IMPLEMENTATION OF AGENT-BASED COMPOSITION OF SEMANTIC WEB SERVICES WITH WSIG FOR MANET

MONIKA¹, YASHPAL SINGH² & S. NIRANJAN³

¹Student, PDM College of Engineering for Women, Bahadurgarh, India
²Assistant Professor, PDM College of Technology & Management, Bahadurgarh, India
³Professor, PDM College of Technology & Management, Bahadurgarh, India

ABSTRACT

Semantic Web services are the extension of the existing Web Service with the well-defined meaning. The Composition of Semantic Web Service is defined as a composite service, a service chain as a set of such general services (atomic or composite) working together to offer a value-added service. This paper describes how the composition of the web service are done and also describes how the JADE add-on WSIG helps to allow agents and web clients to invoke the composite web service.

KEYWORDS: Semantic Web Service, JADE Agents, Composition Web Service, Integration Gateway, Ontology

INTRODUCTION

A Web Service is a software system identified by a URI whose public interface and binding are defined and described by XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by Internet protocols.[1] The growing trend in software architecture is to build platform-independent software components, called Web Services. Applications are to be assembled from a set of appropriate Web Service and no longer be written manually. It has since evolved to cater to governments, businesses, and individuals to make their data Web accessible. However, a large proportion of today’s data on the Web are “understandable” only to humans or custom developed applications. The Semantic Web is the solution which gives data that is understood by machine and the discovery of data is more relevant and less time consuming. Semantic Web is defined as an extension of the “existing” Web in which information is given a well-defined meaning. The ultimate goal of the Semantic Web is to transform the Web into a medium through which data can be shared, understood, and processed by automated tools. The development of enabling technologies for the Semantic Web is the priority of various research communities. One key technology is the emerging concept of Web services. Simply put, a Web service is a set of related functionalities that can be programmatically accessed through the Web.

We identify two types of services: simple and composite. A Simple service is an Internet-based software component that does not rely on other web services to fulfill user requests. A composite service a service chain is defined as a set of such general services (atomic or composite) working together to offer a value-added service. In both case the resulting composition should appear to the user as a single entity with which to interact. Basically, a web service composition is defined as a set of web services which interact according to certain logical rules. These logical rules specify the composition pattern (serial, parallel, etc.) which describe the execution order of the services involved in the composition, and the connection flow established for the data dependences between services.

Composing web services requires the description of each service so that other services can interact with it. The
language for describing operational features of web services, in most all composition languages, is WSDL. In treating a composition as an atomic web service, however, this composition should also carry a WSDL describing the entire functionality of the composition. In these composition descriptions only the abstract description is present, without details on concrete implementation: things such as access protocols or location points.

Web Services are fast emerging as the dominant means for connecting remotely executing programs via well established internet protocols and commonly used machine readable representation.[2] This paper presents a formulation of how Web services, when enhanced with semantic service descriptions, can be more effectively manipulated by software agents, and how composed agent services are worked for fulfill the user requirement. To provide FIPA compliant agents with the means to interact with Web services, and vice versa, a Semantic Integration Gateway is introduced. This Gateway is an iterative development of the Web Services Integration Gateway [3] available as a component of the JADE agent development framework (v3.3). The Ontology is defined for services which providing the means for software agents to access the web service in the composition of the services. In this perspective, identifying a means of connecting agents and Web services and then agent gives response to the user request after selection of the correct one in the composite service. Mobile agent technology provides a new way of communication over heterogeneous network environment. A number of advantages have been proposed and identified which includes: efficiency and reduction of new traffic, asynchronous autonomous interaction, interaction with real-time entities, local processing of data, support for heterogeneous environment and having robust and fault tolerant behavior.[12]

BACKGROUND & RELATED WORK

Background

Web Services

The concept of Web services has garnered a lot of attention and interest in the software industry because of their potential to provide seamless application interoperability. Different organizations define web services differently. IBM defines Web services as “Web services are a new breed of web application. They are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web services perform functions, which can be anything from simple requests to complicated business processes. Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service”. The Web service framework follows a Service Oriented Architecture (SOA). It uses three basic XML based standards WSDL, UDDI and SOAP. WSDL i.e. Web Service Description Language is used to describe the inputs, outputs and the invocation information of a Web Service. These descriptions are then published or advertised in a registry called UDDI (Universal Description Discovery and Integration). UDDI also provides search mechanisms to find or discover these Web service descriptions. Once the required Web service description is found, the Web service can be invoked using SOAP (Simple Object Access Protocol). Figure 1 shows the SOA for Web services and interactions among its different components.[4]
Implementation of Agent-Based Composition of Semantic Web Services with WSIG for MANET

