., Taylor, I., Wang, I., (2006), Programming Scientific and Distributed Workflow with Triana Services, Concurrency and Computation: Practice and Experience Special Issue: Workflow in Grid Systems 2006, V(18) n(10), 1021-1037

Czajkowski, K., Ferguson. D., Foster, I., Frey, J., Graham, S., Maguire, T., Snelling, D., Tuecke, S., (2004), From Open Grid Services Infrastructure to WS-Resource Framework: Refactoring & Evolution, Version 1.0, Whitepaper, February 2004.

DeMichiel, L., Keith, M., (2006), Enterprise Java Beans, Version 3.0: EJB Core Contracts and Requirements, Release 8 May 2006, available at http://java.sun.com/

Dubray, J. J., Amand, S., Martin, J. M., (2006), ebXML Business Process Specification Schema Technical Specification v2.0.4, Committee Specification, 13 October 2006, available at http://www.oasis-open.org/committees/documents.php?wg_abbrev=ebxml-bp

Duftler, J. M., Mukhi, K. N., Slominski, A., Weerawarana, S., (2001), Web Services Invocation Framework (WSIF), OOPSLA 2001 Workshop on Object-Oriented Web Services: Supporting the Development, Deployment and Evolution of Web Services, October 2001 - Tampa, Florida, USA

Duigou, M., (2006), JXTA v2.0 Protocols Specification, IETF Working Group Draft Specification, August 2006

Elenius, D. and Ingmarsson, M. 2004. Ontology-based service discovery in p2p networks. In Proceedings of the First International Workshop on Peer-to-Peer Knowledge Management, Boston, Massachusetts, USA, August 2004, CEUR-WS, August 2004, Vol. 108.

Fang, J., Hu, S., Han Y., (2004), A Service Interoperability Assessment Model for Service Composition, In Proceedings of the 2004 IEEE International Conference on Services Computing (SCC’04), Sept. 2004, Shanghai, China

Farrell, J., Lausen, H., (2006), Semantic Annotations for WSDL, W3C Working Draft 28, Sept. 2006

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T., (1999), Hypertext Transfer Protocol - HTTP/1.1, IETF, June 1999

Ivkovic, I., (2001), Improving Gnutella Protocol: Protocol Analysis and Research Proposals, LimeWire Gnutella Research Contest, September 2001

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Gibbons, P., Karp, B., Ke, Y., Nath, S., Seshan, S., (2003), IrisNet: An Architecture for a World-Wide Sensor Web, IEEE Pervasive Computing, V(2), N(4), October-December 2003

Gong, L., (2001). JXTA: a network programming environment, IEEE Internet Computing, May/Jun 2001, Vol. 5 (3), pp. 88-95.

Gramm, M., Ritter, A., Schiller, J. H., (2004). Efficient selection and monitoring of QoS-aware web services with the WS-QoS framework. In Proceedings of 2004 IEEE/WIC/ACM International Conference on Web Intelligence, Beijing, China, September 2004, 152-158.

Hoff, H., Gronmo, R., Skogan, D., Strand, A., (2005), D7 Specification of the Visual Composition Language (VSCL), SODIUM (IST-FP6-004559) Project's Deliverable, June 2005

JXTA Org, (2006) Project JXTA, available at http://www.jxta.org/, accessed 21 December 2006

Kavantzas, N., Burdett, D., Ritzinger, G., Fletcher, T., Lafon, Y.,Barreto, C., (2005), Web Service Choreography Description Language (WS-CDL) ver 1.0, Nov 2005, http://www.w3.org/TR/ws-cdl-10/

Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K., (2004) OWL-S: Semantic Markup for Web Services, W3C submission, Nov. 2004 http://www.w3.org/Submission/OWL-S/

McGuinness, D., Harmelen, F.,(2004), OWL Web Ontology Language Overview, W3C Recommendation, February 2004, available at http://www.w3.org/TR/owl-features/

Medvidovic, N., Rosenblum, D., Gamble, R., (1999), Bridging Heterogeneous Software Interoperability Platforms, Technical Report, USC-CSE-99-529, Center for Software Engineering, USC, November 1999

Microsoft Corporation (1996), Distributed Component Object Model Protocol-DCOM/1.0, draft, November 1996, available at http://www.microsoft.com/Com/resources/comdocs.asp

Murer, T., Scherer, D., Wuertz, A., (1996), Improving component interoperability information, Proceedings of Workshop on Component-Oriented Programming (WCOP'96) at 10th European Conference on Object-Oriented Programming (ECOOP'96), pp. 150–158, dpunkt, July 1996

Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmer, M., Risch, T., (2001), EDUTELLA: A P2P Networking Infrastructure Based on RDF, Edutella White Paper, November 2001, available at http://edutella.jxta.org/reports/edutella-whitepaper.pdf

Nezhad, H.R. M., Benatallah, B., Casati, F., Toumani, F., (2006), Web service interoperability specifications, IEEE Computer, vol. 39 no. 5, May 2006, pp 24-32

Newmarch, J. (2005) UPnP services and Jini clients. In Proceedings of the 2005 Conference on Information Systems: Next Generations, ISNG 2005, Las Vegas, Nevada, USA.

