INTEGRATING AGENT TECHNOLOGIES INTO ENTERPRISE SYSTEMS USING WEB SERVICES

Eduardo H. Ramírez
Tecnológico de Monterrey, Campus Monterrey
Eugenio Garza Sada 2501, Col. Tecnológico, Monterrey, N.L. México

Ramón F. Brena
Tecnológico de Monterrey, Campus Monterrey
Eugenio Garza Sada 2501, Col. Tecnológico, Monterrey, N.L.México

Keywords: Software agents, Web Services.

Abstract: In this work we present a decoupled architectural approach that allows Software Agents to interoperate with enterprise systems using Web Services. The solution leverages existing technologies and standards in order to reduce the time-to-market and increase the adoption of agent-based applications. Insights on applications that may be enhanced by the model are presented.

1 INTRODUCTION

Software Agents (Jennings and Wooldridge, 1996) and Web Services(W3C, 2003) have become key research areas for a growing number of organizations and they are expected to bring a new generation of complex distributed software systems(Jennings, 2000). But even if Agent technology is little by little finding its way into the mainstream, Web Services have been adopted much more widely and rapidly(Barry, 2003).

Several authors have pointed out some overlapping between Agents and Web Services semantic capabilities (Hunhs, 2002)(Preece and Decker, 2002), issues regarding how they may be competing or complementary technologies remain open(Petrie, 1996). Because of that, research involving Agents and Web Services is mainly focused on building improved semantics(Dickinson and Wooldridge, 2003)(Hendler, 2001) communication languages and interaction protocols(Labrou et al., 1999).

We assume that in order to impact real-world organizations, a greater emphasis should be made on interoperability between agent-based applications and enterprise information systems. Moreover, we believe that the adoption of agent technologies will grow by leveraging existing industry standards and technologies. Therefore the problem we address is an instance of "the legacy software integration problem"(Nwana and Ndumu, 1999),(Genesereth and Ketchpel, 1994).

In this work we present a decoupled architectural approach and design principles, called "Embedded Web Services Architecture" (ESWA), that allows agent-based applications to be integrated into enterprise application environments(Peng et al., 1998) using Web Services, thus allowing them to interoperate with robust conventional systems such as:

- Web-applications and Portals.
- Enterprise Resource Planning (ERP)
- Manufacturing Execution Systems (MES)
- Workflow engines

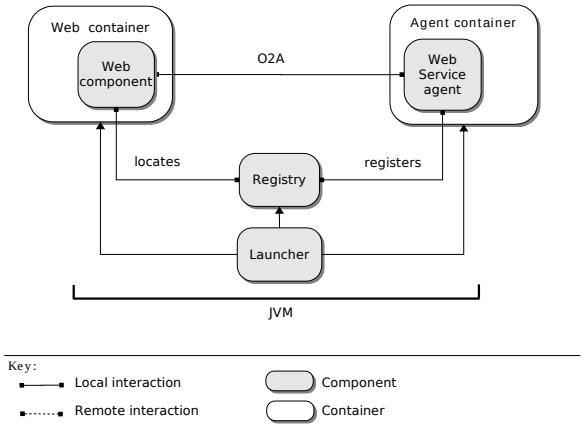
Also, we discuss about the kind of agent-based applications we have found to be suitable for this approach and the nature of Web Services that agents can provide.

