Demonstration of WS2JADE

Xuan Thang Nguyen
Swinburne University of Technology
John Street, Hawthorn
VIC 3122, Australia
Tel. (61)392145623

xnguyen@it.swin.edu.au

ABSTRACT

Integrating Agents and Web services has recently attracted considerable attention from people in both the agent and Web services communities. The future of intelligent agents with autonomous capabilities, which manage and access the widespread Web services infrastructure, is promising. Our demonstration shows how WS2JADE, a toolkit developed at the Centre of Intelligent Agent and Multi-Agent Systems, achieves first steps in this direction. The demonstration describes how Web services can be accessed and used by Jade Agents and how other Agents can take this advantage to build value-added services within e-composition.

1. INTRODUCTION

With the emergence of the Web services standards, universal interoperability between applications is becoming a reality. Web services follow a loosely coupled integration model and use industry standard protocols which facilitate a seamless integration of heterogeneous systems. While the focus of Web services is on infrastructure and interoperability, Agents are well-known for their autonomous and problem solving capabilities in a distributed environment. Therefore, an integration of Web services and Agents could create an environment where each technology can employ and compliment each other's strengths. In this paper, we present an overall description and demonstrations of WS2JADE toolkit, our software developed for integrating Web services and Jade Agents. WS2JADE software is our first effort toward a broader aim of using Agents to access and manage the widespread Web service infrastructure.

2. WS2JADE TOOLKIT

A symmetric integration of Web services and FIPA-compliant Agent platforms has been proposed in [1] as a high-level architectural recommendation from the AgentCities. There have been a few implementations followed this recommendation. WSDL2JADE [4], available from Sztaki's Website at http://sas.ilab.sztaki.hu:8080/wsdl2agent/index.html, can generate agent ontologies and agent codes from a WSDL input file. WSIGS (Web Services Integration Gateway Service) [2][3], implemented by Whitestein Technology, supports bi-directional integration of Web services and Jade Agents. The distinctive feature of our WS2JADE system, as compared to those software, is that WS2JADE allows deployment of Web services as Jade Agents' services at run time. Hence, it provides a greater level of automation in Web services discovery and Web services usages.

In WS2JADE, Web services are visible to FIPA-compliant Agents through proxy Agents which reside in WS2JADE system. Web services are seen by FIPA-compliant agents as Agent services offered by the proxy Agents. The mapping from Web services to proxy Agents are many to many. Since in JADE, an Agent is often single-threaded, offering the same Web services on different proxy Agents allows concurrent accesses to a Web service. Offering more than one Web service on a proxy Agent allows related Web services to be grouped together. In WS2JADE, Web services discovery is done through UDDI proxy Agents. These Agents support special discovery services which can be configured to proxy to any UDDI version 2 servers, including Microsoft and IBM UDDI inquiry servers.

WS2JADE is written entirely in Java. It can run on any machine which has JVM 1.4.x installed. WS2JADE uses Axis 1.1 for handling SOAP messages. Its current version is 1.2 and it can be downloaded from our Website at http://www.it.swin.edu.au/centres/ciamas.

3. DEMONSTRATIONS

3.1 Find-and-Bind demonstration

If Agents can access and use Web services, Agents can consequently discover Web services. This is because Web services searching facilities normally expose their interfaces as Web services. UDDI inquiry and publish engines are examples. The UDDI inquiry interface in WSDL format is published by UDDI.org and can be found at their website: http://uddi.org.

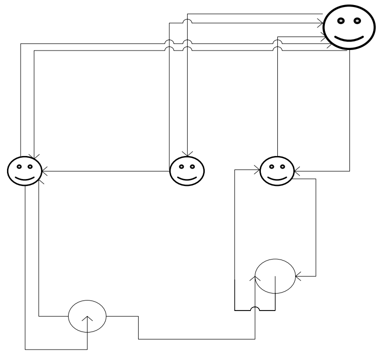

Figure 3-1: Find-and-Bind

In WS2JADE, an Agent which hosts the proxy of Microsoft UDDI inquiry service is started by default. The name of this Agent is UDDI Agent. In this demonstration, a client Agent

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands
Copyright 2005 ACM 1-59593-150-2/05/0007

searches for a Web service through this UDDI agent and uses a selected service from the search result. The interaction flow is illustrated in figure 1. As can be seem from the figure, from the client Agent's side, it needs to do a search on the UDDI Agent (step 1) for a wanted Web service. After getting search results back (step 2) from UDDI Agent. The client Agent, bases on its own references, determines a service it wants to use and queries the WSManager Agent on how to use this Web Service. The WSManager Agent informs the client the address of the Agent which can offer a proxy service of this Web service (step 4). The client now can start use the service (step 5 and 6).

