

## Chapter 13

# BUILDING SEMANTIC BUSINESS SERVICES

Sanjay Chaudhary, Zakir Laliwala and Vikram Sorathia.

*Dhirubhai Ambani Institute of Information and Communication Technology, Post Bag No. 4,  
Near Indroda Circle, Gandhinagar 382 007, Gujarat, India.-*

*sanjay\_chaudhary@da-iict.org, zakir\_laliwala@da-iict.org, vikram\_sorathia@da-iict.org.*

## 1. INTRODUCTION

This chapter aims to provide comprehensive exposure to various issues involved in the development of Semantic Web services based Business Process Orchestration. Marketing of Agricultural Produce is selected as the problem domain. The present discussion covers various topics from understanding the problem up to the identification and resolution of the implementation issues. At each stage, an attempt is made to provide a brief background, current research trends, available techniques, selection of tools and details about implementation steps. As an outcome, the reader shall gain the required skills and sufficient level of familiarity of current standards and research in each area.

In section 1, the reader is introduced to the evolution of Agricultural Marketing in India, and the reforms that are planned for implementation. Section 2 discusses a trading use case in the future market followed by a section dedicated to explain the implementation challenges. A discussion on development lifecycle describes the implementation steps for the proposed system. In the subsequent sections, detailed and step-by-step development procedures are provided with comments on relevant standardization, research approaches and tool-sets.

**Agricultural Marketing** Marketing of agricultural produce is a complex task involving various stack holders, products and business scenarios. In a developing country like India, this activity is influenced by local, socio-economic and cultural characteristics. Evaluating the business processes at

regional or national scale reveals diversity in products, terminology and processes involved to perform complete business activities. While other complex but well-defined business processes are experiencing benefits of services driven e-business; the 'marketing of agricultural produce' has remained untouched by this revolution. Government of India is now planning to introduce agricultural marketing reforms to streamline trading processes involved in all markets throughout the nation. The legal framework is being duly formulated, yet unavailability of proper underlying IT infrastructure will continue to inhibit the implementation and penetration of such technological advancement amongst the users. In absence of such capabilities, the conventional trading transactions will continue to provide meager benefits to a farmer who loses a better price in other potential market, or a wholesaler - who might have got the desired quality product at a lower cost directly from the farm. There is a need to develop affordable and reliable solution that links all the actors involved in the system and provide an environment for a competitive business.

**Evolution of the Agricultural Marketing Process in India** Beginning with the era of barter system, where goods were exchanged for goods, or goods were exchanged for services, through the weekly Bazaar, to more organized Mandi and market yards of the present and the trends towards realizing the reforms in the future markets. This way, the process has evolved to a very matured and complex level (Sreenivasulu V, et al 2001). An informal gathering of the people at a designated place and time has remained a valid model for quite a long time. Today, wholesale spot markets and derivative markets are emerging as hubs for agricultural marketing business (Thomas S, 2003). The trade in this market is heavily influenced by local, socio-economic and cultural characteristics. This is the reason why same product may have different prices at different market yards. Yet the producers have no choice to search for the best available price and forced to sell their products in the local market. Inhibitive transport and storage costs also play a vital role, apart from the urgency to sell the perishable products. Buyers and wholesalers on the other hand face difficulties to purchase desired quality of products at competitive prices. The Model Act (Ministry of Agriculture, Govt. of India, 2003) is formulated to bring reforms in the Agricultural Marketing Process. Additional responsibilities are assigned to the existing Agricultural Produce Market Committee (APMC) to realize the reforms having following objectives:

- To promote setting up of privately-owned markets
- To promote direct sale and contract farming
- To provide transparency in trading transactions
- To provide market-led extension

- To ensure payment on the same day
- To enable value addition in agricultural produce by promoting processing

## 2. TRADING USE CASES IN FUTURE MARKET

As indicated earlier, the Act, a typical trade can span across the markets located at various places. Privately owned markets will be allowed and food processing and other related industries will be encouraged to trade directly with the farmers. Contract farming will also be formalized according to the provisions of the Act. Hence the trading in such a competitive market will be more complex than that of in the existing scenario.

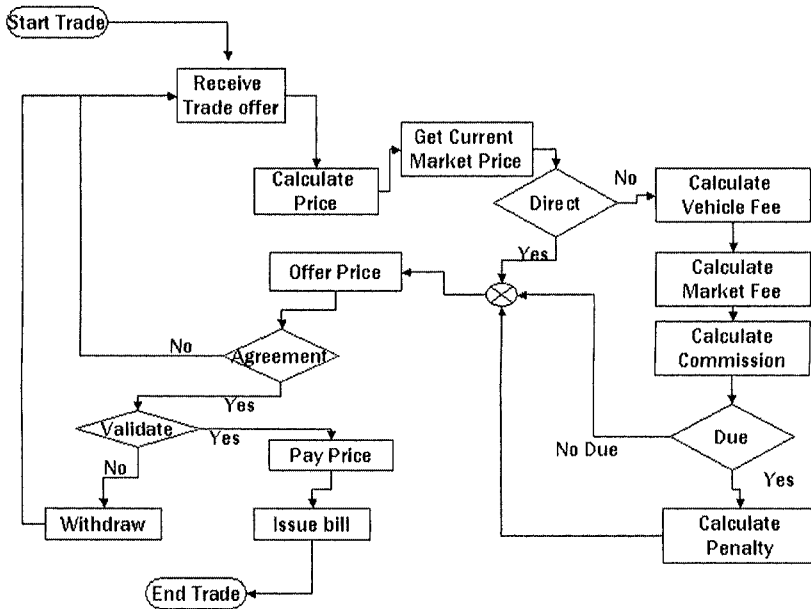


Figure 13-1. Workflow in the Trading Use Case

**Execution of a Trade** This section provides a brief account of a use case to sell an agricultural produce. A seller can come to the market place or in case of a direct marketing; a farmer can express his intention to start a trade of agricultural produce. An authorized market functionary carries out measurement and grading of the produce and collect fees in case a vehicle is used to transport the produce. Price of the produce can be set by tender bid, auction or any other transparent system. In case of direct sale, a seller is

exempted to pay any market fee or commission; otherwise the market fee is imposed on the seller. Only license holders are allowed to carry out any trade in the market area. If they fail to pay any fees to the APMC or fail to pay the agreed-upon price of the purchased good, the APMC may cancel the license of the trader. If the trade is carried out by license holders in a manner explained above, the bill will be issued and the transaction will be recorded in the APMC database. Figure 13-1 represents a simplified workflow capturing few aspects of a typical trade. The trade starts with the intention of the trader to sell a particular agricultural produce. The price is set by the auction or any other transparent system as defined in the Act. Once the agreed upon price is received, it is published for the traders.

### 3. IMPLEMENTATION CHALLENGES

After the implementation of the Act, it is being envisaged that the trading will span across the nationwide markets. To develop and deploy the services to support this vision comes with the challenge of its inherent heterogeneity. In the nation of diverse cultural values, the terminology used in trading varies from place to place. The software and hardware infrastructure at this scale also cannot be expected to be homogeneous throughout. The consequence of this scenario introduces many difficult challenges.