WSDL (Web Service Description Language)

The Web Service Description Language is a standard for describing a Web service. It is an XML grammar for specifying properties of a Web service such as what it does, where it is located and how it is invoked. It can be compared to HTML (Hypertext Markup Language) in the normal WWW which is used to describe a web page. A WSDL document defines services as a collection of network endpoints, or ports. In WSDL, the abstract definition of endpoints and messages is separated from their concrete network deployment. This allows the reuse of abstract definitions. A WSDL document uses the following seven elements in the definition of network services.

- **Type**: A container for data type definitions using some type system
- **Message**: An abstract, typed definition of the data being communicated
- **Operation**: An abstract description of an action supported by the service
- **Port Type**: An abstract set of operations supported by one or more endpoints
- **Binding**: A concrete protocol and data format specification for a particular port type.
- **Port**: A single endpoint defined as a combination of a binding and a network address.
- **Service**: A collection of related endpoints.

UDDI (Universal Description Discovery and Integration)

UDDI is a registry or a directory where Web service descriptions can be advertised and discovered. The person or organization that provides a Web service is called “Service Provider”. A service provider publishes the service description i.e. the “Service Advertisement” in the UDDI. A “Service Requester” i.e. a person or an organization can “request” a Web service by using the search mechanisms provided by UDDI. A Web service can be found or discovered in UDDI either by browsing the categories or by using a keyword-based query mechanism. Figure 1 summarizes the interaction between service provider, service requester and UDDI registry.

SOAP (Simple Object Access Protocol)

SOAP is an XML-based messaging protocol. It defines a set of rules for structuring the messages being exchanged. It can be used for either one way messaging or request response type communication. It does not depend on any specific transport protocol but the standard HTTP (Hypertext Transfer Protocol) is most popular.

ONTOMETRY

We can say an ontology provides -

- A common vocabulary of terms
- Some specification of the meaning of the terms (semantics)
- An agreement for shared understanding for people and machines.

Related Work

There are several articles related to automated Web service composition. Ontology-based Web Service Composition [5] aims to generate a composite service out of semantically described existing services. In this work, the possible automatic compositions are obtained through interface-matching, which checks semantic similarities between
interfaces of individual services. Different services can be integrated to satisfy user requirement.

Rao et al. [6] Propose GraphPlan algorithm for generating a composite web service. But he uses a lot of interaction from user which drifts his project towards manual composition. His approach does not consider input and output schema. This schema is important in ontology mapping. Also, some services can be compatible without having input and outputs. His approach uses one input to a service. However, in our approach, a service can get input more than one service.

SWORD[7] proposed a rule based system for composition of web service. His approach is not suitable for heterogeneous and changing web services. He emphasizes on composition and did not take into account input and output mismatch. In our technique, system checks the input parameters and more than one parameter can be accepted by a service. Like, input and output mismatch can also detected and provide error report.

Thakkar et al[9] consider dynamic composition of Web services using mediator-based agent system architecture. The mediator takes care of user queries, generates wrappers around information services and constructs a service integration plan.

Paolucci et al[8] evaluate a broker for constructing OWL-S Web services. They also identify some drawbacks of the current OWL-S specification and propose a workaround for the problem. Sycara et al [10] describe a methodology for constructing composite services written in DAML-S. Similarly, the preceding article view ontology-based technique which describe the inputs/outputs and conditions or effects of particular services.

**PROPOSED WORK**

The ontology is defined for the composite web service which helps to software agents how and which web service should be called to provide the correct response to the client request. The ontology contains the meaning or definition of the input which is given to agent and which is provides by the user. This paper contains the domain-specific ontology which only describes the data related to only one domain.

The WSIG add-on of JADE is used for the integration of agent and web services. The Gateway offers automatic, bidirectional operation allowing both FIPA compliant agent services and Web services to be registered with it. Agent services and Web services can thereby publish their service descriptions to consumers outside their normal operational domain. The Gateway can then intercept calls to these registered services allowing agents to invoke Web services and vice versa by transforming message encodings and creating service access endpoints. The defined Semantic Web Service with Agents using Web Service Integration Gateway are accessed in the Agent-Based MANET architecture which allows to access the web services in the network with the http protocol. The mobile phones and any other system present in the network can easily access the web service are designed with the Agents and WSIG add-on of JADE.

