Chapter 6

AGENT FRAMEWORKS
Chapter 6 Agent Frameworks

6.1 Mobile Agent Technology

This chapter introduces the different agent technologies, also gives a thorough understanding of these technologies and the advantages of the same over other existing technologies.

Mobile agent paradigm, receiving a great attention, in the past few years, has received an immense consideration. After gaining important stimulation in research area, MA’s have not used in a considerable existent applications. This research also shows an evaluation of different MA technologies and implementation of JMDCHS using JADE Mobile agent technology.

In today’s world there are various agent development platforms available for development of different kinds of application. Very little work is done on a study of which agent technology is good for a development of agent based application. This thesis also elaborated on a study of a variety of agent technologies along with their toolkits in profundity, with analysis of their merits and demerits. This research also gives a comparison among different mobile agent framework like Aglets, Voyager, JADE, TACOMA, Grasshopper, SPRINGS, Tryllian’s Agent Development Kit, and Zeus.

6.2 Existing mobile agent systems

Agents which are java enabled have operating system independent language as well as the standards, inherent in Java such as the object serialization mechanism and Java virtual machine. The different MA framework is given in the following section.

6.2.1 Aglets

Aglet mobile agent system is IBM's product. Aglet is formed with the mixture agent plus applet so that mobility can be brought to Java applets. Aglets are a java based general-purpose agent technology where MA can transmit itself among two agent platforms. It is developed by IBM, Tokyo in 1996. A Java (aglet) object will migrate within internet between two hosts because aglets are abstract class. An aglet as a feature of mobile agent also can dispatch itself to another host to resume the execution, once it
has finished executing on one host, can also discontinue its execution. There is an Applet class in a java library; Aglet paradigm makes use of an event driven approach in this library. To define the aglet behavior, each aglet implements a various events handler operation. [D97][GDM97][CS94]

For agent’s communication which is based on passing of message, aglet uses single-thread model which supports both synchronous as well as asynchronous communication. MA in Aglets uses proxy which refers remote agents for sending messages. Long-running tasks execution is avoided by the programmer because each agent is allocated only thread or else, the agent’s events will be prevented for incoming messages. In case two agents send synchronous message to each other at the same time, then this single-thread model could create deadlocks [GK11][GDM97][D97][DM98].

Even if Aglets have describe specifically in the area of mobile agents, but the dummies it offers are not active dummies is actually drawback of the platform. [TIM03]

6.2.2 Voyager

Voyager, Java based agent paradigm, invented by Object Space in 1997 is a flexible conventional techniques to create network applications. Object Space has created an agent based technology which supports traditional as well as agent related distributed computing techniques created by Object Space. Voyager supports object request brokering so developers can create distributed application using both traditional messaging, such as CORBA or RMI, as well as agent-enhanced techniques [FGD07]. It makes managing of remote communication simple through RMI protocols because it of its distributed computing middleware. The main drawback of Voyager is that, their business software is not freely available, which prevent many researchers from using it. Agent remains alive for a certain time period. They can live remaining not active in a particular time. [TIM03][G99][LMP03]

6.2.3 JADE

It is a Java Agent Development framework, middleware which is most popular among agent-based system now days. JADE has a flexible infrastructure which can be modifiable according to the users’ requirement with JADE’S add-on modules.
JADE has its runtime environment with which a development of an application can be facilitated. This runtime environment provided by JADE, gives an implementation of all the benefits of java over all other language features along with third-party libraries, because of these developers can develop a JADE multi-agent system without much knowledge in an agent theory. Initially development of Jade done by the R & D Department of Telecom Italia s.p.a., later distributed as open source under the LGPL license.\[2\][3][4][5]

It is an agent implementation technology for simplifies the development of agent based application. For supporting the mobility of MA within containers of the JADE AP, the JADE has in-built Agent Mobility Service available. According to the FIPA specifications on querying the AMS Agent Management System, an agent searches the current location of its target. Telecom Italia Lab (Tilab), who originated JADE, initially, developing, improving and maintain JADE.\[TIM03\][15][16].

6.2.4 TACOMA

The next MA framework is a TACOMA means Tromoso And Cornell Moving Agents is developed by Tromoso and Cornell University. Static and dynamic agents of computing system are considered in TACOMA considers. When an agent move among sites, a DATA file stores the status information and CODE file stores a state of agent as the agents which are portable and operate in stateless mode and. The agent program taken from the data file of CODE and run the agent using the file data for its data and information on the status of the agent, when it is at receiver site.

6.2.5 Grasshopper

Grasshopper implementing MASIF support is a relatively latest platform for developing mobile agent application. This application is a telecommunication based applications. Grasshopper is developed by IKV++ in nineteen. Grasshopper MA framework is made of various regions. It provides a graphical user interface to manage agents and their regions. Because of the regions, developer gets dynamic proxies benefit. A call to an agent which is
affecting can end up executing on the dummy agent at origin, is a major shortcoming of Grasshopper is that. [CT99][7]

### 6.2.6 SPRINGS

SPRINGS agent development toolkit is developed by the DISG that is Distributed Information Systems Group at the University of Zaragoza in Spain which concentrates on scalability and reliability. SPRING also propose a hierarchical infrastructure of regions. It does not support agent communication using the standard FIPA, a major drawback of SPRINGS. Also there are no sophisticated security mechanisms. This platform does not provide graphics based tools to the user but still is easy to use.