#### 3.1 Understanding the Process

The reforms suggested by the Act consist of details regarding the trade in legal terms. The Act introduces many new concepts with comprehensive definitions and originates a new vocabulary in the domain. From the software development aspect, the developer must be familiar with the vocabulary to design and execute specific tasks in a prescribed manner. The challenge is to convert the statements of the Act to appropriate formal representation that can be utilized as a benchmark for the further software development effort. For this reason, the domain knowledge representation is a prerequisite with consideration from the following three aspects:

**Terms Used in the Act** To understand the problem, consider Definition 1, defining an Agriculturist in the context of the Model Act.

**Definition 1 (Agriculturist)** *means a person who is a resident of the notified area of the market and who is engaged in production of agricultural produce by himself or by hired labor or otherwise, but does not include any market functionary.*

This definition is made up of some concepts like: "Notified Area", "Agricultural Produce", "Market Functionary". These words are also

properly defined in the act along with other provisions. Yet, the meaning of these terms may not be intuitively clear to the person responsible for the development. Hence, a proper methodology is required to represent all the concepts and relations among terms.

**Terms Used by the Traders** The act expresses various agricultural produces at a level of abstraction. In the real world, one specific agricultural produce is referred with various terms in various regions. The general terms for the produce are further attributed specific terms on the basis of the quality, type, processing and other parameters. While trade is underway, the persons involved in the process usually dwell upon such attributes that are not expressed in the act. One can also predict the change in the terms to attribute the specific crop changing from place to place. Hence a representation is required that covers all the *de facto* terms and concepts prevailing in the domain.

**Terms Used by the Developers** The developers responsible for generating services use specific terms for expressing the functionality of the methods and variables used in the discrete functionality. When, the trade is expected to span across many geographical locations, it is likely that different developers have followed different naming conventions for developing specific service blocks. Hence at the time of composing the services the same terms denoted with different symbols needs to be resolved.

### 3.2 Definition of Business Model

The Act represents only the regulatory aspect of the trading. The implementation details will vary based on the quality of the software development process being followed by the individual developers. Defining a commonly agreeable business model is a difficult problem in case of agricultural domain. For instance, the regulated and privately owned markets may differ in implementation details from Quality of Services point of view (Cardoso, J. and Sheth A 2004). The existing business models may prove insufficient or conflicting. Hence to arrive to a clear definition of a business model is a difficult challenge.

### 3.3 Making Business Web Enabled

It is relatively easy to expose business functionalities over the web with the help of Web services. Yet providing complete business functionality that utilizes several Web services is still a considerable challenge for a large-scale integration. Consistent availability of dependable ICT infrastructure across markets can be an issue especially in under-developed and remote regions of the country.

### **3.4 Access to Information for the Functionaries**

Most of the Functionaries of this e-trading system are dependent on timely and relevant information to carry out informed decisions (Chaudhary S, et al 2004). Current mechanisms for communicating the information include telephone calls, radio, black board and public address systems. Yet it is evident that the scope of all the above-mentioned information communication technology is limited to the local market and is insufficient to meet the information need of traders in future context.

### **3.5 Access to Instruments for End-Users**

Under a futuristic assumption, the end-users may range from a farmer possessing a PDA with a limited computing capability up to an export enterprise that might have implemented enterprise level information system. But in a proposed development cycle they all are clients to the business orchestration services. Hence it is a difficult challenge to cater the need of clients with varying computing capability.

### **3.6 Negotiation Support**

The biggest challenge is the terms used by the farmers and traders, which can be inhibitive in automating the negotiation process. With the advent of distributed e-business systems, interested parties can engage in real-time or near-real time negotiation process. Negotiation Support Systems are in place for more than twenty years, yet enabling the support for negotiation in e-trading will continue to be a difficult challenge.

## **4. DEVELOPMENT**

With the identification of the implementation challenges, this section proposes a development lifecycle for realization of the system. It has been observed that many business services require certain common functionalities that might be implemented and hosted by a separate organization. If Service Orientation is followed for the development; it is likely that such business services are implemented as Web services. There is a possibility that other organizations can be enabled to utilize these loosely coupled services to meet their requirements. Hence, it is quite possible in today's scenario that for a complete business process, an enterprise can make the use of several of such open, reliable and interoperable services (Lu L, Zhu G, Chen J, 2004).

From the service provider's point of view, as these services can be joined together to constitute a complete business process, it has become essential to make provisions for efficient integration with heterogeneous client environments. See (Piccinelli G, Stammers E, 2002) for an impressive historical overview of merger of Business Services with IT services. As the pool of available service grow large, the selection of appropriate services becomes difficult due to heterogeneity at various levels. Semantic Web Process Lifecycle (Cardoso, J. and Sheth A 2004) is proposed to address these integration issues. The development approach being proposed here is based on similar philosophy but with more emphasis on implementation aspects of the Semantic Web services based Business Process. Figure 13-2 represents the lifecycle with eight steps described in the discussion.

#### **4.1 Model Act**

The legal acceptance of the proposed Act by the Government is the starting point of the lifecycle of this development. The provisions in the Act clearly define various entities involved in the trade. It specifies the role of each entity with specific attributes also including the flow of the trading process. Hence, the Act plays a crucial role in identifying the requirements and to derive business logic of the desired system.

#### **4.2 Development of Business Objects**

Based on the provisions in the Act, respective markets are expected to engage in the development of business objects to implement various business functionalities.

#### **4.3 Exposing Discrete Functionality**

While traders are expected to engage in trade over electronic media, there emerges an important requirement to access a small part of business process hosted by a node. To enable standards based uniform access of such small component; the developed business objects are exposed as Web services. With the help of this, trade will be enabled across the nation by giving access to services developed by individual markets.

#### **4.4 Trading in Market**

Once all the business functionalities in various markets are accessible, the trading in the market will take place. The Act has dealt with abstract

terms like Functionaries, Agricultural Products and related terms but at the time of trading, the individual transactions will involve very specific terms used by the traders. Management of instance data will be required to be addressed. Here the methodology for the consideration of terms used in trading will be required to be defined formally.

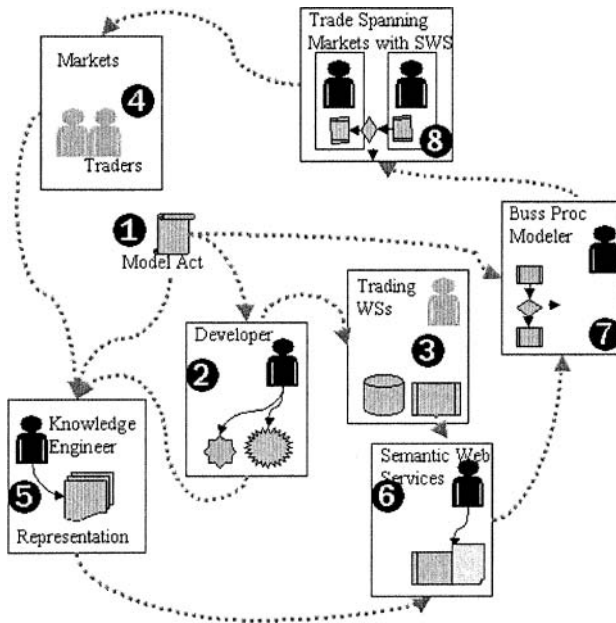


Figure 13-2. Development Lifecycle of Semantic Business Services

## 4.5 Post Compliance Development

After achieving successful compliance of Act in practice, next steps from 5 to 8 as indicated in the Figure 13-2 can be taken as follows:

- **Knowledge Engineering** The knowledge Engineer develops the formal representation of the terms and their relationship as used by traders and developers along with the provisions defined in the Act.
- **Semantic Web service** The formal representation will make it possible to annotate various Web services deployed at different markets. Hence, the concentration will be on semantically enriching the developed Web services to enable appropriate integration.