**WEB SERVICE INTEGRATION GATEWAT FOR SEMANTIC WEB SERVICES**

**Assumptions**

The following assumptions were made when designing the Gateway architecture:

- All agents are assumed to be FIPA compliant and capable of communicating with FIPA-ACL encoded messages.
- All Web services operate using the standard Web service stack consisting of WSDL for service descriptions, SOAP for message encoding and UDDI for directory services.
- All agent services ontologies are exposed in WSDL-S format.
• The Gateway is registered as an agent service in FIPA Directory Facilitators. It is not registered directly in UDDI directories, as its functionality is exposed outside the agent platform via a Web server.

• All invocation related interactions between the Gateway and agents use ACL encoded FIPA-Request and FIPA-Inform per formatives. This is essentially all that is needed to invoke standard request-response Web services.

**Gateway Architecture**

The Gateway is to provide an automatic means of mapping the functional and representational dependencies associated with the invocation of agent services onto those associated with Web services and vice-versa.

The WSIG Agent is the gateway between the Web and the Agent worlds and is responsible for:

• Forwarding agent actions received from the WSIG Servlet to the agents actually able to serve them and getting back responses.

• Subscribing to the JADE DF to receive notifications about agent registrations/deregistrations.

• Creating the WSDL corresponding to each agent service registered with the DF and publish the service in a UDDI registry if needed.

![Gateway Architecture Overview](image)

The process responsible for intercepting DF registrations/deregistration and converting them into suitable WSDLs. This process is completely carried out by the WSIG Agent. The process responsible for serving incoming web service requests and triggering the corresponding agent actions. This process is carried out jointly by the WSIG Servlet (performing the necessary translations) and the WSIG Agent (forwarding requests to agents able to serve them).

The Gateway consists of several components; each linked either directly or indirectly to two registries - a Directory Facilitator (DF) for storing agent service descriptions and a UDDI service for storing Web service descriptions. The DF is the same instantiation as that used by all agents within the agent platform and is not visible outside the platform. The UDDI belongs solely to the Gateway, is visible internally to the Gateway and externally to Web services and Web service clients, but not directly to agents. The Gateway itself is registered as an agent service in the platform DF. Each registry exposes the standard operations: Registration, deregistration, modification and discovery. Use of these registries is central to the Gateway architecture. Any service description that is registered with either the DF or UDDI registry is
automatically translated into an entry for the other. This duplication ensures that any registered Web service is visible to agents via the DF and any registered agent service visible to Web service clients via the UDDI.

![Figure 3: Agents and Web Service Description Linking and Registration](image)

JADE agents publish their services in the DF (Directory Facilitator) providing a structure called DF-Agent-Description and defined by the FIPA specification. A DF-Agent-Description includes one or more Service-Description each one actually describing a service provided by the registering agent. A Service-Description typically specifies, among others, one or more ontologies that must be known in order to access the published service. The actions the registering agent is actually able to perform are those defined in the specified ontologies.

In order to expose an agent service as a web service it is sufficient to set the wsig property to true in the properties of the Service-Description at DF registration time as below

```java
ServiceDescription sd = new ServiceDescription();
....... 
sd.addProperties(new Property("wsig", "true"));
```

Each Service-Description including the wsig property set to true will be mapped to a WSDL. All actions defined in the ontologies specified in the Service-Description will be mapped to WSDL operations.

![Figure 4: Agent Gateway Architecture](image)

The Agent Gateway is an initiative toward achieving dynamic and seamless integration of FIPA compliant software agents and WSA compliant Web services. The Gateway is a middleware layer between the multi agent system and Web services facilitating integration without requiring alteration to existing specifications.

**Agent Gateway Architecture**

The Service Discovery converter enables FIPA agents and Web services to search for one another. Software
agents can discover Web services via UDDI registries and conversely, Web service clients can perform searches for agents and agent services from agent registries such as the FIPA DF. Specifically, the 'UDDI search query to DF search query converter' enables the discovery of Agent platforms by Web service clients via the DF. The 'DF search query to UDDI search query converter' enables the discovery of Web service platforms by software agents via UDDI. The Service Description converter enables service publishing among Software Agents and Web services. Software Agents can publish services in Web Services registries such as UDDI and Web Services can be published in Multi Agent Systems service registries such as the FIPA DF. Specifically, the 'WSDL to DF-Agent-Description converter' enables Web services to publish their services in the DF of Agent Platform, and the 'DF-Agent-Description to WSDL converter' enables Software Agents to publish their services in the UDDI of a Web services platform. The Communication Protocol converter component enables service invocation among software agents and Web services. Software agents can invoke Web services and Web service clients can invoke software agents in Multi Agent Systems. Specifically, the 'SOAP to ACL converter' enables Web service clients to invoke Software Agents, and the 'ACL to SOAP converter' enables Software Agents to invoke Web services.