2 SOLUTION OVERVIEW

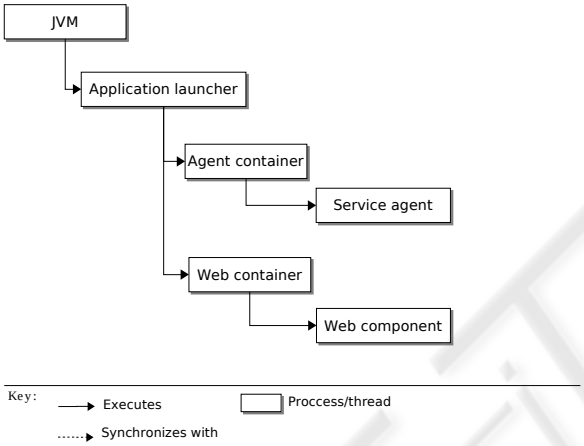
2.1 Architecture

Figure 1: Decoupled architecture top-level view

H. Ramírez E. and F. Brena R. (2005).
INTEGRATING AGENT TECHNOLOGIES INTO ENTERPRISE SYSTEMS USING WEB SERVICES.
In *Proceedings of the Seventh International Conference on Enterprise Information Systems*, pages 11-15
DOI: 10.5220/0002552800110015



(a) Component view



(b) Execution model

Figure 2: EWSA decoupled architecture

As shown in figure 1, the underliying metaphore used to determine the design strategy was the aim to create a "black-box" in which agents can live and perform complex tasks. The main architectural principle consists of decoupling agent-based applications through the exposure of Web Service interfaces. Enterprise applications should not be aware that a service is provided by agents if the system offers a standard SOAP endpoint as interface. Appearing to the world as a conventional Web service or application.

An agent based application which exposes Web interfaces requires the interoperability of Web components and agents and their respective containers, as they are built on different programming models each following different sets of specifications. The relevant components and containers are:

Web container Also called the "Servlet container," is the application that provides the execution environment for the web components and implements the Java Servlet API in conformity with the JSR-154(Sun Microsystems, Inc., 2003) specification. Web containers are usually built within web servers and provide network services related with HTTP request processing.

Web component Servlets are the Java standard user-defined web components. JSR-154 defines them as "A Java technology-based web component, managed by a container that generates dynamic content"(Sun Microsystems, Inc., 2003). They follow a synchronous processing model as they are designed to handle the content of the HTTP requests. The dynamic content delivered in the request may be HTML for web pages or XML(W3C, 2000) for Web Services.

Agent container The execution environment for the agents provided by the agent platform in conformity with FIPA(FIPA, 2002) specifications.

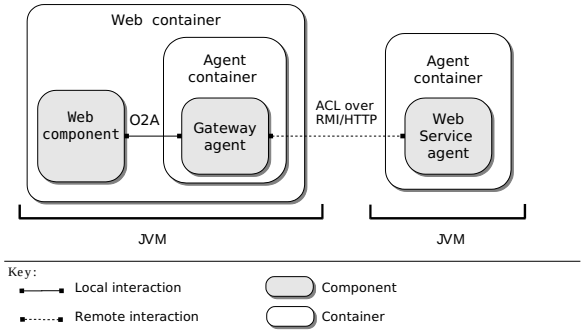
Web Service agent A Java thread that periodically executes a set of behaviours containing the agent tasks. For the purposes of this work we could say that an agent is a "Web Service agent" if it receives and processes requests formulated by a human user or an application in collaboration with a Web component. The requests may be synchronous or asynchronous.

The development of our solution implies an increase of internal cohesion inside agent-based application components. A deeper analysis on which components exist inside the application "black-box" is shown in figure 2(a). The cohesion gain is achieved by the means of an embedded Web container, that allows the agent-based application to process HTTP petitions in a reliable way. As the agent and web containers are both started on the same Java Virtual Machine operating system process, agents and web components may communicate efficiently by sharing object references in a "virtual channel". The resulting execution model is shown in figure 2(b).

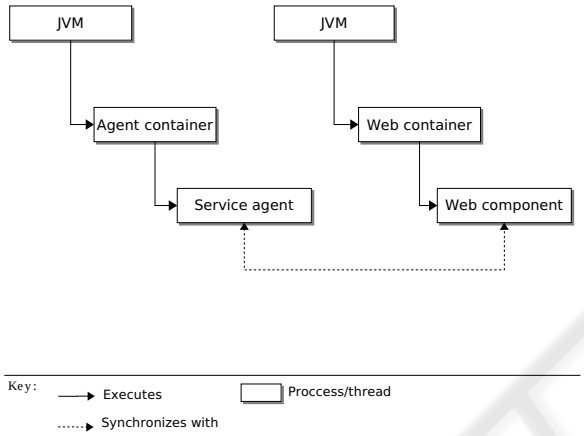
2.2 Implementation

Among FIPA platforms, JADE(Bellifemine et al., 1999) was selected because it is considered well suited for large scale deployments mainly due to its thread-per-agent programming model and the support of "virtual channels" that allow agents to interact with regular Java components(Rimassa, 2003).

