ABSTRACT

In this article, 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. We present case studies of applications that have been enhanced by our proposal.

Keywords: software agents; Web Services

INTRODUCTION

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

Several authors have pointed out some overlapping areas between agents and Web Services’ semantic capabilities (Hunhs, 2002; Preece & Decker, 2002). However, issues regarding how they may be competing or complementary technologies remain open (Petrie, 1996). Because of that, research involving agents and Web Services is focused mainly on building improved semantics (Dickinson & Wooldridge, 2003; Hendler, 2001), communication languages and interaction protocols (Labrou, Finin, & Peng, 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 & Ndumu, 1999; Genesereth & Ketchpel, 1994).

In this work, we present a decoupled architectural approach and design principles called Embedded Web Services Architecture (EWSA) 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 the following:

- Web Applications, Portals, and Content Management Systems (CMSs)
- Enterprise Resource Planning (ERP)
- Manufacturing Execution Systems (MES)
- Workflow Engines and Business Process Management Systems (BPMSs)

This integration allows agents to publish XML (W3C, 2000) Web Services (W3C, 2003b) or standard HTML, thus providing a convenient interface for other distributed components. The Web Service architecture is understood widely as a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically, WSDL) (W3C, 2001). Other systems interact with the Web Service in a manner prescribed by its description using SOAP messages (W3C, 2003a), typically conveyed using HTTP with an XML (W3C, 2000) serialization in conjunction with other Web-related standards (W3C, 2003b).

Regarding the external behavior of the agent-based application, our approach fits into the design paradigm identified as Service Oriented Architecture (SOA). The SOA foundation ideas were introduced by Arsajani (2001); he defines SOA as “the architectural style that supports loosely coupled services to enable business flexibility in an interoperable, technology-agnostic manner.” SOA consists of a composite set of business-aligned services that support flexible and dynamically reconfigurable end-to-end business processes realization using interface-based service descriptions (Borges, Holley, & Arsanjani, 2004).

This article also discusses 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. The rest of the article is organized as follows. In the next section, we provide an overview of our solution approach. Specifically, we discuss the proposed Embedded Web Server Architecture for integrating agent-based applications and enterprise applications and its implementation. Then, we discuss its evaluation and application to some example domains. Finally, the article concludes with some expected business results.

**SOLUTION OVERVIEW**

Instead of making an agent-based application look different, compared to other applications from the outside, which is, indeed, a “religious” point of view frequently in the agent research community, we intend to hide the agentness of a group of agents from the outside.

We contend that agents should solve problems for which they are well suited and
Related Content

Ensemble of SVM Classifiers for Spam Filtering
[www.igi-global.com/chapter/ensemble-svm-classifiers-spam-filtering/10303?camid=4v1a](www.igi-global.com/chapter/ensemble-svm-classifiers-spam-filtering/10303?camid=4v1a)

Lightweight Key Management for Adaptive Addressing in Next Generation Internet
[www.igi-global.com/article/lightweight-key-management-for-adaptive-addressing-in-next-generation-internet/176713?camid=4v1a](www.igi-global.com/article/lightweight-key-management-for-adaptive-addressing-in-next-generation-internet/176713?camid=4v1a)

Ranking Pages of Clustered Users using Weighted Page Rank Algorithm with User Access Period
[www.igi-global.com/article/ranking-pages-of-clustered-users-using-weighted-page-rank-algorithm-with-user-access-period/139738?camid=4v1a](www.igi-global.com/article/ranking-pages-of-clustered-users-using-weighted-page-rank-algorithm-with-user-access-period/139738?camid=4v1a)