**Gateway Operations**

The Gateway has four principal modes of operation:

- Mapping a newly registered agent service into its counter-part Web service counterpart
- The inverse mapping of (1)
- Invoking a Web Service from an agent and
- Invoking an agent service from a Web service client.

When initiated the Gateway agent starts it first registers with the local agent platform DF.

**Registering an Agent Service**

Service registrations are received by the gateway via the agent platform DF in the agent case or by the Gateway UDDI in the Web service case. The standard JADE DF has been extended to incorporate a monitor that verifies if any agent service registrations are of ACL type 'web-service'. If this is the case, the Gateway agent is informed by issuing a FIPA request message. The Gateway, on reception of this message, initiates a behaviour which first calls the ACL-SL0/tModel-SOAP Codec to translate the ACL message and contained SL0 3encoded service description into a tModel. Once the ACL message has been decoded the value of ontology slot is used to resolve the published location of the WSDL-S service ontology associated with the service (if one is available). This reference is passed to the WSDL-S/WSDL Codec and translated to a WSDL representation of the service description that can be used by Web services to identify the features of the exposed agent service.

Once produced, this WSDL description is published via the Web server and a reference included into the tModel, which can then be registered with the Gateway UDDI. Finally, if this process is successful, the Gateway agent initiates a new external endpoint on the Web server to expose an instantiation of the newly created WSDL service description for invocation by Web services and Web service clients. Similar to registration requests, deregistration and modification requests received by the platform DF of type 'web service' will cause appropriate ACL messages to be sent to Currently the Gateway SL codec is restricted to parsing SL0. the Gateway DF, initiating removal/modification of UDDI entries, WSDL schema and invocation endpoints. Discovery requests received by platform DF require no action by the Gateway.
Registering a Web Service

Web service registrations are received by the Gateway via the Gateway UDDI, which is designed to trigger behaviour within the Gateway agent on reception of a new registration. Subsequently the process path is essentially the inverse of that described in the previous section: First the tModel is passed through the ACLSL0/tModel-SOAP Codec to produce an SL0 version of the service description. Simultaneously, the WSDL description of the Web service is parsed into an WSDL-S representation by the WSDL-S/WSDL Codec, the resulting service ontology published internally to the agent platform via the Web server and a reference to it added to the ontology slot of an ACL message created by the Gateway agent to register the new SL0 service description with the platform DF.

Finally, the Gateway agent updates an internal knowledge base that lists all Web services that have a current registration with the platform DF. This allows the Gateway agent to quickly establish whether an incoming Web service invocation request from an agent is valid without needing to verify against the UDDI. Requests for de-registration and modification are treated similarly, with the former leading to removal of the DF and UDDI registrations. Discovery requests to the UDDI again require no action by the Gateway.

Invocation of a Web Service by an Agent

Under the assumption that an agent has established the identity of the Web service it wishes to invoke by searching its local DF, it sends a standard ACL FIPA Request message to the Gateway containing the identity of the Web service to be invoked and any parameters as properties of the request. Initially a received ACL message is parsed by the Gateway as would a normal agent parse an incoming message. Next, the Web service identified in the message content as being that to be invoked is verified as being registered with the Gateway (against the Gateway agent’s internal list.

If not, an error is returned to the invoking agent; otherwise a SOAP message is constructed using the WSDL service description of the Web service to be invoked. If no response is expected from the Web service this SOAP message is then sent; otherwise, a temporary endpoint is established on the Gateway Web server to receive responses. On reception of the response the incoming SOAP message is parsed into an ACL FIPA Inform message and sent to the invoking agent. Finally, the temporary endpoint is removed.

Invocation of an Agent Service by a Web Service Client

For a Web service client to invoke an agent service, the Gateway must first expose that service as a Web service endpoint (as described previously). If this is the case, the incoming SOAP message from the Web service client is parsed into an ACL FIPA Request message containing the SL0 encoded service request and sent to the agent containing the service. Error messages returned from the invoked Web services are treated separately to be invoked.