In this particular implementation the "Launcher" program, initializes and starts an instance of the JADE platform besides an embedded version of the Tomcat Web Server (Jakarta Project - The Apache Software Foundation, 2003). The mentioned "Registry" is nothing but a data structure that holds references to the running Service Agents, implemented as a Singleton (Gamma et al., 1995).



(a) Component view



(b) Execution model

Figure 3: Gateway architecture

Access to the agents source code is required as they need to be recompiled to include the Web Service capability which is encapsulated in a platform specific library. In JADE's particular case, agents are enhanced with a custom behaviour class, that only requires the addition of one line of code.

The architecture is applicable to FIPA platforms other than Jade; however, it would be necessary to port the framework components (Registry and Launcher) using its particular libraries and program interfaces and to add missing components provided by the platform like virtual channels between objects and agents.

2.3 Evaluation and comparison

Our proposal is not the first solution that allow agents to interoperate with web based components. In fact such an architecture was defined by developers(Berre and Fourdrinoy, 2002) of the Jade platform and later implemented on the WSAI Project(Whitestein Technologies, A.G., 2003) as a contribution to AgentCities initiative(Dale et al.,).

The solution assumes the existence of two agent containers, one standalone, that we may call the "main container" and one contained itself within the web container. Each container is executed in a separate JVM system process. WSAI introduces the concept of "Gateway Agent" as an agent living in the "web container", responsible of translating HTTP requests into ACL messages. The general gateway architecture components are shown in figure 3(a).

One of the major drawbacks of the approach resides in the existence of several processes that should synchronize using remote method invocations even if both of them are deployed on the same machine. Additional complexity comes from the fact that it is able to interoperate with any running FIPA-compliant agent platform (even non Java-based ones) without access to its source code.

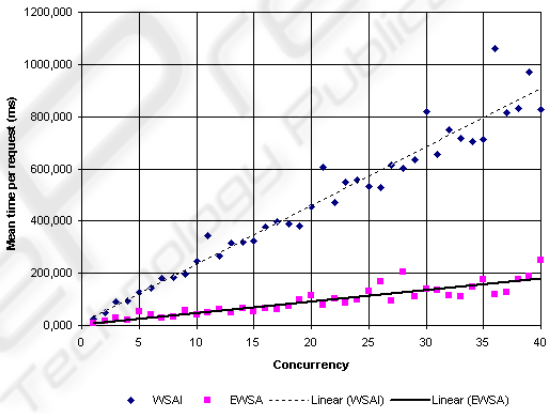


Figure 4: Mean service time for concurrent requests

We believe that in order to build enterprise-class agent-based applications is not critical to provide web-interoperability to an indefinite number of FIPA platforms, therefore, we trade-off this flexibility in favor of good integration with the chosen agent platform, even though the architectural principles remain useful for them. As a result, our framework implementation is simple and provides good performance.

In a benchmark between WSAI and the EWSA decoupled architecture an important performance and scalability gain was observed. A currency exchange Web service is provided in the WSAI platform. The service implementation is trivial as it only consist of a simple mathematical conversion performed by an agent. As shown in figure 4 we may notice that not only EWSA's response times are better, but that they increase at a slower rate with respect to the number of concurrent requests which leads to better scalability. The performance gain in the embedded architecture can be interpreted as an effect of the elimination of network calls overhead between agents and web components.