- **Business Process Modeler** The annotated Web services are utilized by the Business Modeler to design complete business process according to the provisions of the act.
- **Business Process Orchestration Service** The Business Process Orchestration server will accept the requests directly from the traders to initiate and execute the trade (Sorathia V, et al 2005).

## 5. DEVELOPING AGRICULTURAL MARKETING ONTOLOGY

To enable e-trading, one of the important challenges is the utilization of uniform terms across the market. In this section, the development of an ontology that covers all the terms, relations among them and the logical expression of the legal provision are discussed. First the discussion starts with the theoretical aspects covering a methodology to create ontology. This is followed by the current standardization to create the ontology. Next section provides a brief introduction to the W3C standard that has been followed in developing the ontology. An approach is provided for the purpose followed by a step-by-step development of the ontology with the help of selected tool.

### 5.1 Approach

According to the definition of the Agriculturist in the act, any person resident of the notified area, engaged in the production of Agricultural Produce is considered Agriculturist, only if he is not a Market Functionary. This definition uses few terms that also need to be defined clearly. The logical equivalent of the definition can be written as:

```
Agriculturist(X) ←
is_resident_of(X, notified_area) and
is_producing(X, agricultural_produce) and
{not (market_functionary(X))}
```

In the similar manner, the details related to the regulation of marketing of notified agricultural produce can be converted into logical representation. The following list contains few entries for evaluation of a trading instance.

```
selling(seller, X)
buying(buyer, X)
```

```

quantity(X,Q)
is_transported_by(X,head load)
is_less_than(Q,4)
price(X,p)
is_settled_by(P,transparent_system)
is_a(seller, pretty_trader)
is_kind_of(seller, esse_comm_dist_agency)
is_a(seller, auth_fair_price_shop_dealer)
is_a(seller,licensee)
is_kind_of(X,notified_agri_produce)
is_covered_under(X,contract_farming)
is_brought_by(X, licensee)
is_a(current_trade, direct_sale)
is_a(current_trade, ordinary_sale)

```

Once the logical representation is completed, it is converted into standard based representation so that it can be uniformly accessed across the system. There are many standards proposed over a period of time. One of the recent significant standards is OWL and the same is selected for the present experiment.

## 5.2 Step-by-Step Development

There are many tools available to build ontology according to the OWL Standard. Some tools are equipped with the facility of validating the ontology and the reasoning capability to infer new facts from the represented concepts. Some of the leading tools for developing Ontology include Protégé, OilEd, KAON, OntoEdit and OntoStudio. We have selected Protégé as the Ontology builder tool for this experiment. Now we will see the step-by-step instruction to build ontology using Protégé. To design ontology using Protégé the only required prerequisite is the recent version of Protégé with Protégé OWL Plug-in. The recent version of Protégé can be downloaded from the Protégé Web Page<sup>1</sup>. The setup installation program is packaged with the OWL Plug-in, and user can select the installation of this plug-in during the setup. For recent version of the OWL-Plug-in, user can check for the updates at CO-ODE Web Page<sup>2</sup>.

1. **Adding Class** Each class or concept in OWL is considered as a set of individuals. An ontology starts with defining set under the set

<sup>1</sup> Protégé Download Page <http://protege.stanford.edu/download/download.html>

<sup>2</sup> OWL Plug-in for Protégé Download Page: <http://www.co-ode.org/downloads/>

owl:Thing. The present chapter deals with agricultural marketing therefore the concepts in this domain can be appropriately added as subclass of the owl:Thing. In introducing terms as new concepts, the Protégé allows to select different types of classes for appropriate representation. A primitive class is the simplest expression of the domain concept. If not specified, every class being added is added as a primitive class. According to the definition of an Agriculturist, along with other requirements; one specific restriction is that the person should not be a market functionary. This kind of restriction can be covered in OWL representation as follows:

```
<owl:Class rdf:ID="Agriculturist">
  <owl:disjointWith>
    <owl:Class rdf:about="#market_functionary" />
  </owl:disjointWith>
  <rdfs:subClassOf
    rdf:resource="#Agriculture_Market" />
</owl:Class>
```

2. **Adding Property** Each concept generally exhibits a specific functionality. Concepts also possess relationships with other concepts. This feature can be expressed by defining the Property in the ontology. Depending on the context such property can exhibit functional, inverse, transitive or symmetrical relationship.
3. **Adding Restrictions** Many terms in the act are defined using additional terms of the domain. Sometimes definition includes certain constraints to be met to classify the given term. To realize such concepts defined with specific restrictions the OWL has mechanism to define restrictions. The restrictions can be defined with concepts like Universal Existential Quantifiers, cardinality restrictions on the value etc.
4. **Adding Instances** Once classes and the relationships among them are defined with appropriate restrictions, the instance of the class can be added. By opening the *Individuals* tab, the specific class can be selected. The Individual Editor will display all the relevant slots that can be filled to complete the task of adding the individual instance.

For detailed account on step-by-step development of OWL Ontologies, reader is recommended the Practical Guide (Horridge M. et al (2004).

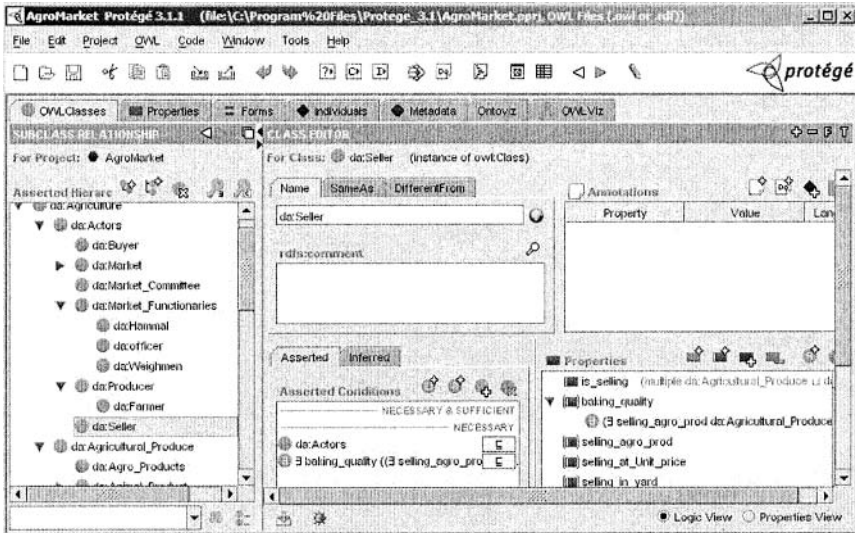


Figure 13-3. Building Ontology

## 6. BUILDING WEB SERVICES

In this section, discussion is focused on creation of Web services that will enable semantic query on the ontology developed in previous section. The following definition of Web services given by (Stencil Group 2001) clarifies the important qualities of Web services that makes it appropriate approach for the given problem.

