Agents for e-Business Applications

A.Negri, A. Poggi, M. Tomaiuolo, P. Turci
Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Parma
Viale delle Scienze, 181A – 43100 – Parma
Tel. +39 0521 905708, Fax +39 0521 905723
{negri, poggi, tomamic, turci}@ce.unipr.it

ABSTRACT

Web services are increasingly utilized by organizations that want to improve responsiveness and efficiency. While they may be used in an isolated way, the need of integrating them as part of workflow processes is more and more felt. However the creation of applications composed of dynamically selected basic services entails facing two essential issues: how to efficiently discover Web services and how to allow and facilitate their composition.

In this paper, we propose an agent-based framework representing an attempt of giving an answer to such problems. Its peculiar characteristic and strength is the integration of the agent technology with other key emerging technologies, that is semantic Web, Web service, rule engine and workflow technologies. The multiagent system, which constitutes the backbone of the framework, represents the "glue" that holds these pieces together and makes them perform properly. The framework has been experimented and evaluated in the realization of a simple, but realistic, prototype of an e-travelling system. The results, though still preliminary, are quite encouraging.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *multiagent systems*.

General Terms

Management, Experimentation, Security.

Keywords

Agent-mediated e-business, ontology, Web services, workflow, security, service discovery and composition.

1. INTRODUCTION

The rapid growth of the Internet, networking systems and Web technologies is stimulating an ever-increasing number of companies to use internet not only to provide e-commerce worldwide, but also to improve internal communication, to help manage supply chains, to conduct technical and market research, and to locate potential partners.

Agent technology is considered one of the most interesting technologies to successfully support these activities, which are gathered under the name of e-business. In fact, besides being an ideal mechanism for implementing complex systems, agent technology is well-suited to applications, like e-business ones, that are communication-centric, based on distributed computational and information systems, and requiring autonomous components readily adaptable to changes. In particular, the use of agent technology for realizing e-business systems provides conceptual simplicity, enhances scalability, and makes interactions in a large collection of information sources become tractable.

Considering their peculiar features and the role they should play in an e-business scenario, it is evident that agents do not necessarily offer new functionalities. To be successful, it is crucial to appropriately engineer and integrate agent technology with other technologies that have found and will find a purpose within enterprise computing. Data-mining, workflows, rule engines, just to mention a few, belong to the former group.

New interesting and promising technologies are represented by the semantic Web and Web services. The vision which is making its way into the research community is to encapsulate the organization's functionalities within appropriate interfaces and advertise them as one or more Web services, which could be integrated, when brought into play, in workflows. This innovative idea brings with it new outstanding opportunities but also new great issues, related mainly to the ability of automatically discovering and composing Web services. An answer to these problems could come from the semantic Web technology.

Recently, we have seen an explosion of interest in ontologies as artefacts to represent human knowledge and as a critical component in several applications; among these the e-business applications. Moreover the "marriage" between agents and ontologies seems to be the kind of technology that can significantly change the face of enterprise software.

On the one hand, ontologies should accomplish the task of giving a precious support to solve two tricky problems: how to efficiently discover Web services and how to make possible the interoperability of heterogeneous Web services. In order to facilitate the resolution of such a structural and semantic heterogeneity, Web services, which play the role of workflow components, will have their interfaces semantically described by ontological concepts.

On the other hand, ontologies enable agents to communicate in a semantic way, exchanging messages which convey information according to explicit domain ontologies.

In this scenario agents represent the "glue" that could hold these pieces together and make them perform properly.

Assuming to adopt an agent-based approach, a typical scenario of an e-business application would be characterized mainly by three actors: service providers, brokers and users, playing roles which would be allocated to different concrete agents. The system architecture would likely be organized in communities constituted by different kinds of agents: service providers, personal assistants and middle agents (e.g. service brokers, user profile managers, workflow managers, etc). In order to achieve their goals (semantic matching, service contracting and so on) these autonomous agents should be able to perform their tasks in cooperation or competition with other agents and to interoperate with external entities (e.g., legacy software systems). Moreover they should show reasoning capabilities and should have a support for dynamic behaviour modification based on business rules. Finally they should be able to build workflows, compose the external Web services and monitor their execution. The entire process should be supported by a distributed trust management.

Clearly the researchers are well aware that such a scenario is quite ambitious and the outlined objectives difficult to achieve in a short period. Indeed there are several overlooked technical issues and the present technology presents significant limitations. Nevertheless the realizations of prototype systems centred on the underlying infrastructure can be of great help in order to raise awareness of these issues and to delineate possible solutions.

Bearing in mind what said above, the aim of the present paper is to introduce a framework, under development at the University of Parma, for the realization of agent-based e-business applications.