3 APPLICATIONS

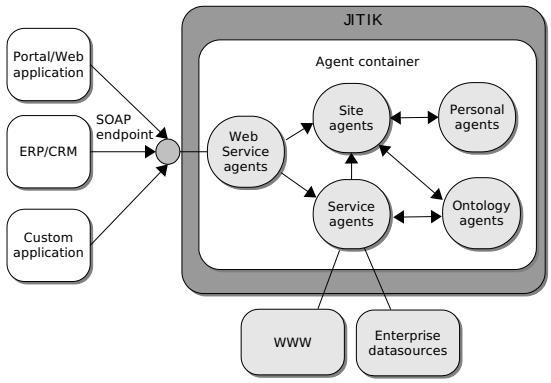


Figure 5: JITIK and enterprise systems interaction

In a general we believe that the proposed integration model is useful to allow gent-based applications to provide knowledge-intensive services, such as:

- Search and automatic classification
- User profile inference
- Semantic-based content distribution

Web-enabled agent systems may serve in a variety of domains. As presented in the JITIK case study, they are well suited to support knowledge distribution in collaborative environments.

3.1 Just-in-time Information and Knowledge

The proposed model have been successfully implemented in JITIK (for Just-in-Time information and Knowledge) that may be defined as a web-enabled agent-based intelligent system capable of deliver highly customized notifications to users in large distributed organizations(Brena et al., 2001). JITIK is aimed to support collaboration within organizations by delivering the right knowledge and information to the adequate people just-in-time. JITIK was designed to interoperate with enterprise systems in order to retrieve and distribute contents in a flexible way.

The JITIK agent model is shown in figure 5. Personal Agents work in behalf of the members of the organization. They filter and deliver useful content according to user preferences. Personal agents are provided of information by the Site Agent who acts as a broker between them and Service agents. For the purposes of this work, the most relevant agents of JITIK are the so called service agents which collect and detect information and knowledge pieces that are supposed to be relevant for someone in the organization. Examples of service agents are the Web Service agents, which receives an process external requests, as well as monitor agents which are continuously monitoring sources of information and knowledge (web pages, databases, etc.).

The ontology agent contains the knowledge about the interest areas to the members of the organization and about its structure(Brena and Ceballos, 2004). That knowledge is hierarchically described in the form of taxonomies, usually one for interest areas and one describing the structure of the organization. For example, in an academic institution, the interest areas could be the science domains in which the institution is specialized, and the organizational chart of the institution gives the structure of the organization.

3.2 JITIK Web Services

JITIK is an example of an agent-based application able to provide knowledge intensive services which may be grouped as follows:

Recommendation services A user's profile is represented by a set of points in the taxonomies, as each user could have many interests and could be located at different parts of the organizational structure. As JITIK keeps track of user interests and preferences it is able to recommend content to users on demand. Recommended content may be used in Portals or Web applications.

Content search and classification One of the main difficulties for web users is obtaining relevant information. In normal conditions people waste a lot of time searching documents on the web, most of the times, because the users must examine the documents in detail to determine if they are really relevant for the search purposes. In the context of JITIK, a service agent that searches the most relevant documents on the web can be constructed. The knowledge that guides the search is handled by the ontology agent where the keywords with which the search engine is invoked are defined. The documents obtained by the search are qualified by a fuzzy system and then the best ones are pushed to the users.

Subscription services JITIK allows users to subscribe to changes in specific areas. Also, users may customize the media and frequency of JITIK notifications using using simple web-based interfaces. Rules may be defined so as messages relative to certain topics are handled with higher priorities. A rule may state that several alerts may be sent to their cell-phone via SMS, and also define that interest-area messages be sent in a weekly summary via email. Organization managers may set high-level distribution rules.

Content distribution services Enterprise applications may deliver content to the system using

its semantic-based content distribution services. When new content is received it is classified and distributed to users who may be interested. Users receive the notifications of new content as specified by their own rules.

As shown above, the EWSA decoupled architecture allows an agent-based application like JITIK to provide enterprise communities with a number of knowledge oriented Web Services, specially useful in large organizations where performance and scalability attributes become critical.