**Definition 2 (Web Service)** *Loosely coupled, reusable, software component that semantically encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols.*

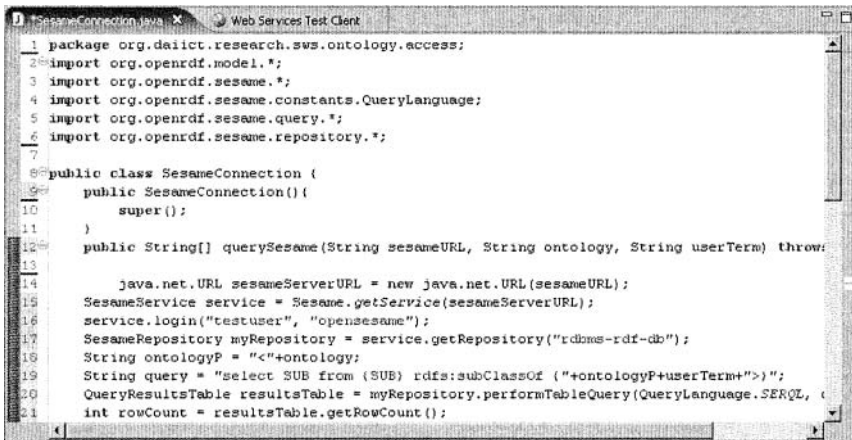
### 6.1 Step-by-Step Development of Web Services

Following section will guide the reader to build Web services to access the ontology stored in a repository.

**Prerequisites** This exercise requires the reader to be familiar with working knowledge of Java and XML programming. Eclipse is used as a primary IDE for the development in this experiment. User is expected to get basic familiarity with the Eclipse development philosophy.

**Hosting Ontology Repository** The Ontology developed in the previous section will be used for annotation of Web service descriptions. Apart from service annotation, it will also help in resolving any ambiguity or in identifying relationship with other terms used by the traders. Ontology Repository is deployed to host the developed ontology that can later be accessible programmatically by various services over standard Internet protocols. Considerable amount of tool is freely and commercially available for hosting the ontology repository. A good repository supports standard ontology formats, standard query languages and capability of persistent storage in popular database products. Sesame is selected to host ontology repository for this experiment. Recent version of Sesame can be obtained from the OpenRDF download page<sup>3</sup>. Configuration of the repository is relatively easy and readers are advised to refer to the product documentation for their system specific configuration steps.

**Building Java Client** As indicated in the discussion of development lifecycle, generation of Business Object is an important step and can safely be considered to be intuitive to most developers. Yet this section provides an example implementation of a client developed in Eclipse to clarify its role in overall development lifecycle. The program discussed here acts as a client to the Ontology Repository. A java program is displayed in Figure 13-4; that takes the input string from the user, creates a valid SeRQL query expression; connects to the server with required credentials and retrieves the response from the repository.



```

1 package org.daiict.research.sws.ontology.access;
2 import org.openrdf.model.*;
3 import org.openrdf.sesame.*;
4 import org.openrdf.sesame.constants.QueryLanguage;
5 import org.openrdf.sesame.query.*;
6 import org.openrdf.sesame.repository.*;
7
8 public class SesameConnection {
9     public SesameConnection(){
10         super();
11     }
12     public String[] querySesame(String sesameURL, String ontology, String userTerm) throw:
13
14         java.net.URL sesameServerURL = new java.net.URL(sesameURL);
15         SesameService service = Sesame.getService(sesameServerURL);
16         service.login("testuser", "opensesame");
17         SesameRepository myRepository = service.getRepository("rdms-rdf-db");
18         String ontologyP = "<" + ontology;
19         String query = "select SUB from (SUB) rdfs:subClassOf (" + ontologyP + userTerm + ">";
20         QueryResultsTable resultsTable = myRepository.performTableQuery(QueryLanguage.SERQL, (
21         int rowCount = resultsTable.getRowCount();

```

Figure 13-4. Java Client for Connecting Sesame Repository

<sup>3</sup> Sesame Download Page: <http://www.openrdf.org/download.jsp>

Figure 13-5 displays successful realization of two of the required functionalities of the development lifecycle. One is the utilization of the ontology and another requirement of exposing the business object as a Web service. As evident in the Figure 13-5, the Eclipse Package Explorer contains list of java files, each encapsulating discrete business functionality. In the middle pane, the methods of the deployed Web service are visible. The `querySesame` method is based on the java file as displayed in the code snippet above. As evident, both the java program and the Web service accept a user input. The code displays the generation of query to retrieve all the concepts that has `rdfs:subClassOf` relationship with the term provided by the user. As displayed in Figure 13-5, the list of sub class of Mango is listed as a result of query prepared based on the term entered by the user, i.e. "Mango".

**Building Web Service** For quite a long time, building Web services has been a difficult task as it involves many technologies, tools and the know-how. Eclipse WTP<sup>4</sup> project is devoted towards making this process relatively easy and therefore selected for the current experiment. Stable Build 1.0 M8 used for the development can be downloaded from the M8 Page<sup>5</sup>. The page also enumerates the requirements for the installation of this version that should be strictly followed.

Once installation is done properly, the following steps to build the required Web service can be followed. Open newly installed Eclipse and select the J2EE perspective. Create new project by following *File → New → Project → Dynamic Web Project*. Along with other trivial requirements, the new project creation wizard requires to select the *Target Runtime*. Click on new button and provide the local Tomcat installation details. The present experimentation was done using *Apache Tomcat v5.0* and *j2sdk1.4.2\_03*. If done properly, the wizard will result in creation of project directory structure that can be explored in the *Project Explorer*. Locate *Java Source* folder to define the source files that will be used for developing the Web services. Reader can create the java programs in the same workspace or import it from the existing project that was discussed in previous section. The program containing the `querySesame` method created to access the ontology repository can be imported into existing workspace.

To build the Web service, select the java file and press right-click to open *New → Other → Web services → Web service*. In the newly opened wizard, select *Bottom up Java bean Web service* and check *Generate Proxy, Test the Web service, Monitor the Web service* and *Overwrite files without*

<sup>4</sup> Eclipse Web Tools Platform (WTP) project: <http://www.eclipse.org/webtools/>

<sup>5</sup> <http://download.eclipse.org/webtools/downloads/drops/S-1.0M8-200509230840/>

**warning options** and press *Next*. The next object selection page will prompt for the **Bean** to be selected. In next **Service** Deployment Configuration page, appropriate Web service **Runtime, server** and **J2EE** versions can be selected. The next page will display Web services Java Bean Identity with the details of Web service **URI, WSDL Folder, WSDL File**. The available methods, style and usage can also be chosen on the page. Clicking on finish will execute the required tasks and if done successfully, user will prompted to start the **Server**. The Web service Client Test page will display options for testing the generated proxy. In the next page of the wizard the reader can select options for publishing the Web service to Public UDDI Registry. Clicking on finish will result in execution of the Web service. Web service Test client will be opened where user can select from available methods. Clicking on the querySesame method will result in test client same as indicated in Figure 13-5. User will be prompted for Sesame repository URL, Ontology hosted on it and the user term. Clicking **Invoke** button will display the outcome of the method in the **Result** section.

