An Agent-mediated Electronic Market of Semantic Web Services

Vedran Podobnik
University of Zagreb
Faculty of Electrical Engineering and Computing, Department of Telecommunications
Unska 3, HR-10000 Zagreb, CROATIA
vedran.podobnik@fer.hr

Gordan Jezic
University of Zagreb
Faculty of Electrical Engineering and Computing, Department of Telecommunications
Unska 3, HR-10000 Zagreb, CROATIA
gordan.jezic@fer.hr

Krunoslav Trzec
Ericsson Nikola Tesla R&D Center
Krapinska 45, HR-10000 Zagreb, CROATIA
krunoslav.trzec@ericsson.com

ABSTRACT
This paper deals with the application of multi-agent systems in the electronic market (e-market). Proposed is a multi-agent system which enables the complete automation of business processes and should be able to automate the semantic comparison of required and advertised Web services according to user preferences. The system consists of semantic and mobile agents responsible for autonomous service discovery, automated negotiation and automated purchase. Specific concepts and ontologies used for creating semantic information and agent interactions, as well as mechanisms that provide the ability of semantic reasoning, are presented. A case study elaborates upon the developed agent-mediated e-market operations and interactions between agents during the transactions simulated in the business process of Multimedia Service.

Keywords
Software agents, Web services, Semantic Web, OWL, OWL-S, semantic matchmaking, electronic market.

1. INTRODUCTION
Today’s Web architecture is primarily directed towards delivering information visually. In the near future, the architecture of the Web will be geared towards applications that intelligibly coordinate information exchanges. There are three technologies vital for this transition: software agents, Web services and the Semantic Web. We will show that successful integration of these upcoming technologies is possible.

The semantic agents presented in this paper enable the discovery and matchmaking of Web services based on ontologies. These ontologies define the description terminology of the corresponding Web services in an exact manner. By applying the language of the Semantic Web, every Web service can be described with an ontology for the description of Web services OWL-S (Web Ontology Language for Services) that utilizes one or more domain ontologies which define concepts important for a particular domain of interest. Concepts in domain ontologies, as well as the relations between the concepts themselves, are specified with OWL (Web Ontology Language). As a result, the language of the Semantic Web is also utilized for defining the content in agents’ communication. Efficient coordination between agents is achieved by applying adequate agent conversation protocols defined by FIPA (Foundation for Intelligent Physical Agents).

This paper continues the work presented in [13]. The mechanisms for agent-oriented semantic discovery and the matchmaking of Web services presented in this paper are similar to those used in [13], although the matchmaking procedure is no longer based on DAML+OIL (DARPA Agent Markup Language + Ontology Inference Layer) but on OWL. The ideas described in [13] are constrained to single-agent systems and enable automation of the Semantic Web service discovery. This paper extends these ideas by creating a multi-agent system which represents the e-market [5, 18, 20] with the abilities of autonomous semantic service discovery, automated negotiation and automated purchase if the requested goods are deliverable through the Internet (various types of information, multimedia content). The concept of an agent-mediated e-market of Semantic Web services presented in this paper enables the complete automation of business processes [16].

This paper is organized as follows. In Section 2 we give a brief summary of Web services, software agents and the Semantic Web. The idea of an agent-mediated e-market with the ability of automated semantic comparison of required and advertised Web services is presented in Section 3. More details about the specific
2. UTILIZED TECHNOLOGIES

2.1 Web services

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It is identified by an URI (Uniform Resource Identifier) and has an interface described in a machine-processable format. Its interface description can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its interface description, using XML-based messages conveyed by Internet protocols.

WSDL (Web Service Description Language) is a language that provides a communication level description for Web services [2]. A WSDL document is basically an XML document specifying the location, operations and methods of a Web service, and instructions on how to access it. WSDL files are XML files with no semantics [14]. WSDL is the industry standard for Web service description.