In the next section we discuss the related work in the fields of the emergent and more established technologies which we aim at integrating with agent technology. Section 4 describes the framework aiming at being the basis for the realization of successful and innovative agent-based e-business applications. In this section we focus mainly on its architecture, the ontological support, the integration with a rule engine and our proposal for a distributed trust management. Section 5 shortly presents a prototype of an e-travelling system realized by using JADE and the aforementioned framework. Finally, we give some concluding remarks and present our future research directions.

2. RELATED WORK

There is evidence from several research studies [2],[33] that agents represent one of the most suitable technologies which can be used to meet the performance needs for e-business applications. Indeed, agents provide both an appropriate level of abstraction in modelling e-business applications and a natural merging of object orientation and knowledge-based technologies that can facilitate the incorporation of reasoning, learning and high-level dialogue capabilities to realize intelligent and adaptive applications. In particular the current interest in using agents for developing e-business applications is rising mostly because different works have shown how agent technology can be leveraged if used together with technologies exploited in the Internet, that is, semantic Web, Web services and workflows [8],[10],[17],[21],[28],[32],[33].

Semantic Web technologies appear to be the right means to provide the semantic integration between data and processes across systems that can be owned by different enterprises [9]. This technology is not completely mature yet; some major activities related to the definition of languages for expressing the semantics of the Web are still in progress [22],[11]. Nevertheless different works have shown how the powerful synergism between agents and semantic Web could be very promising [28],[40] and some efforts have been made in order to define ontology models and develop tools suitable for agents aiming at being truly semantic aware agents. The research community contributions have been mainly devoted to cope with three different issues:

- The formal definition of a standard language for expressing semantics on the web which has led to the Web Ontology Language,
- The development of integral software infrastructures, for writing semantic web applications, offering a variety of tools to engineer ontologies. These supports for the construction of ontology-oriented and ontology-based applications are mainly thought for applications written by using the Java language,
- The development of ontological supports specifically thought for multiagent systems.

The second and third points are strictly connected to the first one since OWL is considered the reference language; therefore the work carried out, starting from OWL, has developed tools more suitable for different contexts.

As far as the second point is concerned, an interesting approach is characterized by the definition of a meta-model that closely reflects the OWL syntax and semantics. This is the case of the modelling APIs of Jena [25], which is the most famous and widely used tool in the sphere of the semantic web (and recently also in the context of multiagent systems), and the OWL API [4] framework. The latter consists of a high-level programmatic interface for accessing and manipulating OWL ontologies. To date, it is shipped with a reference implementation which is main memory-based and not optimized.

Considering the third point, the focus is on the specific needs of multiagent systems, and the objective is to provide a communication support enabling agent to perform the proper semantic checks on a given content expression. A significant example of the efforts made in this direction is represented by the ontological support of JADE [24], designed to represent, using Java objects, a taxonomy of concepts. Such semantically aware agents should then be able to discover, invoke, compose and monitor those Web resources that provide services. In order to make agents able to use a service, they need a computer interpretable description of the service itself and furthermore to know the means by which it is accessible. To that purpose, a community of researchers is developing an ontology of services, called OWL-S, with the aim of providing a semantic orientation to the description of Web services.

To enable software systems for e-business applications, security issues have to be carefully analysed and sound solutions have to be deployed. A number of different solutions for the problems of authentication and authorization in open systems have been proposed in the scientific literature, and some standards have emerged through the years. Most of them are based on some kind of PKI and signed certificates issued by a Certification Authority.

In particular, this is the case of X.509, which is the best known and adopted standard for authentication and authorization. However, its weaknesses have been clearly demonstrated in a number of works [20], above all related to its (mainly politically motivated) effort to create a global directory of unique names. In any case, relying on an external entity as root of all certifications represents an additional, not directly controllable, point of failure for the whole system.

In contrast, different approaches have been proposed, based on local names. Both SDSI [1] and PetName Markup Language [29] allow local names to be used in a global scale by prefixing them with the public key of the principal defining them, in the form of `(key, name)` couples. This way, name conflicts are solved thanks to the uniqueness of the public keys. In [44], authors show that local names and YURLs are more robust than global names to phishing attacks, arguing the root for these attacks lie in the global namespace itself. Moreover, in [30],[31], authors shows that local names and a subset of the SDSI/SPKI standard [14] can be used to implement a distributed RBAC infrastructure, in which local names are interpreted as distributed roles, whose name is localized to their defining principal (key). In [14], delegation certificates are defined as s-expressions, to link separate namespaces and to delegate access rights among principals.

Local names and delegation certificates are the key to build systems adhering *trust management principles* [27]. These systems are completely distributed as they avoid any centralized authority. This way they can easily scale to large peer-to-peer networks, where each node is in charge of protecting its own resources and to show proper credentials when accessing resources of other nodes.