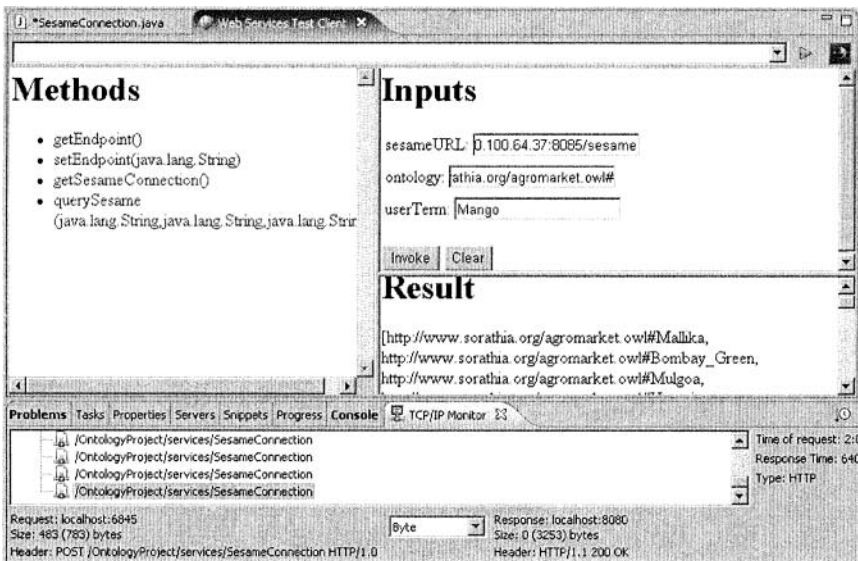


Figure 13-5. Java Client for Connecting Sesame Repository

## **7. SEMANTIC WEB SERVICES**

The discussion just dealt with the steps to expose business objects as Web services. By publishing in public registry these components can be discovered and accessed over the Internet. It is evident from the scale at which Web services based business process integration is being addressed here, that discovery of appropriate services is a critical challenge for a conventional search. Augmentation of Web services with enhanced service description is therefore a prerequisite to any efforts towards automation in discovery, invocation, binding or composition of the developed Web services. It is claimed (Cabral, L., Domingue, J, et al 2004) that there are three different approaches prevailing in the research community to achieve the goal of Semantic Web services. By making sure that services meet the functional requirements, by providing components that fulfills the desired activities or by aggregating vocabulary in service description.

### **7.1 Relevant Standards**

Among important standards OWL-S is derived from DAML-S. It is based on Description Logic. It provides Profile, Process Model and Grounding Ontologies to facilitate Description and Reasoning of the service description files. Web service Modeling Framework (WSMF) is based on a model describing Web service from different aspects. To support scalable communication among the services, this approach recommend emphasizing mediation at syntactic, business logic, message exchange and dynamic invocation levels. As a part of WSMF, the Semantic Web enabled Web services (SWWS) Project is planned to provide framework to support description, discovery and mediation. Another project under WSMF is Web service Modeling Ontology (WSMO), which provides formal service ontology and language. Based on UPML (Unified Problem Solving Method Development) the Internet Reasoning Server (IRS II) is another important framework for Semantic Web services. It consists of Task Models, Problem Solving Methods (PSM), Domain Models and Bridges. While these approaches introduce specific solutions for respective philosophy, the METEOR-S (Patil, Oundhakar et al. 2004) is offered to resolve the issues by leveraging advantage of semantics with the existing standards. It provides complete lifecycle of Semantic Web Processes including development, annotation, discovery, composition and orchestration.



## 7.2 Approach

These research approaches have their own unique merits but to expect a large-scale penetration of any one approach in real life implementation (Cardoso, J., Miller, J., et al. 2004) is too early to predict. Adding semantics to the service description may seem to be an easy alternative. Here, the requirement for the developer is to be able to use the "Concepts" of the domain for which the services are being developed. This is typically realized by selecting and using the relevant ontology. Easy access to the ontology therefore should be integrated with the annotation process. Penetration of concepts of ontology and decent tool support has resulted in development of large ontologies in various domains. Manual annotation using these large ontologies may turn out to be tedious job. Need was felt for a mechanism that enables the user to use and manage specific ontologies for describing the Web services that are being developed. To reduce the manual effort the mechanism can be designed to support automatic or semi-automatic matching process with little user intervention. One such approach (Patil, Oundhakar et al. 2004) was proposed to enable semi-automated annotation of the existing service descriptions with ontology by employing machine learning techniques. In the case of Agricultural Marketing, the ontology covering all the concepts and relations can be utilized uniformly across all the markets to avoid any ambiguity related to the expressed terms. We have adopted the METEOR-S Web service Annotation Framework to annotate WSDL files with known vocabulary. In this approach semi automatic annotation of services is made possible by adopting the schema-matching technique.

## 7.3 Step-by-Step Development

This section explains how a process of Semantic Web service Annotation is carried out. For semantically annotating the existing Web services, the method is explained in detail with the tool named *METEOR-S Web service Annotation Framework* (MWSAF). MWSAF is Eclipse based tool and can be downloaded from LSDIS tool downloads page<sup>6</sup>.

Before starting with the MWSAF, the ontology and all the involved Web services should be assessable to the developer. In this experimentation Agricultural Marketing Ontology will be used for annotation. In current discussion, the business objects developed based on the defined use case and were exposed as Web services. After making provisions of the prerequisites,

<sup>6</sup> MWSAF Download Page: <http://lsdis.cs.uga.edu/projects/meteor-s/mwsaf/>

extract the downloaded package to a convenient place. Open the Eclipse IDE and click on **File** → **Import** → **Existing Project into Workspace**. When prompted for selection of the root directory, click on browse to locate the place where the archive was extracted. If the project is successfully imported in the workspace, the **Package Explorer** will display project files under the root folder named **MWSAF**. In the directory tree, locate the `mwsaf.resource` folder to explore the content. Open `mwsaf.properties` file to edit the entries to suit the installation. Appropriate changes in **MWSAF HOME**, `NegativeDictionary` and other properties should be made. To run the program, locate `MWSAF.java` file in `mwsaf` package from the package explorer, right click and select **Run as - Java Application**. In a user-friendly graphical interface, user is provided many options in **File** and **Tools** menu. To begin with the annotation exercise, click on **File** → **Open** → **Open WSDL From**. Here one can choose to select either File or URL option. In this experiment, the `ReceiveTradeOff-er.wsdl` is selected. As we want to use the Agricultural ontology to annotate the WSDL file, we will now select **File** → **Open Ontology From File** and locate `AgroMarket.owl` as developed earlier. In the left side of the GUI, the WSDL can be explored. Similarly in the right pane, the ontology file can be traversed. Next step is to select **Tools** → **Match Web Service**. MWSAF matches the terms used in the WSDL with the concepts given in the ontology. The middle pane displays the concept mappings and other statistics as displayed in Figure 13-6. Developer can select the acceptable matches out of the offered ones by clicking on the radio button of each offered mapping. These selected mappings can be accepted by clicking on the **Accept Mapping** button. Once the mapping is over, the WSDL file is ready to be annotated. This can be done by clicking on **File** → **Write WSDL**. The result of this option annotated WSDL file `ReceiveTradeOff-erAnnotated.wsdl` will be stored in **AnnotatedWSDL** sub-directory of the root. For detailed account on step-by-step development, the reader is recommended to read the User Guide (LSDIS Lab 2004) available on the tool Web Page.

