IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITP4169

Java Mobile Agent and Project Management

F. Xue and K. Y. R. Li
School of Business Systems and School of Multimedia Systems, Monash University, Australia
Raymond.Li@infotech.monash.edu.au

ABSTRACT

This article examines what is a Mobile Agent and how it can help businesses to implement client-server enterprise computing solutions. A Java Mobile Agent-based project management system prototype is presented to demonstrate the main features of Mobile Agents and how they help to enhance communication processes and facilitate security within the project environment.

PREAMBLE

Business transactions require human activities, such as information collection and analysis, and human interactions, such as negotiation. With the rapid growth of Internet technology over the past few years, e-business has started to transform the ways in which we conduct our business. Metaphors, such as e-shopping carts and e-shops, are helping us to implement Business to Consumer Business. These working models often require a buyer to visit the vendors' web sites. The data collected is then analyzed and a transaction completed, often without much interactive negotiation. Other technologies now exist to provide better ways to implement e-business. Mobile Agents, which can enable automation and negotiation, are receiving a lot of attention both from researchers and industrialists. Mobile Agents are a new approach that can provide flexible enterprise-computing solutions instead of relying on traditional message-based architectures. Mobile Agents are made possible by emerging technologies such as Object-Oriented Technology, Remote Methods Invocation and the Internet.

A Mobile Agent can migrate from machine to machine in a heterogeneous network under its own control [3]. It is capable of roaming wide area networks (WANS) and particularly the world wide web (WWW); interacting with foreign hosts; collaborating with other Agents; gathering information on behalf of its owner; and coming 'back home' having performed the duties pre-defined by its users.

Technically, a Mobile Agent is a special program that moves itself, an executable object, between various computers in one, or multiple systems. It conducts designated tasks pre-defined in its internal codes. During its self-execution, it can decide to move to another machine [1]. As intelligent agent, it can interact with other agents, respond to external messages, and travel along a "self-directed" itinerary.

In terms of Object-Oriented Programming, a real application consists of lots of functional objects, each an abstraction of related attributes and methods. Normally, these objects are activated only in local machine memory. An Agent, however, is an object that is specially programmed so that it can move to different locations. The object can linger in the memory of a foreign machine and perform functions for which it was programmed. Agents are autonomous. They can remain unattended for a long time and be activated automatically when pre-defined conditions occur. Alternatively, Agents can be programmed to become collectable garbage and to release its resources.

Agents can co-operate or communicate. This occurs when an Agent makes the location of some of its internal objects and methods known to other Agents. In this manner, an Agent exchanges specified data or information with other Agents without necessarily giving all information away.

Agents can also work together collaboratively. Multi-Agents can be deployed so that each Agent performs one sub-function to complete a particular task. Agents communicate with each other while performing their functions in the same way as a team of humans communicates when completing a task.

Mobility is neither a unique necessity nor a sufficient condition for Agent-hood. An Agent can be executed on one computer system (Stationary Agent), or be executed on different systems at different times (Mobile Agent).

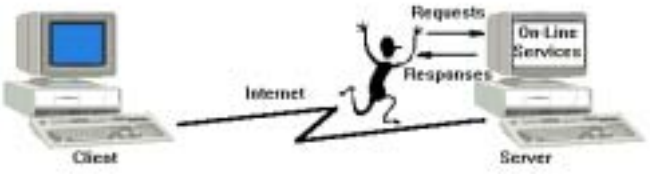
EXAMPLES OF MOBILE APPLICATIONS

Agent applications include electronic commerce, group collaboration, workflow automation, active messaging, event monitoring, information gathering, and distributed simulation and network management. The following three examples demonstrate the use of Mobile Agents.

Agent-Based Web Information Retrieval System

An Agent is sent from the client to the server to obtain information. The Agent obtains the information at the server and returns to its owner's machine with the requested information.

Figure 1: An agent-based information retrieval system



Agent-Based Web Search Engine

A mobile agent's owner, with an instruction to purchase a particular commodity, launches it onto the web. The Agent will first visit the machine that provides an index directory service to determine which shops to visit. The Agent then visits all the selected shops and obtains the price of the item intended to purchase from each shop. The Agent then makes the decision to go back to the shop that provided the best price and has stock available to fill the order. The Agent places the order and travels back to report to its owner.