To conclude just a few remarks on JADE since it is considered the reference implementation of the FIPA specifications and one of the most used and promising agent development framework. The present release of JADE tries to provide agent developers with a support integrating almost all these technologies, even if in our opinion only partially. As a matter of fact, JADE agents can exploit an ontological model of the application domain to improve their interactions, are able to interact with external Web services [17] and finally different works have shown how the integration of a JADE agent with the Jess rule engine is feasible. But this simply represents a first step towards an effective integration with the aim of supporting real e-business applications. In fact a detailed analysis shows its weaknesses.

The idea which mostly inspired the design of the JADE content language and ontological support was to define an ontology independent abstract model of the content language that could be subsequently bound to any domain ontology representation expressed using an object-oriented data model. This ontological support has been conceived when the Semantic Web was on its very early stage of research and development and OWL was not already established as a standard. Consequently its expressive power is clearly limited with respect to OWL and basically allows expressing taxonomy of concepts, predicate and actions and therefore it is not able to represent completely the different application domains where JADE agent may be used.

JADE provides the integration with the JESS rule engine [26], which is probably the most known Java rule engine implementing the Rete algorithm [16]. This integration is realized through a so-called JessBehaviour that allows the encapsulation of a JESS rule engine inside a JADE agent and has the duty of storing and retrieving information in/from the rule engine. The main limits of this solution are: i) the rule engine is completely hidden to the other agents of the system and there is not any support for the cooperation among different rule-based agents (i.e., agents encapsulating a JESS rule engine) and ii) JESS is a commercial software and so we have additional costs if we plan to realize commercial applications by using JADE together with JESS.

At the beginning of the last year a new release of JADE was announced providing a service allowing the integration of agents with Web services. This service, called WSIGS (Web Services Integration Gateway Service), is a stand alone, encapsulated application that provides transparent, bidirectional transformations between JADE agent services and Web services [17]. To date, the WSIG supports only simple WSDL description of Web services, without taking into account emerging technologies related to the semantic Web, like OWL-S specifications. Another limit of this integration service is that it does not provide any means that agents can use to automatically compose Web services.

Finally, JADE also provides some basic facilities, to sign and/or encrypt ACL messages. But, to have a complete and usable security infrastructure, these facilities must be augmented with tools enabling the distributed management of trust relationships.

3. EXTENDING JADE FOR E-BUSINESS APPLICATIONS

To overcome the limits of the present release of JADE, we have realized GAIN (Grid Agent INfrastructure), an agent based framework that allows the automatic composition of Grid tasks and Web services through the use of workflow technologies [32]. Its architecture is based on a society of agents, mostly composed of two kinds of agents: component managers and workflow managers.

Each component manager is associated to one or more Web services and is responsible for the interaction with them. Through the use of the WSIG JADE add-on, the component managers are able to invoke a Web service, converting ACL messages into WSDL descriptions and vice versa. Moreover, a component manager allows a flexible provision of services defining "on the fly" the features of the services (prize, timing, etc.) through a set of business rules managed by a rule engine and modifiable by the operators of the service provider through a Web interface.

Workflow managers have the goal of supporting users in the process of building the workflows, composing external Web services and monitoring their execution. To accomplish this complex activity the workflow managers provide the users with two alternative automatic procedures:

i) Predefined workflow; the workflow is extracted from a repository of standard and common templates, e.g. templates used in previous computations. In this case the duty of the workflow manager is to support the user in the selection of the most appropriate Web services for the execution of the different workflow tasks. The workflow manager is able to select a matching service thanks to the exploitation of a shared ontology that gives a common knowledge background to all the agents in the system.

ii) Dynamic workflow; the workflow manager, according to the user's requirements, creates a new workflow, composing the atomic services available in the system. This is done by applying a planner (we have realized extending the SGP planner [39]) that works on the operators extracted from the OWL-S descriptions of the Web services, provided by component managers. After the composition of the final workflow, the workflow manager is able to update it and possibly replace those Web services that are failed or no more available or cannot satisfy the execution time constraints.

Moreover, GAIN offers to the users the possibility of manually building workflows. In this case, a personal assistant (i.e. an agent, associated with each user active in the system, responsible for the interaction between the user and the other parts of the system) helps its user presenting her/him the tasks (Web services) that can be composed and possibly informing her/him when the realized workflow does not satisfy the composition rules, coming from the related OWL-S descriptions. When a complete workflow is realized, the user can ask its personal assistant to delegate the workflow execution to a workflow manager. The enactment is clearly a problematic phase. When a workflow is going to be executed, a Web service could be no more available due to the expiration of a timeout, a failure of a resource or other unpredictable problems. In this case the workflow manager helps the user finding a new solution, creating a new contract phase with all the component managers that are able to satisfy the task and suggesting to the user the replacement of the failed service with the new one.

So far we have given a concise description of the system architecture and the responsibilities of the major system components, intentionally leaving out the treatment of the issues connected to the tools needed by the agents, in order to carried out their activities. In the following subsections, we will go into details, illustrating our proposals and the implemented tools.