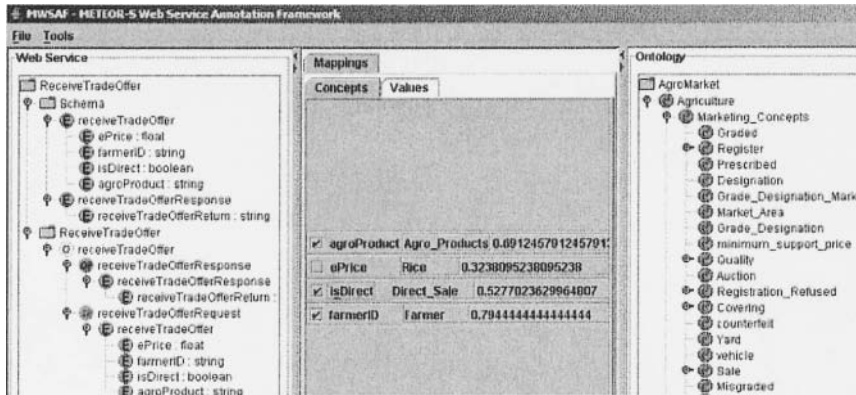


Figure 13-6. Annotating WSDL File

## 8. BUILDING BUSINESS PROCESS MODEL

To employ Web services as Business Services, there are many issues needed to be resolved. Apart from general issues identified earlier in the chapter, specific problems in management of communication, handling of data, handling exceptions and support for transaction among collaborating Web services are very critical.

### 8.1 Relevant Standards

One straight solution for addressing these critical issues is to achieve consensus based standardization in the process. XLANG, BPML, WSFL, BPSS, WSCL, WSCI, BPEL4WS and WS-Choreography enlist the result of a standardization process that prevailed in last five years (XML Cover Pages 2004). These disjoint and parallel standardization efforts have resulted in the problem of heterogeneous and sometimes conflicting specifications (Parastatidis S, Webber J, (2004). Subsequently, the very objective of Web service, i.e. interoperability, is clearly defeated. Composition of Web service is also affected by this problem. Initially, to achieve the composition amongst the Web services, Microsoft introduced a structural construct based XLANG and in parallel IBM brought graph-oriented processes based WSFL. BPEL4WS has emerged as a business process definition standard as a result of consensus among the organizations that initially promoted different standards for the same. In its next version, the popularly known BPEL4WS 1.1 will be renamed as WS-BPEL 2.0. BPEL4WS allows proper management of messages being exchanged between involved in carrying

out a complete business process (Andrews T, et al 2003). BPEL relies on WSDL for its known capability of describing incoming and outgoing message for a given Web service. This helps in designing and implementation of Web services based business process management functionality. Business processes deployed on BPEL engine are Web service Interfaces that can be accessed platform in-dependently. By correlating the messages, BPEL4WS provides mechanism to preserve Web service state. This also helps in long running transactions, which may have several situations where certain completed tasks should be undone due to some erratic condition. To achieve this task BPEL4WS supports structured programming constructs like conditional branching, loop, sequence and flow. BPEL4WS also supports fault handling and compensation mechanism. BPEL4WS employs various constructs like Variables, Partners, Partner links, Flow, Sequences etc, some of them will be elaborated the coming section.

## **8.2 Approach**

For developing model for business process the trading use case described in section 2.1 is utilized. The approach here is to derive a business workflow based on the flow chart depicted in Figure 13-1. The next important step is to select appropriate tool to realize the design and execute the business process. Readers can choose from a vast amount of tools supporting the workflow. Oracle BPEL Process Manager, Biztalk Server, IBM WebSphere Business Integration Server Foundation and Cape Clear Orchestrator provide support with their commercial application packages. Among a few noteworthy open source tools: ActiveBPEL, JBoss jBPM, MidOffice BPEL Editor (MOBE), Bexee BPEL Execution Engine and IBM BPWS4J available as Alphaworks software can be considered.

## **8.3 Step-by-Step Development**

Development and deployment of a business process requires two separate tasks. The first is to create the model of the process from available business logic and the collaborating Web services. The second step is to host the process on a BPEL execution engine. Many commercial and open source tools allow both of these facilities in a single package. Here separate discussion is provided for each step.

**Business Process Modeler.** IBM BPWS4J Editor is used for designing the business process. BPWS4J is successfully tested on JDK version 1.4.1

and Tomcat version 4.1.24<sup>7</sup>. The system used for designing business service therefore must be configured with these prerequisites. The selected editor can be downloaded as `bpws4j-editor-2.1.zip` file from the tool web page<sup>8</sup>. Next step is to extract the downloaded zip file from download directory to Eclipse root directory. This will result in creation of a sub-directory `com.ibm.cs.bpws.tools.bpwsbuilder` under the Plug-ins directory of Eclipse installation. The installation can be verified by starting the Eclipse instance. Open Window→Open Prospective→Others. In Select Prospective window, select BPWS and press OK. A new file can be created by opening File→New→Other. In New window, from the given list of Wizards, click on the **BPWS** folder to select BPWS File and follow the steps.

Now the actual business process modeling begins. The Figure 13-7 displays the building of a new process in the BPWS prospective. To make the decision about what is to be added, now the focus of attention will switch between the BPEL modeler and the problem workflow. In a typical scenario of a business process, a process can span across various existing systems hosting many Web services. Mapping this to our application, a trade process requires interactions of various Web services hosted by APMC and partner organizations. WSDL of respective Web services explains the service invocation and other details.

