Chapter 3
Extensible Java EE-Based Agent Framework – Past, Present, Future

Milan Vidaković, Mirjana Ivanović, Dejan Mitrović*, and Zoran Budimac

Abstract. EXtensible Java EE-based Agent Framework (XJAF) is a modular, FIPA-compliant multi-agent system developed by the authors of this chapter. The main motivation behind the development of XJAF was to demonstrate how existing, standardized Java EE technologies, tools, and libraries, such as JNDI, JMS, and EJB, can be used to implement a large subset of functionalities required from a multi-agent system. Immediate direct benefits of this approach are shorter development time of the system itself, delegation of agent load-balancing to the enterprise server, flatter learning curve for new developers of the system, etc. The first implementation of XJAF has been published several years ago and has since been used in several real-life applications. In the meantime, some disadvantages and weaknesses of the system were noticed, and the work is underway to provide a new implementation with an improved quality. The most recent focus of improvements has been on the addition of fault-tolerant techniques, and the increase of interoperability through a SOA-based design and web service interfaces.

1 Introduction

Agent technology represents one of the most consistent approaches to distributed software development. Software agents can be defined as executable software entities with varying degrees of intelligence, that act autonomously while pursuing their goals.

Milan Vidaković
Faculty of Technical Sciences, University of Novi Sad, Serbia

Mirjana Ivanović · Dejan Mitrović · Zoran Budimac
Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Serbia
e-mail: [dejan@dmi.uns.ac.rs](mailto:dejan@dmi.uns.ac.rs)

* Corresponding author.

springerlink.com © Springer-Verlag Berlin Heidelberg 2013
A multi-agent system (MAS) is a software system with an infrastructural support for its agents. Its core functionalities include the agent life-cycle management, a messaging infrastructure, and a service subsystem that effectively supports agents, giving them the possibility of accessing resource, executing complex algorithms, etc. In addition, MASs in many cases offer security features (e.g. agent code and data integrity, and encryption of messages), a connection mechanism that allows cooperation of agents in physically distributed environments, and a support for agent persistence and mobility.

By analyzing MAS solutions that existed back in 2003, however, it was concluded that in the majority of cases many of the functional requirements were implemented from scratch. Developers used their own implementations of the message exchange infrastructure, directories of agents and services, and security features, instead of re-using technical solutions to these problems already present in Java EE.

EXtensible Java EE-based Agent Framework (XJAF) is a FIPA-compliant MAS developed by the authors of this chapter. Given the authors’ own practical experience in developing Java EE-based applications, as well the strong interest in the agent technology (e.g. [40]), the initial motivation for the development of XJAF was to evaluate whether MASs can benefit from re-using existing, standardized Java EE tools and libraries. At the same time, there was a need for an efficient virtual central catalog implementation for the library information system BISIS. Additionally, the DIGLIB system for Networked Digital Library of Theses and Dissertations required a framework to harvest metadata from heterogeneous content providers. The two main design goals for XJAF have thus been defined:

- To provide an efficient and standards-compliant MAS implementation. Java EE has been chosen as the main implementation platform, given its success in the development of scalable, secure, and reliable software solutions for large enterprises.
- To develop a framework with aforementioned problems in minds, but with a high level of reusability. This has been achieved by defining a system of abstract services that can be specialized to serve the needs of a particular application.

The choice of Java EE as the XJAF implementation platform has proven to be beneficial. Direct advantages of this approach were shorter development time of the system itself, and harnessing of advanced programming features such as runtime load-balancing. More concretely, XJAF relies on the following Java EE technologies for its functioning:

- **Java Naming and Directory Interface (JNDI)**: used for implementing directories of agents and services.
- **Java Message Service (JMS)**: provides the communication (inter-agent messaging) infrastructure.
- **Enterprise JavaBeans (EJB)**: used as placeholders for agents and services
- **Java Serialization**: supports agent persistence and mobility.
- **Java PKI API**: used for implementing some of the security features.