### 6.2.7 Tryllian’s Agent Development Kit

Tryllian’s Agent Development Kit (Tryllian’s ADK) contains various Habitats extend over servers, each hosting a number of Rooms, is a complete development and implementation environment. One or more Rooms that share a Java Virtual Machine make a Habitat. The services provided Tryllian’s ADK are agent lifecycle administration, message, inter-Habitat journey, Room and agent persistence and security model.

### 6.2.8 Zeus

The ZEUS is a developed by British Telecommunications Lab. It is open source, freely available software. The ZEUS is used for the fast implementation of novice multi-agent system by abstracting into a toolkit the common principles and components. In order to analyse, plan implement and manage MA application, ZEUS offers a collection of software elements and tools. Moreover, a runtime environment, which enables applications to be observed along with reports tool, statistics tool, agents and society viewer are provided by ZEUS. It has an outstanding debugging and GUI, general purpose planning and scheduling mechanism. The documentation in ZEUS is extremely scrawny, which lead to problems to create new applications. It supported platforms: Java 2, implemented standards are FIPA, makes use of communication language like KQML and ACL. It does not provide mobility.
6.2 Comparison of Mobile agent Platforms:

Depending on major features, the different mobile agent platforms are compared as shown in Table 6.1. The following characteristics are evaluated for each mobile agent technology.

- the major elements of that platform
- Whether it supports communications, if yes then whether it is synchronous or asynchronous.
- What programming language is supported by mobile agents?
- whether proxy consent supported or not
- whether a movement by specifying a user-friendly name is allowed
- download available freely
- contains graphical tools
- Support for a security mechanisms

Table 6.1 a table to show Comparison between various Mobile agent technologies

<table>
<thead>
<tr>
<th>MA Platform Features</th>
<th>Aglets</th>
<th>Voyager</th>
<th>JADE</th>
<th>Grasshopper</th>
<th>TACOMA</th>
<th>SPRINGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>Partial security</td>
<td>Partial &amp; Secured channel</td>
<td>Strong</td>
<td>Partial</td>
<td>Uses firewall agent</td>
<td>Partial</td>
</tr>
<tr>
<td>Model</td>
<td>Events</td>
<td>Procedural</td>
<td>Traditional procedural</td>
<td>Procedural</td>
<td>Behaviors</td>
<td>Procedural</td>
</tr>
<tr>
<td>Communication technique</td>
<td>Synchronous and Asynchronous</td>
<td>All methods</td>
<td>Only Synchronous</td>
<td>Synchronous and Asynchronous</td>
<td>Only Asynchronous</td>
<td>Only Asynchronous</td>
</tr>
<tr>
<td>Elements</td>
<td>Contexts</td>
<td>Servers</td>
<td>Places</td>
<td>Places</td>
<td>Agencies</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
<td>-------------</td>
<td>------------</td>
<td>------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Agents(agents) Agents Regions (RNSs)- Tahiti-Agents Habitats Platforms Plugins</td>
<td>Agents</td>
<td>Regions</td>
<td>-Containers Main container-DF, AMS, MTS</td>
<td>Agencies</td>
<td></td>
</tr>
<tr>
<td>Mobility</td>
<td>Aglet transfer protocol</td>
<td>Java object series</td>
<td>Dynamic proxies (mobility is location wise)</td>
<td>Dynamic Proxies (region server)</td>
<td>In-built agent mobility service</td>
<td>Mobility through Transfer control protocol</td>
</tr>
<tr>
<td>GUI Based tools</td>
<td>Some</td>
<td>No GUI</td>
<td>GUI is there</td>
<td>No GUI</td>
<td>GUI is there</td>
<td>Some GUI</td>
</tr>
<tr>
<td>Organization</td>
<td>IBM Tokyo research</td>
<td>Object space</td>
<td>IKV++ Distributed Information Systems Group</td>
<td>Telecom Italia Lab</td>
<td>Tromoso and Cornell University</td>
<td></td>
</tr>
<tr>
<td>Operating system used</td>
<td>JDK 1.1.x on Win32, OS/ 2 Warp Version 3 and 4, AIX 4.x, Solaris for SPARC.</td>
<td>UNIX, Linux Windows</td>
<td>Windows NT/ 9x, Solaris Should run on all platforms supporting JDK 1.1</td>
<td>Linux</td>
<td>All (with JRE) Unix,Win95, Win NT, PDA systems</td>
<td></td>
</tr>
</tbody>
</table>
This research analyzed eight agent development tools developed by different groups. On comparison, Jade agent development environment seems most attractive. It is open source platform, purely designed in Java, provides consistency in API, standards compliant, popular, available, maintained, supported by a community and more. It provides good security features and supports sound agent mobility. Among other tools Voyager is commercial tool and doesn’t comply with FIPA standards. Zeus supports FIPA standards but doesn’t provide agent mobility. Aglet also doesn’t comply with FIPA, lacks security and scalability. Anchor provides good security but doesn’t follow FIPA specifications, thus lacks scalability. Thus JADE agent development toolkit is most balanced toolkit among the eight discussed in this work. JADE is free open source, good documentation, very good GUI, acceptance of users, used in many development projects. That becomes a reason of selecting JADE for the deployment of multi database decision making in a catastrophe healthcare system.