The Personal Adaptive Web Sentinel (PAWS)

PAWS is an Agent based library of information. Based on the owner's preferences and the index services, such as Alta Vista and

Figure 2: Agent-based shopper assistant

This paper appears in *Issues and Trends of Information Technology Management in Contemporary Organizations,* the proceedings of the Information Resources Management Association International Conference.

Yahoo, the Agent updates existing information, adds new information and purges old data from the personal information library automatically.

PAWS itself, is a set of small communicating modules in which the kernel is responsible for controlling the other modules and for the communication with external Agents [2].

Why Mobile Agents?

Mobile Agent technology should be seen as an alternative approach to traditional client-server architecture, as well as a better solution for distributed systems. In the case of the management of distributed resources, a comparison between a client-server solution and a Mobile Agent-based approach demonstrates that Mobile Agent technology offers a number of advantages. These include flexibility and scalability of the system, load balancing, on-demand services, low traffic in the network, better performance in execution and many others. The advantages are due to the manner in which Mobile Agents treat distribution problems by using local interaction and mobile logic. The following list describes several situations where agent techniques can generate better solutions than traditional a approach:

1. "Local messages are often between 1,000 and 100,000 times faster than remote messages"[7]. If a system is required to send a great number of messages to objects in remote locations, an Agent can be constructed to visit each remote machine in turn and send the messages locally. Besides the efficiency of execution in a local machine, this approach also reduces lots of network I/O communication. It can also eliminate data transaction waiting time.
2. If a task must be performed independently of the computer that launches the task, a Mobile Agent can be created to perform this task. [7] An Agent can move into the network and complete the task in a remote program. For example, an Agent can be launched onto the network from a PDA utilizing wireless technology. After the launch, the PDA can be switched off and turned back on later to retrieve the Agent, together with the retrieved data.
3. To execute programs in parallel, processes can be partitioned amongst several agents who migrate to remote machines and collaborate to achieve the overall goals.
4. Remote agents can be sent and stationed in a remote location and monitor local events.
5. An agent can be designed for a portable device such as PDA, notebook or Java pager that is only occasionally connected to the network. The agent can return home after the source machine is reconnected.

According to Harrison, Chess, and Kershenbaum, of the IBM T.J. Watson Research Centre:

"While none of the individual advantages of Mobile Agents… is overwhelmingly strong, we believe that the aggregate advantage of Mobile Agents is overwhelmingly strong."

A Mobile Agent Assisted Project Management System Prototype

A typical mobile agent has three basic components; functionality, itineraries and mobility. This project was designed to construct an agent that could be launched by a project manager onto the network. The Agent will then travel, based on the itinerary specified by the project executives, to visit each server or workstations at the specified remote sites. The mobile agent will interact with the Stationary Agent at the remote machine to get permission to obtain from the site's project database the status of specified project activities. Based on the functions that the Agent is programmed to perform and the status of the project, several different responses will occur. Upon completion of both the itinerary and remote tasks, the Agent will return and submit to the project manager the retrieved information.

The Development Environment

There are many Agent tools from different vendors such as IBM and Objectspace. Table 1 lists some of the popular Agent software packages available on the market.

Table 1: Some popular agent platforms

Product	URLs	Language	Description
AgentBuilder®	www.Agentbuilder.com	Java	Integrated Agent and Agency Development Environment
Agentalk	www.ked.ntt.co.jp/csl/msrg/topics	LISP	Multi-Agent Coordination Protocols
Aglets	www.trl.ibm.co.jp/aglets/	Java	Mobile Agents
Concordia	www.meitca.com/HSL/Projects/Concordia/	Java	Mobile Agents
DirectIA SDK	http://www.animaths.com/	C++	Adaptive Agents
IGEN™	http://www.cognitiveAgent.com/	C/C++	Cognitive Agent Toolkit
Intelligent Agent Factory	http://www.bitpix.com/	Java	Agent Development Tool
JACK Intelligent Agents	http://www.Agentsoftware.com.au/jack.html	JACK	Agent Development Environment
Microsoft Agent	msdn.microsoft.com/msAgent/default.asp	Active X	Interface creatures
Network Query Language	http://www.nqli.com/		Programming Language
Pathwalker	http://www.fujitsu.co.jp/hypertext/flab/free/paw/	Java	Agent-oriented programming library
Voyager	http://www.Agentbuilder.com/AgentTools/Voyager	Object Space	Java

Four of the above tools were evaluated. They were Aglets from IBM Japan [4]; Concordia from Mitsubishi Electric Information Technology Centre America [5]; JACK Intelligent Agents from Agent Oriented Software; and Voyager from Objectspace.