<sup>7</sup> Tomcat: <http://archive.apache.org/dist/jakarta/tomcat-4/archive/v4.1.24/>

<sup>8</sup> BPWS4J Editor: <http://www.alphaworks.ibm.com/tech/bpws4j/download>

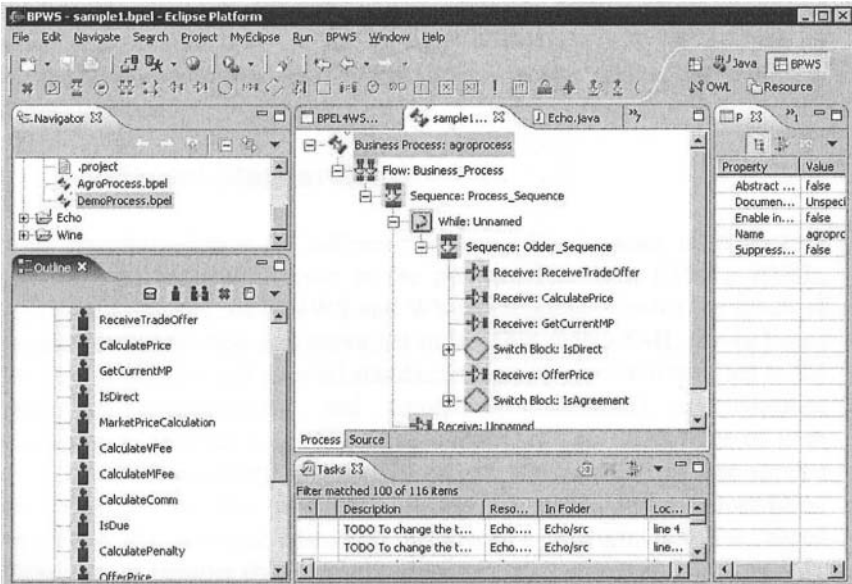


Figure 13-7. BPWS Editor

The discovery of appropriate services here can be enhanced with semantic annotation as described in the previous section. Here, according to the BPEL4WS specification, the message conversation amongst the partners is defined in Partner Link Types. It also defines the role played by individual Web services in the whole transaction. The portType role dictates the allocation of messages to appropriate receivers. For a quick overview on step-by-step development of BPEL4WS document using BPWS4J Editor, reader is recommended the Reference Guide (Mukhi N. 2002) and for detail reference see (Stemkovski V et al 2003). The code snippet contains a part of Marketing.bpel that indicates the Partner links, role and portTypes in our experiment.

```
<partnerLinks>
  <partnerLink name="CalculatePrice"
    xmlns:ns1="http://10.100.64.38:8080/FirstWS/services/CalculatePrice"
    partnerLinkType="ns1:CalculatePricePLT" myRole="CalculatePriceService"/>
  <partnerLink name="IsDirect"
    xmlns:ns2="http://10.100.64.38:8080/FirstWS/services/IsDirect"
    partnerLinkType="ns2:IsDirectPLT"/>
</partnerLinks>
```

Stateful interactions among the Web services in a given business process is achieved by message exchange. Content of these messages include data vital to the application. Variables are the artifacts that hold the data in the

messages. The code snippet contains variables defined in the marketing application.

```
<variables>
  <variable name="agroProduct"
    xmlns:ns15="http://10.100.64.38:8080/FirstWS/services/ReceiveTradeOffer"
    messageType="ns15:agroProduct"/>
  <variable name="price"
    xmlns:ns16="http://10.100.64.38:8080/FirstWS/services/CalculatePrice"
    messageType="ns16:price"/>
  <variable name= .....
```

It can be noted that the declaration of a variable consists of a unique name and message type, which is a XML Schema simpleType. While variables are used to store intermediate state data in messages between partners, it also warrants the need to exchange the data between variables. BPEL4WS specification introduces the notion of *assign* activity to copy data amongst variables. It also supports construction and insertion of new data using expressions. In agricultural trade scenario, calculating price requires this activity to extract market prices and add market fees. The code snippet displays an activity that copies end point references between partner links.

```
<flow name="AgroMarket"> <sequence name="AgroProcessa_Sequence">
<receive name="CalculatePrice" partnerLink="CalculatePrice"
  xmlns:ns20="http://10.100.64.38:8080/FirstWS/services/CalculatePrice"
  portType="ns20:CalculatePrice" operation="calculatePrice" >
</receive>
<reply name="CalculatePrice" partnerLink="CalculatePrice"
  xmlns:ns21="http://10.100.64.38:8080/FirstWS/services/CalculatePrice"
  portType="ns21:CalculatePrice" operation="calculatePrice"
  variable="price">
</reply>
```

Code snippet given above depicts one of the most vital elements in BPEL specification instrumental to achieve concurrency and synchronization amongst the partners. The *Flow* construct enables the grouping of activities. Depending upon the conditions defined, it is possible to execute all or selected activities within the scope of the Flow. This is critical to achieve concurrency in real-life business scenarios. Invoking partner service is one of the most common activities for any business process. Depending upon the business logic, these invocations can be synchronous or asynchronous. Synchronous invocation can possibly result in error condition that returns as WSDL fault. It is necessary therefore to make provisions for efficient fault handling while using the invoke operation.

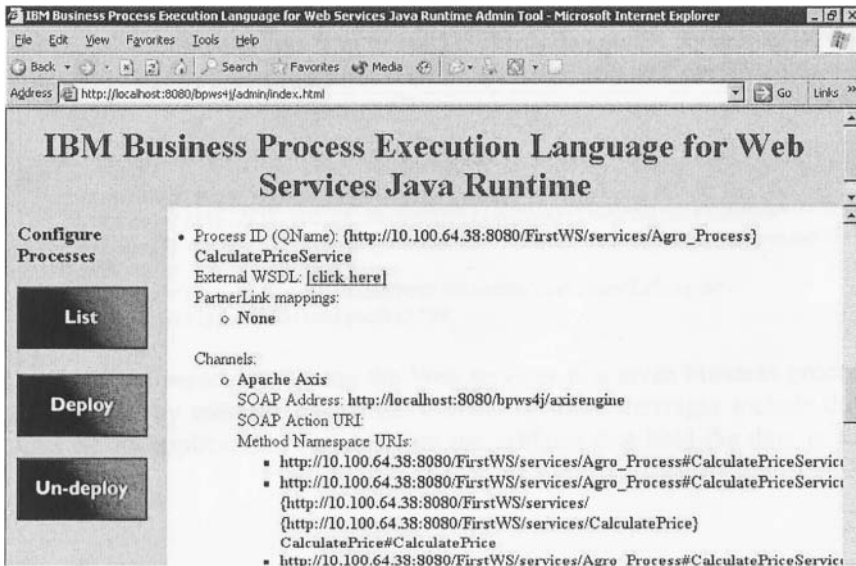


Figure 13-8. Business Process Hosted on BPEL Engine

**BPEL Engine BPWS4J Engine Version 2.1** is selected as the BPEL Engine for this experiment. The tool can be downloaded from BPWS4J page<sup>9</sup> on IBM AlphaWorks Site. To install the Engine simply copy the `bpws4j.war` file into Tomcat's `webapps` directory. The installation can be verified by accessing `http://localhost:port/bpws4j` using any standard web browser. The hostname and port can be replaced according to the case, for example `http://10.100.64.-38:8080/bpws4j`. As depicted in the Figure 13-8, the BPWS4J Engine can be managed by accessing the administrator interface available at `http://localhost:port/bpws4j/admin/index.html`. Three basic operations are allowed namely List, Deploy and Un-deploy the services, can be accessed by clicking on any of respective button in the left pane. To deploy the services, the *Deploy* option can be accessed by clicking on the deploy button. Two inputs are required to deploy the process. The BPEL file generated by the modeler and WSDL file that describes the process are to be deployed. After providing all the required inputs, the process can be deployed by clicking Start Serving the Process button. The deployed service can be accessed by pointing the web browser to `http://localhost:8080/bpws4j`. The page represents the services

<sup>9</sup> BPWS4J Engine <http://www.alphaworks.ibm.com/tech/bpws4j/download>



hosted on the BPEL Engine. Figure 13-8 represents the services hosted on the BPEL Engine.