A UDDI (Universal Description, Discovery and Integration) repository is a business registry. After businesses have registered their services with UDDI, the registry uses standard industry taxonomies, or classification schemes, to categorize businesses, services and service types (so called Yellow pages). Typically, when a business registers a Web service, in addition to the usual information, the business also stores a WSDL description of the service or a reference to the corresponding WSDL document. This specification then enables a user to easily connect to the Web service. For UDDI there are alternatives, e.g. SDEC (Services and Data Exchange Catalogue) or WSIL (Web Services Inspection Language). Figure 1 presents the classical Web service architecture.

2.2 Software agents

A software agent is a program which acts on behalf of its owner or user while conducting complex information and communication actions over the Web. From the user’s point of view, agents improve efficiency by reducing the time that is required for executing personal or business tasks. The goal of the technology of software agents is also to reduce the owner’s efforts while transacting business.

There are numerous possible applications of software agents in the real world. However, these applications are limited by the fact that agents do not have the ability to reason in a semantic manner. Therefore, they cannot properly substitute a human in executing certain tasks. A new idea which helps solve this problem is the technology of the Semantic Web. Figure 2 presents the relations between the main features of intelligent software agents [1, 6, 7, 8].

2.3 The Semantic Web

Today’s Web is a remarkable source of information. This information is mostly presented using HTML (Hypertext Markup Language). HTML is designed to allow Web developers to display information in such a way that is accessible to humans for viewing via Web browsers. While HTML helps us visualize information on the Web, it does not describe the information in such a way that would enable software programs to find or interpret it. Enriching the Web with a semantic dimension represents a key element in the transformation from information to knowledge. The Web can reach its full potential only if it becomes a place where data can be shared and processed by automated tools in the same way as by people. The idea is to have programs that are able to share and process data, even when these programs have been designed totally independently.

3. AGENT-MEDIATED SEMANTIC WEB SERVICES

With the proliferation of Web services, it is becoming increasingly difficult to find a Web service which will satisfy the desired requirements. There are various approaches to solving this problem. One of them, presented in [19], proposes an OWL-S/ UDDI Matchmaker which takes advantage of UDDI’s outspread in the Web service technology infrastructure and OWL-S’s explicit capability representation. In order to achieve the desired symbiosis, OWL-S profile descriptions are stored inside an UDDI registry which is enhanced with an OWL-S matchmaker module that can process the OWL-S descriptions. Our approach to solving the problem of the discovery of eligible Web services uses a similar idea, but the practical implementation is slightly different. Namely, we use software agent technology which enables the complete automation of business processes.
With the Web service mechanisms currently available, users need to find an appropriate service manually and determine whether a particular Web service provides the functionality they require. This is mostly done by browsing a registry, such as UDDI, or by directly obtaining a Web service description from a business partner. Since businesses try to minimize the amount of time and effort put into maintaining and running their applications, we were motivated to extend the classical architecture of Web services with Semantic Web and software agent technologies. As a result, we implemented an agent-mediated e-market with the ability of automated semantic comparison of required and advertised Web services according to user preferences.

As a result, we implemented an agent-mediated e-market with the ability of automated semantic comparison of required and advertised Web services according to user preferences.

The concept of the Semantic Web enables businesses to discover Web resources by content and not just using a plain keyword search, which is commonly done nowadays. As a result, Web services which are accompanied by their corresponding semantic information become meaningful to computer programs. Such information allows computer programs to autonomously decide whether a particular Web service satisfies certain requirements or not.

Automation of the discovery process is supported by software agents due to their ability to perform autonomous task execution adjusted by their owner’s profile and the current state of the surrounding environment. Importing the technology of software agents into the classical Web service model also enables businesses automated negotiation and automated purchase. Although this paper concentrates on describing the ability of autonomous semantic service discovery, automated negotiation and automated purchase are also important issues in process of business automation. This is because, in combination with automated discovery, they form a solution for the automation of complete business transactions. Automated purchase is not a complex problem because it is supported by the basic concepts of software agent technology which enable mobile agents to migrate throughout a network and transport data between network nodes. Thus, if the requested goods are deliverable through the Internet, a mobile agent has the ability to deliver these goods directly to its owner. Automated negotiation is a much more complex issue since it presumes that a software agent has mechanisms implemented (very often sophisticated) for ensuring the best possible deal for its owner. Therefore, an agent with the ability to negotiate must be grounded on mechanisms of artificial intelligence (AI).