3.1 Ontological Support

As highlighted several times within this paper, the multi-agent systems "powered" with an ontological support seem to be the key to the success of the e-business applications. JADE already makes available an ontological support but as mentioned above it is quite limited. In order to provide a JADE agent with an adequate expressive power (i.e., equivalent to the one offered by OWL DL), it is necessary either to replace or to integrate the JADE ontological support. In the attempt of finding a suitable solution to this problem one has to choose among the proposals described above and others, each characterized by different domain knowledge modelling techniques and answering different needs. The majority of the research work in this field is thought for the semantic Web. But while in the vision of the semantic Web the increasing interest in ontologies is driven by the large volumes of information available and by the need of automating many information retrieval activities, in the agent context the focal point is slightly different and it is mainly on communicative acts - communications which implies actions. Agents would use ontologies to perform the proper semantic checks on a given content expression, and therefore ontologies should include concepts (objects of the domain of the discourse) but also predicates (assertions on properties of concepts) and actions (that agents can perform in the domain). Moreover a peculiar characteristic of the agent community is the heterogeneity of resources available and the roles played by different agents of a system. This leads us to claim that a one-level approach, based on a single ontology model and the corresponding tool, with the aim of being omni comprehensive seems to be seldom feasible. In our opinion a good compromise is represented by choosing different approaches in different contexts. Our solution was to realize a compound tool, called OWLBeans [8], that allows the use of ontologies described by using OWL DL [3]. These ontologies can be used by agents for performing their tasks in cooperation with other agents, for interoperating with external entities (e.g., legacy software systems) and for performing a semantic matching of Web services, described by using OWL-S and having inputs and outputs associated with concepts belonging to a domain ontology.

OWLBeans is based on a two-level approach with the aim of coping with both the issues of managing complex ontologies and of providing ontology management support to lightweight agents, which seldom need to deal with the whole complexity of a OWL DL ontology. Therefore, lightweight agents maintain the simple JADE ontology support whereas one or more dedicated agents, acting as ontology servers, are able to use and manage complete OWL DL ontologies and provide the service to the agents that need it.

The main functionality of OWLBeans is to extract JADE ontologies from OWL DL ontologies realizing a set of ontologies usable by JADE agents, with the obvious shortcoming that not all the information maintained in the original OWL ontologies are taken into account. Therefore, for all those systems that need a complete support for OWL DL ontologies, OWLBeans offers a set of ontology server agents implemented as JADE agents, providing a common knowledge base and reasoning facilities. These ontology servers use the Jena toolkit to load, maintain and reasoning about OWL ontologies. The other agents of the system do not need to know anything about the Jena toolkit given that these ontology servers provide them with a set of simple actions for querying and manipulating the ontologies. Furthermore, ontology servers take into account proper authorization mechanisms. In particular, the underlying trust management support (discussed in the following subsection) has been leveraged to implement a certificate-based access control. Only authenticated and authorized principals will be granted access to managed ontologies. A delegation mechanism allows the creation of communities of trusted entities, which can share a common ontology, centrally managed by the ontology server.

Finally, despite the fact that the JADE ontological support is quite simple, it could still be complex for some devices with limited resources such as smart phones. This is the reason why we have decided to improve OWLBeans adding a further feature which allows agents to import taxonomies and classifications from OWL ontologies, in the form of a hierarchy of Java classes with the purpose of providing very simple artefacts to access structured information. Given its modular architecture, based on an intermediate ontology model, OWLBeans also provides further functionalities, e.g., saving a JADE ontology into an OWL file, or generating a package of JavaBeans from the description provided by a JADE ontology.

3.2 Production Rule Management

As for the ontological support, in order to cope with the limits of the current JADE support for rule engines, we realized a software library, called D4J (Drools4JADE) [6], that integrates JADE agents with the Drools rule engine [13]. Drools is a well known, freeware implementation of the so-called Rete-OO algorithm. Apart of its open-source availability, one of the main advantages of Drools is exactly the fact that it is not just a literal implementation of the Rete algorithm, but rather an adaptation for the object-oriented world. This greatly eases the burden of integrating the rule engine and the application rules with the existing external objects. In Drools, asserted facts are simple Java objects, that can be modified through their public methods and properties. Where Jess requires hundreds of lines of code, for example to simply access an ACL message mapped into a Java object, Drools rules can obtain the same result in a dozen of easy-reading code lines.

D4J guarantees both the advantages of full rule-based agents, i.e., agent whose behaviour and/or knowledge is expressed by means of rules [23],[38], and the advantages of rule-enhanced agents, i.e., agents whose behaviour is not normally expressed by means of rules, but that use a rule engine as additional component to perform specific reasoning, learning or knowledge acquisition tasks [19]. In facts, in D4J, the Drools rule-engine is integrated into an agent as a JADE behaviour, but it also provides an API for interacting with it through ACL messages allowing both remote storing and retrieval of knowledge and the cooperation among different rule-based agents. Moreover, this API allows rules mobility, i.e., a rule-based agent can move a rule to another rule-based agent.