Once the receiving agent processes the service invocation, if required, it will return a response to the Gateway as a FIPA Inform message. This response is then parsed by the Gateway into a SOAP message and returned to the invoking Web service client. The Gateway agent uses the FIPA conversation id ACL message parameter to track conversation contexts between itself and any agents participating in requests for, or delivery of service. This helps to ensure that messages are correctly correlated.[2]. The composition technique aims to find the optimal composition of service considering semantic matching of parameters. Generally, our agents-based Web services composition mechanism consists of three sequential phases[2],[11]:

Service Registration

This is the way to register the agent service.
Implementation of Agent-Based Composition of Semantic Web Services with WSIG for MANET

```java
try {
    DFSerive.register(this, dfad);
}
catch (Exception e){
    log.error("Problem during DF registration", e);
    doDelete();
}
```

**Figure 5: Dynamic Service Agent Registered and Started**

**Figure 6: Agents are Shown as Registered**

**Construction of the Service Graph**

In the graph construction phase matching input output of services are logically connected in a way that they form a potential workflow. In our example DynamicService contain the logic how to access the other two web services.
Service Discovery Together with Composition

**OUTPUTS**

- When input provide by user as ZIPCode. The Agent registered for the Dynamic Web Service is invoked and the invoked Agent calls the Web Service which takes ZIPCode as input and provides output as weather information for the corresponding ZIPCode. The Web Service invoked and response is obtained and the response is provided to Agent and Agent respond to the user.
• When input provide by user as City name and Country. The Agent registered for the Dynamic Web Service is invoked and the invoked Agent calls the Web Service which takes City name and Country as input and the response is provided to Agent and Agent respond to the user.

![Figure 10: City & Country Service Responded Output](image)

• When input provide by user as Longitude and latitude. The Agent registered for the Dynamic Web Service is invoked and the invoked Agent calls the Web Service which takes longitude and latitude as input and provides output as Sun Set & Rise information for the corresponding input. The Web Service invoked and response is obtained and the response is provided to Agent and Agent respond to the user.

![Figure 11: Sun Set & Rise Service Responded Output](image)

• When input provide by user as Email ID. The Agent registered for the Dynamic Web Service is invoked and the invoked Agent calls the Web Service which takes Email ID as input and provides output as Email ID is exist and valid. The Web Service invoked and response is obtained and the response is provided to Agent and Agent respond to the user.
WEB SERVICE ACCESS IN MANET

The Web Service can be access by the client using the web access pages. The composite web services are running on the server side and the client use the web site or web page to access the web service over the network. The http protocol is used to access the web service and the port number on which the server running. The Agent based Mobile Ad-hoc Networks architecture which was already proposed has been taken as base to test the agent- based web services whether these are working properly or not. As the result, these web services providing the correct result in the Agent Based Mobile Ad-hoc Network architecture.

![Diagram of Agent Based Mobile Ad-Hoc Network](image)

**Figure 13: Architecture for Agent Based Mobile Ad-Hoc Network [13]**

The web service access in the network is shown in below figure 14. The Web Service is running on the main server side which have the composite web service which decide to which web service is called as the input given by the user or client. The Web Service can be accessed by laptop or mobile phone using http protocol. The User passes the input. The Agent gets the input provide by the user and invokes the composite web service which then decide to which web
service the call is transferred. The appropriate web service is invoked and the result is created and the output given back to agent which gives response to the user request. This is the whole process how the agent and web service communication takes place. This communication is worked in the above mentioned Agent-Based Mobile Ad-hoc Network where the mobile and other systems are tested to access the web services.

Server running and responding to the client

![Server running and responding to the client](image)

CONCLUSIONS

This paper uses the Semantic concept with Web Service Integration Gateway that enables seamless, bidirectional and automatic connectivity between agent services and Web services. The Gateway supports the cross registration of both agent services and Web services, i.e. agent services can be registered in UDDI and both WSDL and WSDL-S Web service descriptions in FIPA DF repositories. From this point, any registered Web service can be invoked by an agent by addressing an invocation call to the Gateway. Equally, as the Gateway exposes registered agent services as endpoints using an internal Web server, Web services can also make invocation calls into an agent platform. The Gateway makes use of WSDL-S service ontologies to represent the native service descriptions of both agent services and Web services in a common, semantically expressive encoding. This allows agents to compose mixed workflows, as WSDL-S process models, consisting of both agent services and Web services as required. The web services are exposed in the Agent based MANET architecture which are accessed by the mobile phones and the other systems and the result of these this proposed work worked accurately with this Agent-Based MANET architecture.

REFERENCES