Figure 3 presents our model of an agent-mediated e-market. We introduced three types of agents into classical architecture of Web services:

1. **Human Requester**: The user who initiates the service request.
2. **Requester Agent**: An agent that represents the user's preferences and requests services on their behalf.
3. **Provider Agent**: An agent that represents the service provider and offers services.
4. **Registry Agent**: An agent that manages the registry of services.

With the introduction of software agents, the Web service discovery process becomes more efficient and automated. The diagram illustrates the steps involved in the discovery, binding, and invocation of services, as well as the role of agents in this process.
services (Requester Agent, Provider Agent and Registry Agent). Requester Agent acts on behalf of its human owner in the processes of Web service discovery, negotiation and purchase. Provider Agent is representative of Web service provider. Registry Agent enables Provider Agents to publish their Web service descriptions and enables Requester Agents to find eligible Web services. By importing these three types of agents we extend the classical architecture of Web services with the capabilities of autonomous service discovery, automated negotiation and automated purchase. There is no need for Web service providers to change their services or Web service providing entities, but simply to upgrade their architecture with an agent (the Provider Agent in Figure 3). This agent’s role is to serve as a gateway between the software agent environment and the Web services environment. A Web service architecture modeled in such a manner has the capability to simultaneously function in “classical” and “agent” mode.

Our model of an agent-mediated e-market was implemented as a multi-agent system using the JADE (Java Agent Development Framework) agent platform [4]. Agents communicated by exchanging ACL (Agent Communication Language) messages. Efficient coordination between agents was achieved by applying adequate interaction protocols. Two types of pre-defined FIPA conversation protocols – FIPA Request and FIPA Contract-Net [15, 17] were used.

4. CREATING SEMANTIC INFORMATION
In the context of the Semantic Web, an ontology refers to a description of concepts and relationships between these concepts in our area of interest. Therefore, an ontology is the terminology used for a given domain of interest. Pre-defined ontologies allow software agents to interpret the meaning of Web resources. Ontologies can refer to other ontologies, and thus create domain-dependent terminologies which describe certain concepts and relationships in more detail. Ontologies are explicit semantic models which include taxonomies of terms and semantic relations which help interpret queries and reason with knowledge [2].

"People can’t share knowledge if they don’t speak a common language" [9]. This statement can be transferred from “people” to “machines”. The exchange of knowledge is possible only if the communication participants speak a common language, i.e. they are able to map a sign to an object in the same way. This means that both evoke the same concept when using the sign (see Figure 4). Ontologies should enable machines to achieve such a common understanding.

OWL (http://www.w3.org/TR/owl-features) [2] was developed by the W3C and was intended to be used for cases where the information contained in documents needed to be processed by applications, as opposed to situations where the content only needed to be presented to humans. It is a revision of the DAML+OIL [12] ontology language, incorporating lessons learned from the design and application of DAML+OIL. OWL extends RDF (Resource Description Framework) and RDFS (Resource Description Framework Schema) statements to provide a rich logic language based on Description Logics (DL) [3]. DL is a formalism that can be used for knowledge representation and reasoning. It facilitates finding implicit consequences of explicitly represented knowledge.

Figure 4. Multiple interpretations of sign "Jaguar"

Figure 5. The structure of OWL-S ontology

OWL-S (http://www.w3.org/Submission/OWL-S) [2] is an OWL based technology for describing the properties and capabilities of Web services in an unambiguous, computer interpretable markup language. Existing Web service description mechanisms, such as WSDL, provide a low-level communication layer for Web services. As a result, existing mechanisms define only how to access a Web service, while OWL-S defines why to use a certain Web service. The execution of four basic tasks is enabled by the OWL-S architecture. Namely, the OWL-S language enables users and software agents to automatically discover, invoke, compose and monitor Web resources offering services under specified constraints. The three main parts of an OWL-S ontology are (see
Figure 5: a service profile for advertising and discovering services; a service model, which gives a detailed description of a service's operation; and a service grounding, which provides details on how to interoperate with a service via messages. The existing grounding enables us to align the semantic specification with the implementation details described using WSDL.