Given their nature, business rules often refer to domain specific concepts and, especially when dealing with data on the semantic Web, these concepts are part of a domain ontology. To better support this scenario, rule-enhanced JADE agents should be augmented with a tool for the automatic transformation of concepts, relations and individuals of an OWL ontology [5] to java classes, properties, and instances. The D4J framework can be integrated with OWLBeans, which enables the extraction of JavaBeans from an OWL ontology. The JavaBeans can then be directly asserted as facts into the working memory of Drools.

3.3 Distributed Trust Management

Out of the box e-business applications are not certainly possible if security problems are not analyzed and addressed. Our framework supports the implementation and deployment of secure systems, adhering trust management principles. For this purpose, local names, interpreted as distributed roles, and delegation certificates are made available, to build peer-to-peer networks of trusted entities.

The accurate release of authorizations is often the most critical point of security systems. Ideally, systems should respect the principle of *least privilege*, but this often contrasts with other requirements, as easiness of understanding, scalability and manageability. In this respect, the RBAC model [37] has proven to be a good abstraction to manage large and complex systems, up to corporate and virtual-organizations environments.

Following the RBAC model, each resource manager of our system (i.e. each node in the peer-to-peer network) has to deal with three main concepts: principals (i.e. authenticable entities which act as users of resources and services), permissions (i.e. rights to access resources or use services) and roles [12].

A many to many relationship binds principals and the roles they are assigned to. In the same way, a many to many relationship binds permissions and the roles they are granted to, thus creating a level of indirection between a principal and his access rights. This also leads to a better separation of duties (between the assignment of principals to roles and the definition of role permissions), to implement privilege inheritance schemes among superior and subordinate roles and to permit temporary delegations of some of the assigned roles towards other principals. The fundamental principle here is that each node is in charge of defining its own roles, and of assigning principals to them.

Dynamic delegation of access rights is made possible through the use of delegation certificates, whose structure is based on the theory of [14]. But we have avoided s-expressions, preferring XML to them, as it provides a better ground to exploit and integrate standard technologies. In particular, in recent times SAML [36] have emerged as a language to express properties of authenticable principals, encoded in the generic form of signed security assertions. SAML assertions can easily bind public keys to local names, and certify the links between two different local namespaces, as it happens in SDSI/SPKI certificates.

Both providers and consumers of services exploit this infrastructure to build a distributed P2P network of trust relationships, embodied and quantified in terms of signed delegation certificates. These certificates, attached to signed request messages, list the set of roles the requester is assigned to, and they are signed by trusted authorities according to customizable policies. The delegation certificates owned by the agents can also be used to delegate roles, and thus access rights, to other agents, to allow them to complete the requested tasks or to achieve delegated goals [7].

Delegation is particularly important to deal with the activation of intermediate agents, acting between the human user and the concrete service providers, i.e. personal agents, workflow managers and agents providing composite services. In this case, privileges must be forwarded in the form of delegation certificates from the user toward each agent in the chain, up to the final service provider, which will check them for consistency with local security policies. These policies are stored and managed locally as XACML documents [43].

It is worth noting that similar solutions are made possible in GRID environments by issuing *proxy certificates* [41]. These are essentially locally signed X.509 certificates, allowing not only to specify the identity of a subject, but also to grant only a limited set of privileges to it. But, being an extension to the X.509 standard, they end up with neither being fully compliant with existing infrastructures, nor avoiding the fundamental problem of centralized authorities. Besides, they do not even enjoy the advantages of XML-based solutions, as human readability, extensibility and interoperability.

To cope with the security problems coming from the remote utilization of the rule engine and from the mobility of rules, also D4J exploits this security layer to implement: i) the authentication of the agents, ii) the checking of permissions owned by the agents, and iii) the delegation from an agent to another agents of its permissions. Therefore, only authenticated and authorized agents can successfully ask another to store and retrieve knowledge or to accept new rules. In D4J, the infrastructure is used to enforce security policies at two different levels: proper authorization is necessary to modify the working memory and the rule set of an agent; moreover, each rule is associated with a specific protection domain, limiting the resources made accessible when it is scheduled for execution.

4. AN E-TRAVELLING APPLICATION

The framework has been experimented in the realization of a prototype of an e-travelling system which provides both atomic services (e.g., the purchase of a flight ticket) and composite services (e.g., the reservation of various hotels and restaurants and the purchase of flight tickets for a business or holiday trip). The behaviour of this system can be described introducing three different kinds of actors: service providers, service brokers and users.

Service providers provide atomic services and make them available through a set of Web services.