## **9. CONCLUSION**

This chapter has demonstrated the complete development cycle of semantic Web services based Business Process Orchestration for Agricultural Marketing. In this chapter, challenging real-life business problem, which is considered for implementation at a large scale is explained. Basic complexities involved in the problem are revealed from diverse aspects. The business process integration in such a challenging environment was addressed by the adding semantics into the service description. Various aspects and phases involved to develop semantic business services are discussed with many example implementations of related concepts, tools and techniques to achieve over all goal to demonstrate a full lifecycle of the development process for e-trading of agricultural produce.

## **10. QUESTIONS FOR DISCUSSION**

Beginner:

1. Why simple Web services are not sufficient for all complex real-life applications? Evaluate Semantic approach as one of the solutions of this problem.
2. Are existing UDDI registry systems capable to support publishing and discovery of Semantic Web services?

Intermediate:

1. Explain the ways in which properties can be characterized in OWL. Explain each property type by taking proper examples.
2. What is the purpose of developing Standard Upper Ontology? What role it is expected to play in interoperability?

Advanced:

1. Explore how rule based knowledge representation can be developed on the top of the developed ontology. Find out the major recommendations in this direction including standards, models and architectures

## Practical Exercises:

1. Explore protégé SWRL plug-in to build rules for the developed ontology.
2. Build the rules according to RuleML and provide a comparative note on the capabilities of SWRL and RuleML.

## 11. SUGGESTED ADDITIONAL READING

- Stuckenschmidt, Heiner & Harmelen, Frank Van: *Information Sharing On The Semantic Web*. New York. Springer Verlag, 2005. 3-540-20594-2.: This book tries to provide insight in applying semantics to enable information sharing by providing sound insight of theory, standards, tools and techniques for developing Ontologies to realize the goal of semantic web.
- Gomez-Perez, Asuncion, Fernandez-Lopez, Mariano & Corcho, Oscar: *Ontological Engineering: With Examples From The Areas Of Knowledge Management, E-Commerce And The Semantic Web*. London. Springer, 2004. 1-85233-551-3: This book provides excellent account of practical aspects to develop Ontologies along with discussion on application in different domains.
- Fensel, Dieter: *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, 2nd ed.. (2nd ed.) Berlin. Springer-Verlag, 2004. 3-540-00302-9: Provides effective overview of current state-of-the-art in Ontology related topics.
- Singh Munindar P. & Huhns, Michael N.: *Service-Oriented Computing: Semantics, Processes, Agents*. England. John Wiley and Sons, 2005. 0-470-09148-7: Apart from introduction to Web services and Semantic Web related standards, this book covers a wider prospective including topics on Enterprise Architectures, Service Oriented Computing (SOA), Execution Models, Transactions and Coordination Frameworks. It also includes discussion on advance research topics like multi-agent systems, service selection, security and related issues.
- Zimmermann, Olaf, Tomlinson, Mark & Peuser, Stefan: *Perspectives on Web services: Applying SOAP, WSDL and UDDI to Real-World Projects*. New York. Springer, 2003. 3-540-00914-0: This is an excellent reference book for professionals as well as students planning to concentrate on the emerging areas of Web services and SOA. Case studies given in the book provide professional approach to develop Web services.

## 12. REFERENCES

- Patil, A., S. Oundhakar, et al. (2004) MWSAF - METEOR-S Web Service Annotation Framework. 13th Conference on World Wide Web, New York City, USA.
- Andrews T, et al, (2003), Business Process Execution Language for Web Services, Version 1.1.
- Cabral, L., Domingue, J., et al (2004) Approaches to Semantic Web Services: An Overview and Comparisons. In: First European Semantic Web Symposium (ESWS2004), Heraklion, Crete, Greece (2004)
- Cardoso, J. and Sheth A (2004) "Introduction to Semantic Web Services and Web Process Composition", First International Workshop on Semantic Web Services and Web Processes Composition (SWSWPC 2004), "Semantic Web Process: powering next generation of processes with Semantics and Web services", Revised Selected Papers, LNCS, Springer-Verlag Heidelberg, Vol. 3387, pp.1-13, 2005. ISBN: 3-540-24328-3.
- Cardoso, J., Miller, J., et al. (2004) "Academic and Industrial Research: Do their Approaches Differ in Adding Semantics to Web Services", First International Workshop on Semantic Web Services and Web Processes Composition (SWSWPC 2004), "Semantic Web Process: powering next generation of processes with Semantics and Web services", Revised Selected Papers, LNCS, Springer-Verlag Heidelberg, Vol. 3387, pp.14-21. 2005. ISBN: 3-540-24328-3.
- Chaudhary S, Sorathia V, Laliwala Z, (2004) Architecture of Sensor Based Agricultural Information System for Effective Planning of Farm Activities. IEEE International Conference on Services Computing, 2004 (SCC'04), Conference Proceedings, pp. 93-100
- Horridge M. et al (2004) A Practical Guide To Building OWL Ontologies With The Protégé-OWL Plug-in Edition 1.0. <http://www.co-ode.org/resources/tutorials/-Protégé-OWL-Tutorial.pdf>
- LSDIS Lab (2004) MWSAF User Guide, <http://lsdis.cs.uga.edu/projects/meteors/-mwsaf/downloads/mwsaf-users-guide.pdf>
- Lu L, Zhu G, Chen J, (2004) "An Infrastructure for E- Government Based on Semantic Web Services" SCC'04, Conference Proceedings, pp . 483-486
- Mukhi N.(2002) :Reference guide for creating BPEL4WS documents Quick reference for the BPWS4J editor. <http://www-128.ibm.com/developerworks/web-services/library/ws-bpws4jed/>
- Parastatidis S, Webber J, (2004) Assessing the Risk and Value of Adopting Emerging and Unstable Web Services Specifications, SCC'04, Conference Proceedings, pp. 65-72.
- Piccinelli G, Stammers E, (2002) "From E-Processes to E- Networks: an E-Service oriented Approach", International Conference on Internet Computing,
- Sorathia V, Laliwala Z and Chaudhary S, (2005) Towards Agricultural Marketing Reforms: Web Services Orchestration Approach, '2005 IEEE International Conference on Services Computing (SCC 2005).
- Sreenivasulu V, Nandwana H, (2001) Networking of Agricultural Information Systems And Services in India, INSPEL 35(2001) 4, pp 226-235
- Stemkowski V, Tihankov A,Razumovsky K, (2003) Implementation of the BPEL4WS demo: <http://www-128.ibm.com/developerworks/edu/ws-dw-ws-bpelws-i.html>
- The Stencil Group (2001) Defining Web Services, <http://www.perfectxml.com/X-analysis/TSG/WebServices.asp>
- Ministry of Agriculture, Government of India (2003). The draft model legislation: The State Agricultural Produce Marketing (Development and Regulation) Act.

Thomas S, (2003): Agricultural Commodity Markets in India: Policy Issues for Growth, Technical report, IGIDR, Bombay, India.

XML Cover Pages (2004) Standards for Business Process Modeling, Collaboration, and Choreography <http://xml.coverpages.org/bpm.html>