Figure 6 depicts the Semantic Web languages. HTTP (Hypertext Transfer Protocol) protocol supported by URI mechanisms provides assistance for sharing ontologies all over the Web. XML supported by Unicode standard provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents. XML Schema is a language for restricting the structure of XML documents and also extends XML with datatypes. RDF is a datamodel for objects ("resources") and relations between them. It provides a simple semantics and can be represented in an XML syntax. RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for generalization-hierarchies of such properties and classes. OWL adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, richer typing of properties, characteristics of properties (e.g. symmetry), and enumerated classes. OWL-S is built upon OWL and provides support for description of Web services in semantic manner.

Figure 6. The Semantic Web languages

OWL isn’t sufficient in providing software agents with semantic abilities. Software agents must implement adequate knowledge-based mechanisms which can be used for knowledge extraction from OWL documents and for understanding the corresponding semantic information. A Matching Algorithm is a mechanism which assists in selecting an eligible Web service for given user preferences. In this work, we use an existing matching algorithm from [13, 21] based on the available semantic information encoded in OWL-S. More specifically, it matches (compares) advertised service parameters with the requested needs, i.e. parameters. This match obtains results with some degree of similarity, i.e. the comparison is assigned a rank. Such a ranking will eventually become relevant since it is highly unlikely that there will always be a Web service available which offers the exact functionality requested. Users (or software agents that act on behalf of users) can make a decision based on these rankings on whether they want to use of a certain Web service which doesn’t exactly match the desired functionality. The Matching Algorithm compares both all input and all output parameters of the advertised service with all input and all output parameters of the requested service. Moreover, it also considers parameters classification, and allow customization through plug-ins [21]. All these parameters are defined through a semantic description in OWL-S document, using a service profile. Matching procedure is logically divided into four stages, each independent on other three. The final result (MATCH or FAIL) will be based on the results of each matching stage ("logical AND"). These stages are: input matching, output matching, profile matching and user-defined matching. Implemented Matching Agent uses two components to realize semantic matchmaking process of requested and advertised Web services. These components are (see Figure 7) [11]:

- OWL Inference Engine
- OWL-S Matchmaker

Figure 7. Software components required for implementing a Matching Agent

These components are necessary to accomplish complex reasoning tasks, including Java understandable interpretation of requesters’ and provider’s service written in OWL-S. The OWL Inference Engine component is used to transform OWL files in the form appropriate for the OWL-S Matchmaker component that,
applying the Matching Algorithm, semantically compares transformed OWL files and calculates degree of similarity between Web services. The OWL Inference Engine represents off-the-shelf DL reasoner for ontologies written in OWL, named OWLJessKB, which uses Jena API to parse OWL-S descriptions, fetched somewhere from the Web, into SVO (subject-verb-object) triples. OWLJessKB also uses Jess (Java Expert System Shell) [10], rule-based engine and scripting environment, which can be used for creation of agent KB (Knowledge Base) populated with facts and rules. Jess uses Rete algorithm (pattern matching mechanism) to process facts and deduce new information according to rules. It is used to enable the Matching Agent to process SVO triples, represented as Jess facts, according to Jess rules that represent OWL axioms. Consequently, the Matching Agents can understand semantics of OWL-S descriptions using the OWLJessKB component and use the OWL-S Matchmaker component to semantically compare requested and advertised.