Service brokers help other agents to find services and to compose them. All brokers share an ontological model to describe the provided services. Service providers that like to register to one of these brokers need to provide the mapping of their Web services to such an ontological model. Travel operators have also the opportunity of directly delegating the complete management of their services to one or more service brokers. In this case, travel operators and service brokers need to share the service status (i.e., the free/busy places of a flight, the free/busy rooms of a hotel, etc.) and travel operators need to provide their service brokers with the business rules in order to make them able to compute the prices of the services. Business rules are not static, but need to be dynamically updated by the travel operators. For this reason, service brokers provide a Web interface that allows a travel operator to list its business rules, add and remove new rules and identify dependencies among them.

A user interacts with a service broker through a Web interface too. She/he can either ask for an atomic service, receiving from the broker an ordered list of the feasible solutions that should satisfy her/his requirements, or a composite service, receiving, for example, information concerning the bookings and tickets needed for a trip or the different prices of the services necessary to organize a conference. In particular, in the case of a composed service, the Web interface assists her/him in defining the composed service through a graphical construction of a workflow in which each step corresponds to a component of the service. This is an iterative process that, for instance, initially allows a user to fix the ticket for flights and the booking for hotels, and then, after all the terms have been settled (dates, places, etc.), to add booking for special restaurants, purchasing of tickets for theatres plays, etc.

Moreover, service brokers maintain a profile of registered users and use such profiles to send them information about special offers that can be of their interest.

The core of the realized system is a multi-agent system that is deployed on the servers of the service brokers. Service providers have only the duty of maintaining the servers managing their services (the service management can also be delegated to the service brokers). Users need simply a Web browser and an Internet connection.

The multi-agent system is based on five types of agents: service providers, workflow managers, personal assistants, user profile managers and service brokers. Figure 1 shows the interactions between the different types of agents and between the agents and the software systems used by the providers to manage the services they offer.

Service providers are agents that combine the behaviour of a GAIN component manager agent powered by the D4J framework. These agents have the following responsibilities: i) to provide an interface (compliant to the ontology shared by service brokers) for the Web services offered by the providers' management systems, ii) to advertise the offered services at the service brokers (a service provider can offer services through different service brokers), and iii) to manage the business rules used to compute the features of booking and selling contracts.

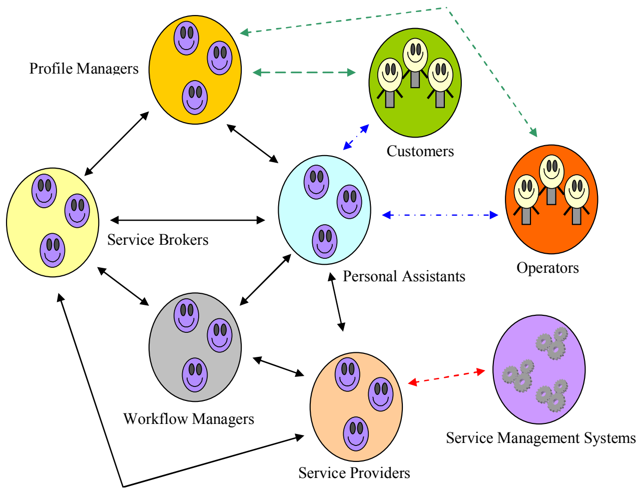

Figure 1 – Interactions between the different types of agents and between agents and service management systems.

Workflow managers are instances of the GAIN workflow manager agent and so their responsibilities concern mostly the management of workflows, strictly related to the composition of services. In particular, they are responsible for: i) checking the workflow submitted by the users, ii) building and/or completing the workflows that satisfy the requests of the users, iii) starting and monitoring the execution of the workflows, and possibly iv) updating the workflow structure if a component fails during its execution. They receive an explicit delegation from the user's personal assistant, which is responsible for providing them with all privileges needed to accomplish the task. These privileges can be further delegated to service providers, in order to enable access to crucial resources.

Personal assistants are agents that are in charge of helping service provider operators and customers of the system in their actions. In particular, they help service provider operators in the maintenance of business rules and customers in both the searching and execution of atomic services and in the construction and execution of composite services. Moreover, they have the duty of building and updating the profile of the users of the system. Users interact with the personal assistants through Web servers. On the one hand, these Web servers provide users with the appropriate Web interfaces. On the other hand, they allow the personal assistants to push information to the users.

Personal assistants are respectively created and destroyed at the login and logoff of the users. Therefore, personal assistants need to delegate the maintenance of the user profiles to persistent agents, called profile managers. These agents are mainly responsible for exchanging the user profiles with the personal assistants when they are created and destroyed. Moreover, they have the duty of exploiting user profiles to look for new and advantageous offers (e.g., airplane tickets with reduced prize) that may be of interest and possibly to send users information about them through emails. Personal assistants are also responsible for the management of user's credentials, in the form of identity certificates submitted directly by the user at login, and other delegation certificates received by other peers in the course of business interactions.