4 CONCLUSION

We have presented an architectural approach aimed to allow integration of multi-agent systems as Web-Services components. Besides its simplicity, the advantage of this approach is that it provides an efficient way of interoperating agent-based subsystems with web-centric loosely-coupled systems. We think this solution is a good compromise given the current status of technology, and that it allows rapid integration of modular systems conforming to open standards.

We presented experimental evidence to support our claim of efficiency. We presented as well a case study, which is the application of our architecture to the JITIK system, a multi-agent system to deliver information items to a distributed community of users.

In the near future we intend to test our architecture in other real-world systems integrating agents in a web-based framework. We are currently studying methodological issues to guide the development of hybrid agents-web systems, as current agent-development methodologies need to be strongly enhanced in order to fit our architecture.

REFERENCES

Barry, D. K. (2003). *Web Services and Service-Oriented Architectures: The Savvy Manager's Guide*. Morgan Kaufmann.

Bellifemine, F., Poggi, A., and Rimassa, G. (1999). JADE - A FIPA-compliant agent framework. In *Proceedings of PAAM'99, London*.

Berre, D. L. and Fourdrinoy, O. (Jun 2002). Using JADE with Java Server Pages. In *JADE documentation*.

Brena, R., Aguirre, J., and Treviño, A. (2001). Just-in-Time Information and Knowledge: Agent technology for KM Bussiness Process. In *Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference*.

Brena, R. and Ceballos, H. (2004). A Hybrid Local-Global Approach for Handling Ontologies in a Multiagent System. In *Proceedings of the 2004 2nd International IEEE Conference Intelligent Systems. Varna, Bulgaria*.

Dale, J., Willmott, S., and Burg, B. Agentcities: Building a global next-generation service environment.

Dickinson, I. and Wooldridge, M. (2003). Towards practical reasoning agents for the semantic web. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 827–834. ACM Press.

FIPA (2002). FIPA Abstract Architecture Specification.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Genesereth, M. R. and Ketchpel, S. P. (1994). Software agents. *Commun. ACM*, 37(7):48–ff.

Hendler, J. (2001). Agents and the Semantic Web. *IEEE Intelligent Systems*, 16(2).

Hunhs, M. (Jul-Ago 2002). Agents as Web Services. *IEEE Internet Computing*, 6(4):93 –95.

Jakarta Project - The Apache Software Foundation (2003). The Tomcat Web Server v. 4.1.

Jennings, N. and Wooldridge, M. (18 Jan. 1996). Software agents. *IEE Review*, 42(1):17 –20.

Jennings, N. R. (2000). On agent-based software engineering. *Artificial Intelligence*, 177(2):277–296.

Labrou, Y., Finin, T., and Peng, Y. (March-April 1999). Agent communication languages: the current landscape. *IEEE Intelligent Systems*, 14(2):45 –52.

Nwana, H. S. and Ndumu, D. T. (1999). A Perspective on Software Agents Research. *The Knowledge Engineering Review*, 14(2):1–18.

Peng, Y., Finin, T., Labrou, Y., Chu, B., Long, J., Tolone, W. J., and Boughannam, A. (1998). A multi-agent system for enterprise integration. In *Proceedings of the 3rd International Conference on the Practical Applications of Agents and Multi-Agent Systems (PAAM-98)*, pages 155–169, London, UK.

Petrie, C. J. (Dic 1996). Agent-Based Engineering, the web and intelligence. *IEEE Expert*, 11(6):24–29.

Preece, A. and Decker, S. (Ene-Feb 2002). Intelligent web services. *IEEE Intelligent Systems*, 17(1).

Rimassa, G. (Jan 2003). Runtime Support for Distributed Multi-Agent Systems. In *Ph. D. Thesis, University of Parma*.

Sun Microsystems, Inc. (2003). JSR-000154 Java(TM) Servlet 2.4 Specification (Final release).

W3C (2000). Extensible Markup Language (XML) 1.0 (Second Edition).

W3C (Aug 2003). Web Services Glossary, Working Draft.

Whitestein Technologies, A.G. (2003). Web Services Agent Integration Project.