5. CASE STUDY: MULTIMEDIA SERVICE

5.1 Description of domains of interest
To demonstrate how a developed agent-mediated e-market operates, we simulated the business process of multimedia content subscriptions. This means that Web service providers offer subscriptions to a desired multimedia content scope with a certain guaranteed level of QoS (Quality of Service). For the purpose of conducting this simulation, three OWL ontologies (describing Multimedia Service, Service Providers and Scope) were created. Figures 8, 9 and 10 show the OWL ontologies describing the Multimedia Service, the Service Providers and the Scope domains, respectively. Comprehension of these three domains of interest is necessary to understand the demonstration of the semantic matchmaking of Web services in the created e-market. After describing all three domains of interest with appropriate OWL ontologies, an OWL-S description of the desired Web service can be made [13, 15].

5.2 Description of Web services
In this section, we describe some important attributes of real descriptions of Web services in order to illustrate the presentation of the ideas introduced in this paper. There are four IOPEs (input, output, precondition and effect) characterizing Web services which are included in the case study. They are as follows:

- **one precondition**

  A precondition for using a certain advertised Web service is that the potential user is a subscriber of the corresponding Web service provider. Service providers are described by the Service Providers domain ontology shown in Figure 9.

- **one input**

  Each potential user of a Web service providing multimedia content subscriptions wants to receive only that content which contains the information that he/she is interested in. Therefore, potential users define their desired multimedia content scope as input. Available scopes are described by the Scope domain ontology shown in Figure 10.

- **one output**

  The output defines the type of multimedia service used to send the content provided by the advertised Web service to the user. Available types of multimedia service are described by the Multimedia Service domain ontology shown in Figure 8.

---

4 Proposed agent-mediated e-market was implemented during the fifth Summer Camp organized by Ericsson Nikola Tesla in cooperation with the Faculty of Electrical Engineering and Computing, University of Zagreb [17].
The effect of using some of the advertised Web services is level of QoS guaranteed to the user. Available levels of QoS are described by the Multimedia Service domain ontology shown in Figure 8.

For the purpose of testing the developed e-market, we created OWL-S descriptions of three available Web services offering the multimedia content subscriptions. We will refer to these services as Provider#1 Web service, Provider#2 Web service and Provider#3 Web service. All three descriptions are similar, but differ in their IOPE values. Their corresponding IOPEs are shown in Table 1.

<table>
<thead>
<tr>
<th>Table 1. The advertised Web services’ IOPEs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precond-</td>
</tr>
<tr>
<td>condition</td>
</tr>
<tr>
<td>Provider#1</td>
</tr>
<tr>
<td>Provider#2</td>
</tr>
<tr>
<td>Provider#3</td>
</tr>
</tbody>
</table>

We also created OWL-S descriptions of three Web services requested by users interested in utilizing the service of multimedia content subscription. We will refer to the descriptions of the requested Web services as Requester#1 Web service, Requester#2 Web service and Requester#3 Web service. Their corresponding IOPEs are shown in Table 2. The structure of these OWL-S descriptions is similar to the OWL-S descriptions of the offered Web services. However, they differ in their IOPE values.

<table>
<thead>
<tr>
<th>Table 2. The requested Web services’ IOPEs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precond-</td>
</tr>
<tr>
<td>condition</td>
</tr>
<tr>
<td>Requester#1</td>
</tr>
<tr>
<td>Requester#2</td>
</tr>
<tr>
<td>Requester#3</td>
</tr>
</tbody>
</table>

In the Table 3, the matching relationships between the advertised and requested Web services are shown. Relationships between entities in the ontologies that describe our domains of interest (the ontologies shown in Figures 8, 9 and 10) decide whether the requested and advertised Web services match or not [21].