Service brokers are the agents managing information about the services offered by the connected service providers. In fact, on the one hand, service providers register their services by the service brokers sending them an OWL-S description of their services. On the other hand, both personal assistants and workflow managers can send queries to the service brokers to look for services satisfying user requirements.

Up to now, we have not experimented the system with real users and real services, instead we have tested and evaluated the system functionalities implementing some "artificial" services and involving a group of students, acting either as service provider operators or as customers. Some of the information used by the service providers, implemented just for the experimentation of the system, comes from the Web site of some real service providers (e.g., flight companies, hotel brokers, etc.). With the view to doing it, we have realized an application able to get information from a Web site and then to make available this information through a Web service. This application is independent from the structure of the Web site because it builds queries and extracts information from the responses on the basis of an XML file containing the patterns of both queries and answers. Therefore, we are able to simulate a set of different service providers by simply defining the patterns for querying the Web site of a service.

5. CONCLUSION

In this paper, we have presented an agent-based framework for realizing e-business systems that integrates agent technology with other technologies that have found, and will find, a purpose within enterprise computing: web services, workflows, ontologies and rule engines.

We are well aware that the current multi-agent solutions, aiming at facilitating and supporting the realization of e-business applications, need to be improved since the technologies used are still not completely mature. However a lot of researchers and software developers are really interested in giving a significant contribution in this direction, driven by the motivation of providing a strengthening of the related standards and new methodologies, algorithms and implementations to realize real flexible, adaptive intelligent e-business systems [2],[15].

Our future activities will be oriented towards the aforementioned goal. In particular, we will continue working on the JADE software environment in order to both improve the integration of the JADE agents with the most interesting knowledge and internet-oriented technologies and realize real adaptive agents that will be the basis of next and future e-business applications. At present, we are working in two main directions: i) to provide a full OWL DL support through a home-made framework supplying ontology management and reasoning functionalities, with the main purpose of reducing the amount of computational resources and time required (with respect to the Jena engine), ii) to enhance the agent-based and graphical support, that users can currently use when they define their workflows or modify the set of business rules controlling their services.

6. REFERENCES

[1] Abadi, M. On SDSI's Linked Local Name Spaces. Journal of Computer Security 6(1-2):3-22.1998.

[2] AgentLink III. Agent Technology Roadmap. Available from http://www.swsi.org/resources/wsmf-paper.pdf.

[3] Akkermans, H. Intelligent E-Business - From Technology to Value. IEEE Intelligent Systems, 16(4):8-10, 2001.

[4] Bechhofer, R. Volz, and P. Lord. Cooking the semantic web with the OWL API. In Proc. Int Semantic Web Conference, pp. 659-675, Sanibel Island, FL, 2003.

[5] Bechhofer, S., van Harmelen, F., Hendler, J. Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., & Stein, L.A. OWL Web Ontology Language Reference. 2004. Available from http://www.w3.org/TR/owl-ref/.

[6] Beneventi, A., Poggi, A., Tomaiuolo, M., & Turci, P. Integrating Rule and Agent-Based Programming to Realize Complex Systems. WSEAS Trans. on Information Science and Applications, 1(1):422-427, 2004.

[7] Bergenti, F., Rimassa, G., Somacher, M., Botelho, L.M. A FIPA Compliant Goal Delegation Protocol. Communication in Multiagent Systems, Vol. 2650, pp. 223-238. 2003. Springer.

[8] Bergenti, F., Poggi, A., Tomaiuolo, M., Turci, P. An Ontology Support for Semantic Aware Agents. In Proc. Seventh International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2005 @ AAMAS), Utrecht, The Netherlands, 2005.

[9] Berners-Lee, T., Hendler, J., Lassila O. The Semantic Web - A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. 284(5):34-43, 2001.

[10] Buhler P.A., Vidal, J.M. Towards Adaptive Workflow Enactment Using Multiagent Systems. Information Technology and Management, 6(1):61-87, 2005.

[11] de Bruijn, J., Polleres, A., Lara, R., Fensel, D. OWL DL vs. OWL Flight: Conceptual Modeling and Reasoning for the Semantic Web. In Proc. of the 14th Int. World Wide Web Conference (WWW2005), pp. 623-632, Chiba, Japan, 2005.

[12] Chadwick, D.W., Otenko, A. RBAC Policies in XML for X.509 Based Privilege Management. In Proc. of the 17th Int. Conf. on Information Security (SEC2002), pp. 39-54, 2002.

[13] Drools software and documentation. Available from http://drools.org.

[14] Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Ylonen, T. SPKI Certificate Theory. RFC 2693, 1999.

[15] Fensel, D., Bussler, C. The Web Service Modeling Framework WSMF. Electronic Commerce Research and Applications 1(2): 113-137, 2002.