Aglets

An agent developed by IBM, Aglets is a Java-based object that can move from one host on the Internet to another. In other words, an Aglet agent running on one host machine is able to halt execution whenever needed, dispatch itself to another remote machine, and resume its previous process there. The Aglet carries with it its program code as well as its state (data) while moving from one machine to another. The ability for the machine to safely host Aglets is provided by a built-in security mechanism [6] [8] [9]. In particular, aglet is now an open source.

Concordia

To develop and manage Mobile Agent applications on computer networks, CONCORDIA provides a full-featured framework. With Java Virtual Machine, access to information on any sites is available. Concordia applications can process data even if the user is disconnected from the network. Both desktop computers and wireless portable communication devices such as laptops, PDA and Smart Phones, are supported.[5]

JACK Intelligent Agents

The JACK Intelligent Agent System provides the architecture and capability for developing and running software Agents in distributed applications. It uses a Java-extension Language (JACK) to retain all the benefits of the Java language. JACK has outstanding security, but is not as easy to use as advertised. It is an environment to build and integrate commercial multi-agent systems using built-in components. It has a special language specification, compile methods and object-oriented design to allow easy extension for new agent models.

SELECTING AGENT DEVELOPMENT PLATFORMS

The above tools are evaluated based on the following criteria:

1. Stability and security
2. Commercial software with updating service
3. Compatibility on platforms, such as Java
4. Mobility and functionality
5. Convenience for use

Based on the above criteria, Voyager ORB was selected.

Voyager® is a Java-based Object Request Broker (ORB). It is an enhanced broker using the technology of Mobile autonomous Agents and remote method invocation. It also supports CORBA with distrib-

uted services such as directory, persistence, and publish subscribe multicasting. With Voyager ORB®, we are able to create sophisticated network applications quickly and easily. Voyager® follows Java message syntax to construct remote objects, send them messages, and move them between applications. [7] Voyager allows java objects to move themselves and resume executing. In this way, the instance of an object can act independently on the behalf of its owner, even when its home is disconnected.

Voyager® ORB is intended for business-application programmers and B2B component builders.

> "Voyager's remote messages are adequate for many applications, and simple object mobility is often enough to close the gap between two objects communicating on a network. However, as we become familiar with the power of Agents, we may find many ways to Agent-enhance your current and future programs. Developers who prefer the traditional approach to building distributed systems will appreciate the ease of use and universal architecture of Voyager" [7].

PROJECT PROTOTYPE

The project prototype was implemented and tested on a Windows platform. The experiment was conducted using two desktops, one a client running Windows98 and the other a server running Windows 2000 Server.

MS Project2000 was used to create a project and its data was exported to a Microsoft Access database, which was then uploaded to the server. Within the project, there is a 20% completed activity called "Testing" and another activity called "Reviewing" with unknown status. On the server, a stationary agent is created which responds to a friendly Agent arriving at the machine. The stationary agent is programmed with the following functions.