<table>
<thead>
<tr>
<th>Table 3. Matching relationships between advertised and requested Web services</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Provider#1</td>
</tr>
<tr>
<td>Provider#2</td>
</tr>
<tr>
<td>Provider#3</td>
</tr>
</tbody>
</table>

5.3 Operations on the implemented e-market

5.3.1 Types of agents

We will now demonstrate how the developed agent-mediated e-market operates. Figure 11 illustrates the e-market supported by two semantic registries of available Web services. Each registry contains two specialized agents – a Discovery Agent (DA) and a Matching Agent (MA). The agents representing the Web service providers and Web service requesters in the created market are referred to as the Provider Agent (PA) and the Requester Agent (RA), respectively. A brief description of the types of agents in the modeled e-market follows:

- **The Discovery Agent (DA)**

The Discovery Agent is a representative of one semantic registry for PAs and RAs. When the DA is contacted by another agent, it must first determine whether the other agent is a PA or an RA (if it is neither of them, the DA ignores the contact message). If the other agent is a PA, then the DA negotiates the possibility of advertising that agent’s Web service in its semantic registry. If the other agent is an RA, then the DA provides that agent with a list of all the PAs whose advertised Web services semantically match the Web service requested by the RA. During all this actions, the DA is interacting with an MA, either providing the MA with certain information or requesting certain information from it.

- **The Matching Agent (MA)**

The Matching Agent facilitates semantic matching of Web services which is inherent in every semantic registry. The MA is only obliged to store data regarding advertised Web services in its corresponding registry and for the semantic matching of these advertised Web services with a Web service requested by an RA. The MA simply responds to the requests made by the DA, and is therefore never in direct contact with a PA or an RA. To conclude: the MA fulfills the functionality of a semantically enhanced e-market while the DA makes this functionality available to PAs and RAs.

- **The Provider Agent (PA)**

The Provider Agent represents an agent which offers a certain Web service. Initially, the PA wishes to advertise its Web service in one or more semantic registries. After successfully advertising the service, the PA waits to be contacted by an RA that is interested in the Web service it is providing.

- **The Requester Agent (RA)**

The Requester Agent requires a certain Web service. First, the RA contacts the DA requesting information regarding the PAs which represent advertised Web services which semantically match the RA’s requirements. Upon acquiring the desired information, the RA contacts all the proposed PAs and makes a deal with the most appropriate one (e.g. with the PA that offers the lowest price for providing the requested Web service).
5.3.2 Agents’ interactions

We now present descriptions of agents’ interactions in the created e-market:

1: FIPA CONTRACT-NET (PA (Initiator) and DA (Responder))

A PA wishes to advertise its Web service. It uses the FIPA Contract-Net interaction protocol because this protocol enables it to send \textit{cfps} (Call for Proposal) to multiple DAs. After the PA receives proposals from all the DAs, it chooses the one which offers the lowest price for advertising its Web service. Interaction 1.1 is a part of this conversation.

1.1: FIPA REQUEST (DA (Initiator) and MA (Responder))

After a DA receives a message of acceptance for its proposal (from the PA), it requests its MA to update the database of advertised Web services adding this new one.

2: FIPA CONTRACT-NET (RA (Initiator) and DA (Responder))

An RA wishes to find the Web service most appropriate to its needs. After an RA receives proposals from all the DAs to which it sent \textit{cfps}, it tries to contact all the proposed PAs to find the one which offers the lowest price for utilizing the requested Web service. At the end of this conversation, the RA sends messages to all the DAs with information of its satisfaction regarding the proposed PAs. Interactions 2.1, 2.2 and 2.3 are all parts of this conversation.

2.1: FIPA REQUEST (DA (Initiator) and MA (Responder))

After a DA receives a call for its proposal (from the RA) containing information regarding the requested Web service, it requests from its MA to semantically match all the advertised Web services from the MA’s database with the requested Web service. The MA returns a list to DA of all the PAs that advertise Web services which successfully passed the MA’s semantic matchmaking. It is important to note that the process of matchmaking does not result in an agreement between the Web service requester and the Web service provider – this is the purpose of the negotiation process which follows the matchmaking procedure (interaction 2.2) [22].

2.2: FIPA CONTRACT-NET (RA (Initiator) and PA (Responder))

After a RA receives proposals from all the DAs, it tries to contact all the proposed PAs. After the RA receives proposals from all the PAs, it chooses the one which offers the lowest price for utilizing the requested Web service.
2.3 FIPA REQUEST (DA (Initiator) and MA (Responder))

After a DA receives a message containing information regarding the RA’s level of satisfaction with the proposed PAs, it sends a request to its MA. In that request, it asks the MA to update the ratings of the advertised Web services in its database.