To examine what happen inside WS2JADE, as mentioned before, WS2JADE proxies the MS UDDI (step 1A) through the UDDI Agent to fulfil the client request at step 1. After step 3, the WS Manager creates a new Agent and deploys a new Agent service which is a proxy of the wanted Web service. The WS Manager also registers this Agent on the DF (Directory Facilitator). The address of this Agent is returned back to the client Agent. WS invocation is again done indirectly through the proxy service of the newly generated Agent.

3.2 Composition of Web services

This demonstration is a next step of the previous one. We have a scenario in which three Agents: P (Pay Friend), A (Amazon), and G (Global Transport), have a composition plan for offering online item purchase. Such a plan needs to take into account online payment transaction and product delivery. In the plan, P is responsible for client payment. A is responsible for shopping cart. G is responsible for item delivery.

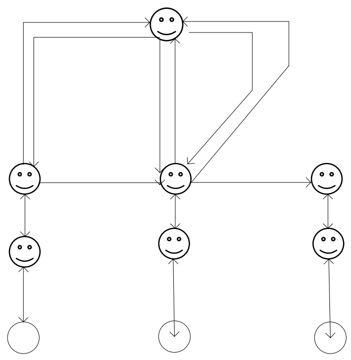

Figure 3-2: Web services composition with Agents

The interaction sequence is depicted in figure 2. First, the client Agent searches for the products and add them to its shopping cart (step 1 and 2, repeated). Once the client Agent does a checkout, Agent A informs it payment details with Agent P. The client Agent then contacts P to do payment. After the payment is made, Agent P informs Agent A whether the payment is successful (step 6.1). If it is, Agent A sends a message to Agent G and asks for item delivery.

On the abstract level, no concrete implementation of services is described in the plan. The services that these Agents use are Web services. P, A, and G Agents, base on the requirements of their own services, try to find and bind Web Services. How this can be done with WS2JADE is explained in the first demonstration. We skip these steps and assume that after find-and-bind steps, Agent P becomes a proxy of PayFriend WS, Agent A becomes a proxy of Amzaon WS, and Agent G becomes a proxy of GlobalTransport WS. The composite service now can go into operation. Because of the demonstration purpose, transactions are preferably not commited. PayFriend WS and GlobalTransport WS emulate essential functionalities in the interfaces of PayPal and Global Transport Web Services, however, without real transactions to any banks.

4. EVALUATIONS AND FUTURE WORK

Although WS2JADE has demonstrated its effectiveness in working with different popular Web services as shown in the above demonstrations, there are still areas that could be further improved. One-way integration is one of them. At the moment we are reluctant in any Agent to Web services implementation as we believe that there is still a lack of substantial theoretical work on the topic of agent to Web Service integration, especially in the areas of translating Agents' stateful communication model into Web services' stateless communication model and building asynchronous interaction framework for Web services. This is a subject of our on-going research. Our current and future work also involves improvements of the semantic processing capability of WS2JADE's ontology management component.

5. ACKNOWLEDGMENTS

I would like to acknowledge the contributions of Alasdair Grant and Mohan Baruwal Chhetri in WS2JADE 1.2 development.

6. REFERENCES

[1] Agentcities Web Services Working Group. "Integrating Web services into AgentCities", Technical Recommendation available at http://www.agentcities.org/rec/00006/

[2] D. Greenwood, M. Calisti, "An Automatic, Bi-Directional Service Integration Gateway", ", *IEEE Systems, Cybernetics and Man Conference*; 10-13 October, 2004, the Hague, Netherlands

[3] D. Greenwood, M. Calisti, "An Automatic, Bi-Directional Service Integration Gateway", ", *IEEE Systems, Cybernetics and Man Conference*; 10-13 October, 2004, the Hague, Netherlands

[4] L. Zs. Varga,Á. Hajnal: "Engineering Web Service Invocations from Agent Systems". Proceedings of the 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003, Prague, Czech Republic, pp. 626-635, June 16-18, 2003.

[5] Telecom Italia Lab. JADE (Java Agent Development Framework), available at http://sharon.cselt.it/projects/jade