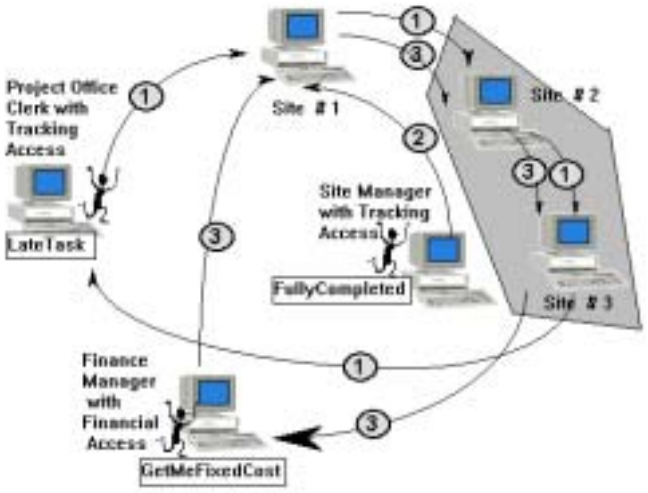
1. A stationary agent at the server is programmed to sense any incoming mobile agents.
2. When an Agent arrives to request submitting of data to the MS Project database, the first thing is to determine whether the visiting Voyager Agent is a friendly one, that is, the incoming Agent supplies the right key. If a right key is supplied, the Stationary Agent will perform the data manipulation on the server on behalf of the Mobile Agent.
3. If the incoming Agent supplies a right key and the key has financial access privileges, the Stationary Agent on request would take the value of the actual *fixed cost* of the Activity "Testing" from the MSProject database and pass it to the Mobile Agent.
4. If the incoming Agent supplies a right key and the key has tracking access privileges, the Stationary Agent on request will obtain the value of *% work completed* from MS Project Database on behalf of the agent. In addition, if the value of the *% work completed* is less than 100%, the Stationary Agent will retrieve from the MS Project database the content of *note object* field (text imputed by site personnel explaining why the Activity "Testing" is not yet completed) and pass it to the Mobile Agent.
5. An Agent named *GetMeFixedCost* was created with financial access privileges, launched from the client and sent to the server.
6. An Agent was created with tracking access and was launched from the client and sent to server. The Agent is named *LateTask.*
7. An Agent with the name *FullyCompleted* was created with tracking access privileges, assigned 100% as the value of the *% work completed* and sent to the server.

COMPUTATION EXPERIENCE

Figure 3 illustrates the various processes that are taking place. Site # 2 and Site #3 are added to provide an extended view of a complete system.

1. Agent *GetMeFixedCost* comes back with the *fixed cost* that is associated with Activity "Testing"
2. Agent *LateTask* comes back with the Note object for the Activity "Testing" from the MS Project database.

Figure 3: Agent-based project management system



3. Agent *FullyCompleted* updated the activity's value of the *% work completed* and was *garbage collected* in the network

This project prototype demonstrates the mobility, functionally and autonomous features of a Mobile Agent. Another important aspect this prototype points out is that the system provides security. Only friendly Agents can interact with the Stationary Agent. The Stationary Agent would prevent any unauthorized access to data that the incoming Agent did not have access to. The Voyager Security module is considered to further enhance the security.

CONCLUSION

The mobility, functionality and autonomous features of Mobile agents make them powerful emerging technology for enhanced enterprise computing.

The ability of Mobile agents to communicate and negotiate with one another mimics how humans interact with each other in the real world. Mobile agents bring back human interaction and negotiation to our cyberworld.

The project prototype has demonstrated that they can be used to facilitate communication and security services within a project management environment. The prototype also highlights the power of mobile agents to provide security and control access.

The prototype provides the foundation to build an Agent-based, fully automatic alert system for project management. Under such a system, a Mobile Agent can be deployed as a virtual inspector to visit or reside on the servers at remote sites.

REFERENCES

[1] Arnaud Sahuguet, About Agents and Database,1997, http://www.cis.upenn.edu/~sahuguet/Agents/Agents_DB.pdf
[2] Anders Falk and Ing-Marie Jonsson, "PAWS: an Agent for WWW-retrieval and filtering", Media Lab, Ericsson Telecom, 1996. http://www.fek.su.se/forskar/program/imorg/dok/1996-08-28/erimedlab/paws/NewPAAM.doc.html
[3] http://actcomm.dartmouth.edu/~rgray/present/mobicom96/idea.html
[4] IBM Agent Lab: http://www.trl.ibm.com/aglets/
[5] Mitsubishi Lab: http://www.meitca.com/HSL/Projects/Concordia
[6] Danny B. Lange and Mitsuru Oshima, Programming and deploying Java Mobile Agents with Aglets, 1998.
[7] Objectspace.com: Voyager ORB Documentation. http://support.objectspace.com/doc/index.html
[8] Don Gilbert and Peter Janca, "IBM Intelligent Agents White Paper", IBM Corporation, May 1997.
[9] Danny B. Lange and Daniel T. Chang, "IBM Aglets Workbench". A White paper, IBM Corporation, September 1996.