After the RA resolves which Web service is most appropriate, it makes a deal with the corresponding PA. The RA then provides its owner with some basic information regarding the chosen Web service. The RA attains that information from the OWL-S ontology describing it. The information is presented to the RA’s owner through a pop-up window. As can be seen in Figure 12, the RA’s owner can choose between two types of information for previewing.

First, the owner is offered the possibility to see the Web service provider’s contact information. This option can be very useful if the RA’s owner has any questions regarding the rules of utilizing the chosen Web service. Recall that the RA’s owner did not participate in the process of searching for the most appropriate Web service. The owner only instructed his RA about his/her requirements after which the RA autonomously found the corresponding Web service. Therefore, it is likely that the RA’s owner will need some extra information regarding the chosen Web service. The other type of information available to the RA’s owner for previewing is a brief description of the chosen Web service.

Figure 12. The Web service provider’s contact information

Figure 13 shows the communication which took place between the described agents during the simulation of one transaction and presents how the modeled e-market operates. Figure 13 is actually a screenshot of the agents’ communication represented through the JADE’s Sniffer Agent. Although the developed e-market does not have a constraint on the number of agents participating, to enable straightforward presentation we included only two PAs, two semantic registries (with one DA and one MA each) and one RA.

The initial interactions are executed by the PAs while negotiating the publishing of their Web services in the semantic registries. Each PA advertises its Web service only in the registry which offers the lowest price for publishing the service. After that, the RA tries to find an eligible Web service. First, the RA contacts the DA for proposals of appropriate Web services. Then, after the MA performs matchmaking and the RA receives proposals of eligible Web services, the RA negotiates the terms of using the Web services directly with PAs. In this example, both PAs offered appropriate Web services so the RA decided to use the service which was offered at a lower price. Finally, the RA sends information regarding its level of satisfaction with the proposed PAs to DAs and then the MAs update the ratings of the advertised Web services in its databases. For a more detailed description of the agents’ communication presented in Figure 13, refer to Figure 11 and the associated explanations.
6. CONCLUSION
There is no doubt as to the direction of the development of the Web. The architecture of the Web will become geared towards applications which intelligibly coordinate information exchanges. This paper tries to demonstrate a fragment of the possibilities that such a Web of the future will provide. Namely, it presents an integration of three technologies: software agents, the Semantic Web and Web services. The significance of this integration is shown through a case study which simulates transactions in an agent-mediated e-market with the capability to perform an automated semantic comparison of required and advertised Web services according to user preferences. A comprehensive description of, not only the syntax, but also the semantic meaning of the service is used to find the most suitable match. The idea of the Semantic Web, incorporated through OWL-S ontologies, enables software agents to perform the autonomous discovery of eligible Web services. We extend the classical architecture of Web services with the following options: autonomous semantic service discovery, automated negotiation and automated purchase if the requested goods are deliverable through the Internet. The result is a system which enables the complete automation of business processes.

From the user’s point of view, the integration of the above mentioned technologies provides the user with a software assistant which can replace him/her in executing numerous tasks. Currently, users need to perform most of these tasks themselves due to the restriction of computer machines to the syntactic level of reasoning.

From the point of view of the IT (Information Technology) industry, using these technologies in the processes of product and service creation gives them a possibility for significant improvements. Utilizing these technologies provides IT companies with the opportunity to use product and service recycling much more frequently. This is possible since software agent technology enables the creation of generic agents whose behaviour is determined by the knowledge they have. That knowledge is completely separated from the agents and it is incorporated in the Semantic Web.

This paper is the foundation for further research of numerous possible applications of the above mentioned technologies in the real world. Application of these technologies in the global economics of the future is doubtless. Therefore, further research in this field is very promising.

7. REFERENCES