Oriol, M. (2002) Peer Services: From Description to Invocation. In Proceedings of the 2002 International Workshop on Agents and Peer-to-Peer Computing, July 2002, Bologna, Italy, Springer Berlin / Heidelberg, LNCS 2530/2003, 21-32

Pantazoglou, M., Tsalgatidou, A., and Athanasopoulos, G. (2006), Discovering Web Services and JXTA Peer-to-Peer Services in a Unified Manner, In Proceedings of the 4th International Conference on Service Oriented Computing, December 2006, Chicago, USA, Eds. A. Dan and W. Lamersdorf, 104-115

Papazoglou, M., Krämer, B., Yang, J., (2004), Leveraging Web Services and Peer-to-Peer Networks, Book Chapter in Advanced Information Systems, Feb 2004, Springer Berlin / Heidelberg 2681/2003, 485-501

Pautasso, C., Alonso, G., (2004) "From Web Service Composition to Megaprogramming" In Proceedings of the 5th VLDB Workshop on Technologies for E-Services (TES-04), Toronto, Canada, August 29-30, 2004

Qu, C., and Nejdl, W. (2004) Interacting the Edutella/JXTA Peer-to-Peer Network with Web Services. In Proceedings of the 2004 Symposium on Applications and the Internet, IEEE Computer Society.

Ryan, N., (2005), Smart Environments for Cultural Heritage, In Proceedings of 24th International Symposium on Reading the Historical Spatial Information in the World, pp 17-33, Kyoto, Japan, February 2005

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Ruiz, A., Corchuelo, R., Martín,O., Durán, A., Toro, M., (2000), Addressing Interoperability in Multi-Organisational Web-Based Systems, Proceedings of the 2nd ECOOP Workshop on Object Interoperability (WOI'2000), Sophia Antipolis, France, June 12, 2000

Siegel, J., (1996), CORBA Fundamentals and Programming, Wiley, 1996

Strang, T., Linnhoff-Popien, C. (2003), Service Interoperability in Ubiquitous Computing Environments, Proceedings of International Conference on Advances in Infrastructure for Electronic Business, Education, Science, Medicine, and Mobile Technologies on the Internet (SSGRR2003w), L'Aquila, Italy, January,2003

Sun Microsystems, (2005), "JXTA v2.3.x: Java Programmers Guide", available at: http://www.jxta.org, April 2005

Tanenbaum, A. S., (2003), Computer Networks, Prentice Hall, fourth edition, 2003, ISBN: 0-13-066102-3

Tsalgatidou, A. Athanasopoulos, G., Pantazoglou, M., Pautasso, C., Heinis, T., Gronmo, R., Hoff, H., Berre, A-J., Glittum, M., Topouzidou, S. (2006) Developing Scientific Workflows from Heterogeneous Services, ACM SIGMOD-RECORD v(35) n(2) 22-28, June 2006

Tsalgatidou, A., Athanasopoulos, G., Pantazoglou, M., (2005), Generic Service Model Specification, Technical Report, available at: http://www.di.uoa.gr/~gathanas/TR/gesmo-1.0-report.pdf

Vallecillo, A., Hernandez, J., Troya J., (2000), Woi'00: New issues in object interoperability, LNCS (1964): ECOOP'2000 Workshop Reader, pp. 256–269, Springer

Vogels, W., (2003), Web Services Are Not Distributed Objects, IEEE Internet Computing, Nov.-Dec. 2003

WSIF Org, (2006), Web Service Invocation Framework, accessed at 21 December 2006, available at http://ws.apache.org/wsif/

ENDNOTES

1 Sodium project http://www.atc.gr/sodium/

2 SeCSE project http://secse.eng.it/

3 The visual language used in this example is the Visual Service Composition Language developed in the SODIUM project (Hoff, 2005).

4 More details on the abstract XML elements and attributes can be found at (Fallside, 2004)

5 WSIF leverages WSDL4J to support the parsing and the in-memory representation of WSDL information elements. However, WSDL4J supports only WSDL 1.1 descriptions. Furthermore according to WSDL 1.1 extensions may be provided only to the Service, Port and Binding elements.

6 The OMG UML Profile and Metamodel for Services (UPMS) RFP document can be retrieved from the following link: http://www.omg.org/cgi-bin/doc?soa/06-09-09.pdf

Aphrodite Tsalgatidou is a permanent member of the academic staff at the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens since 1999. She holds a diploma degree in chemistry from the same institution and an MSc and a PhD in computer science from the University of Manchester, England. Previously she was with the Hellenic Telecom Organization and a research director in a private company. Aphrodite is the director of the S^3 Laboratory (www.s3lab.com) *which pursues research in service engineering, software engineering and software development; she has participated in and/or managed a large number of projects in these areas and has published relevant papers. Her current research focuses on service interoperability, service discovery and composition, peer-to-peer systems and business process management and interoperability.*

*George Athanasopoulos is a researcher and member of the S^3 Laboratory (*www.s3lab.com*) and a PhD-candidate at the Department of Informatics and Telecommunications of the National and Kapodistrian*

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

*University of Athens (*www.di.uoa.gr*). He holds a diploma from the Department of Computer Engineering and Informatics of the University of Patras* (www.ceid.upatras.gr) *and has participated in various research and development projects during his work experience. His research interests include service-oriented computing with a focus on the context-adaptable composition of heterogeneous services comprising various types of services such as Web services (either stateless or stateful), P2P services, sensor and actuator services, etc. Additional research interests are related to distributed systems and software architecture modeling as well as modern programming methodologies.*

Michael Pantazoglou is a PhD student and research scientist at the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens. He holds a diploma degree in informatics and telecommunications from the same institution. Being a member of the S^3 Laboratory, he has participated in several research and development projects. His current research interests include service-oriented technologies with a focus on service discovery, semantic technologies, P2P computing, and the application of information retrieval and natural language processing techniques in data management.

Copyright © 2008, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.