[16] Forgy, C. Rete: A Fast Algorithm for the Many Patterns/Many Objects Match Problem. Artificial Intelligence, 19(1):17-37, 1982.

[17] Greenwood, D., Callisti, M. Engineering Web Service-Agent Integration. In IEEE Conference of Systems, Man and Cybernetics, 2004. Available from http://www.whitestein.com/resources/papers/ieeesmc04.pdf.

[18] Gibbins, N., Harris, S., Shadbolt, N. Agent-based semantic web services. In Proc of the 12th International World Wide Web Conference (WWW2003), Budapest, Hungary, 2003.

[19] Gutknecht, O., Ferber, J., Michel, F. Integrating tools and infrastructures for generic multi-agent systems. In Proc. of the 5th International Conference on Autonomous Agents. Montreal, Canada, 2001.

[20] Gutmann, P. (2000). X.509 Style Guide. Available from http://www.cs.auckland.ac.nz/~pgut001/pubs/x509guide.txt

[21] Hendler, J: Agents and the semantic web, IEEE Intelligent Systems, 16(2): 30-37, 2001.

[22] Horrocks, I. and Patel-Schneider, P. F. A proposal for an OWL rules language. In Proc. of the Thirteenth International World Wide Web Conference (WWW 2004), pp. 723-731, 2004.

[23] Hindriks, K.V., de Boer, F.S., van der Hoek, & W., Meyer, J.C. Control Structures of Rule-Based Agent Languages. In Lecture Notes In Computer Science, Proceedings of the 5th International Workshop on Intelligent Agents V, Agent Theories, Architectures, and Languages, Vol 1555, pp. 381-396, 1998, London, UK: Springer-Verlag.

[24] JADE software and documentation. Available from http://jade.tilab.com.

[25] Jena software and documentation. Available from http://jena.sourceforge.net.

[26] JESS software and documentation. Available from http://herzberg.ca.sandia.gov/jess.

[27] Khare, R., Rifkin, A. Weaving a Web of trust, World Wide Web Journal Special Issue on Security, 2(3):77–112, 1997.

[28] Labrou, Y. Agents and ontologies for e-business. Knowledge Engineering Review, 17(1):81-85, 2002.

[29] Lambda for Humans – The PetName Markup Language. Available from http://www.erights.org/elib/capability/pnml.html

[30] Li, N. Local Names in SPKI/SDSI. In Proc of the 13th IEEE Computer Security Foundations Workshop. 2000.

[31] Li, N., Mitchell, J.M.. RT: A Role-based Trust-management Framework. In Proc of the Third DARPA Information Survivability Conference and Exposition (DISCEX III), pp. 201-212, 2003. Washington, D.C.

[32] Negri, A., Poggi, A., Tomaiuolo, M., & Turci, P. Dynamic Grid Tasks Composition and Distribution through Agents. Concurrency and Computation: Practice and Experience, 2005.

[33] Papazoglou, M.P.. Agent-oriented technology in support of e-business. Communication of ACM, 44(4):71-77, 2001.

[34] Papazoglou, M.P. The World of e-Business: Web-Services, Workflows, and Business Transactions In Lecture Notes In Computer Science, CAiSE '02/ WES '02: Revised Papers from the International Workshop on Web Services, E-Business, and the Semantic Web, Vol 2512, pp. 153-173, 2002. London, UK. Springer-Verlag

[35] Poggi, A., Rimassa, G., Tomaiuolo, M. Multi-user and security support for multi-agent systems. In Proc. of WOA 2001, pp. 13-18, 2001. Modena, Italy: Pitagora.

[36] SAML - Security Assertion Markup Language. Available from http://xml.coverpages.org/saml.html.

[37] Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E. Role Based Access Control Models. IEEE Computer 29(2):38-43, 1996.

[38] Schroeder, M., & Wagner, G. Vivid agents: Theory, architecture, and applications. Applied Artificial Intelligence, 14(7):645-676, 2000.

[39] Sensory Graph Planner software and documentation. Available from http://www.cs.washington.edu/ai/sgp.html.

[40] Silva, N., Rocha, J., Cardoso, J. E-Business Interoperability Through Ontology Semantic Mapping. In Proc. of the Processes and Foundations for Virtual Organizations, pp. 315-322, 2003. Lugano, Switzerland.

[41] Tuecke, S. Welch, V. Engert, D. Pearlman L. Thompson, M. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, 2004. Available from ftp://ftp.rfc-editor.org/in-notes/rfc3820.txt.

[42] Weikum G. Special Issue on Infrastructure for Advanced E-services, IEEE Data Engineering, 24(1), 2001.

[43] XACML - Extensible Access Control Markup Language. Available from http://xml.coverpages.org/xacml.html.

[44] YURL - Decentralized Identification. Available from http://www.waterken.com/dev/YURL.