A Multi-Paradigm Approach for Mobile Agents

K. P. Tripathi
Bharati Vidyapeeth Deemed University
Institute of Management, Kolhapur (MS), India.
kptripathi06@gmail.com

ABSTRACT
Mobile agent technology offers a computing paradigm in which a program, in the form of a software agent, can suspend its execution on a host computer, transfer itself to another agent-enabled host on the network, and resume execution on the new host. Mobile Agents (MAs) are increasingly gaining attention in both research and the commercial world. In terms of software, mobile agents are active, autonomous software objects containing both computational logic and state information. Agent technology has many forms such as station, mobile and multi-agent, etc. This paper is an attempt to highlights a paradigm approach for mobile agents. Here MAs will be addressed as tools for mobile computing, MAs have been used in applications ranging from network management to automatic software distribution, as well as information management.

KEYWORDS

1. INTRODUCTION
A mobile agent is a computer entity capable of reasoning, use the network infrastructure to run in another remote site, search and gather the results, cooperate with other sites and return to his home site after completing the assigned tasks. As the Internet constantly expands, the amount of available online information expands correspondingly. The issue of how to efficiently find, gather and retrieve this information has led to the research and development of systems and tools that attempt to provide a solution to this problem. These systems and tools are based on the use of MAs’ technology. Mobile agents are processes (e.g. executing programs) that can migrate from one machine of a system to another machine (usually in the same system) in order to satisfy requests made by their clients. Mainly, a mobile agent executes on a machine that hopefully provides the resource or service that it needs to perform its job. If the machine does not contain the needed resource/service, or if the mobile agent requires a different resource/service on another machine, the state information of the mobile agent is saved in a defined manner, then transfer to a machine containing the necessary resource/service is initiated, and the mobile agent resumes execution at the new machine. [1][2]

2. MOBILE AGENT TECHNOLOGY CONCEPT

The mobile agent approach is a relatively new paradigm in the distributed systems environment. MAs have been developed as an extension to mobile code approach (e.g. applet) and could replace the client-server model and its architectures in the near future. Many researchers in this field have extended the mobile-code concept to “mobile object” in which an object (code + data) are moved from one host to another. The mobile agent approach extends this concept further by moving code, data and state (thread) from one host to another as well. MAs run at one location, move with their state to another host, and continue execution at that host. Mobile code and mobile objects are normally moved by an external entity while MAs are usually migrated autonomously. Before the mobile agent paradigm appeared, many approaches have been proposed and developed for communication between client and server such as, Message Passing (MP), Remote Procedure Call (RPC) and Remote Evaluation (REV). In RPC, the client sends data as parameters to a procedure that resides at the server. The procedure will be executed on the server and the results will be sent back to the client. The REV is a different architecture from RPC. Instead of calling a remote procedure at the server side, the procedure itself will be sent from the client to the server to be executed and returns the result. Briefly, in RPC the data is transmitted between the client and server in both directions. In REV, code is sent from the client to the server, and the data is returned. In contrast, a mobile agent is program (encapsulating code, data, and state) sent by a client to a server. However, in a client/server model, a server is a machine that provides some service (or a set of services) and a client (most often another machine) makes requests for those services through a communication channel (e.g. wireless or wired). Communication between the client and the server is usually through message passing. Thus, when a client needs a particular service, it usually sends a request message to the server that contains the needed service as shown in Fig 1. A limitation of the client-server model is that the client is limited to the operations provided at the server. If the client needs a service that a particular server does not provide, it must find a server that can satisfy the request by sending outmode messages to other servers. This clearly is an inefficient use of network bandwidth. In addition, this kind of communication may increase the networks traffic, waste network bandwidth and causes delays of the reply due to server down time or crashes.

The mobile agent appears to tackle significant problems whether in wired or wireless communication such as...
disconnection operations, increased network traffic and others. Moreover, mobile agents can play a major role in the wireless communication after the failure of the Java RMI in increasing the performance over slow wireless links. Once the mobile agent has migrated; the connection between client and server will be disconnected. \[7\][8][12] This saves network bandwidth, especially in a wireless environment. When a mobile agent finishes its job at the server, it will then be ready to reconnect to its host.

![PDA (wireless mode)](image)

**Fig. 1. Client-server paradigm**

### 3. MOBILE AGENT CHARACTERISTICS

A number of characteristics have been identified by. MAs should be autonomous, having the ability to act without direct external interferences. In other words, they have some degree of control over their data and states. Missoula is Interactive in communicating with the environment and other agents. MAs should be adaptive. In other words, they have ability to respond to other agents or their environment. Mobility is the core property in a mobile agent concept whereby the agent has the ability to migrate or transport itself from one node to another within the same environment or from node to another node in a different environment autonomously. Proxy, MAs may act on behalf of someone or for the benefit of some entities (e.g. software systems). In order to act on behalf of others, mobile agents must have at least a minimal degree of autonomy. Proactive, MA should be a goal-oriented entity, and take an initiative in responding to an environment. Intelligent, MAs may have certain degree of intelligence, base on knowledge in order to act efficiency. Coordinative, MAs should be able to perform data transfer activities in sharing with other agents within the given environment. Learning refers to a mobile agent’s ability in gaining information about the current environment, which will help MAs to modify its behavior. Cooperative, Missoula is able to coordinate with other agents to achieve common purpose. Ragged, MAs should be able to deal with errors when encountered and during their (errors) occurrence. Researchers have also attempted to classify the types of the agent based on the above attributes. In many cases, it is not necessary to have an agent with all the above properties. This will be determined by the nature of the task which the agent should achieve. For example, Microsoft’s software agent does not necessarily need to be mobile and cooperative but it must be smart and proactive. “This agent proposed to enhance the user interface of their applications with interactive personalities in the form of animated characters. These characters can move freely within the computer display, speak aloud (and by displaying text onscreen), and even listen for spoken voice commands”. However, as mentioned earlier several types of agent exist such as collaborative agents, interface agents, mobile agents, information agents, reactive agents, hybrid agents and smart agents. These types of agent are built based on their purpose. For this work the mobility factor must exist as the main feature alongside one or many of the previous attributes depends on the need of that characteristic. \[13][18]

![PDA (wireless mode)](image)

**Fig. 2. Mobile agent paradigm**

### 4. WHY USE MOBILE AGENTS TECHNOLOGY?

The nature of mobile agents can lead to many benefits. It can be used in areas like network management, mobile computing, information management, web services, remote software management and others. Mobile agents can increase the performance in these areas by achieving the following tasks: Reduce the network load, it allow users to package and migrate their operations tube carried out locally. This is useful where huge volumes of data are required to be processed; the data will be manipulated locally rather than transferred over the network. In other words, move the computation to the data rather than the data to the computation. Overcome network latency, as can offer great opportunities to respond to real-time systems where delay is not acceptable due to its mobile and adaptive properties. Encapsulates protocols, in distributed systems where many different protocols may be used to communicate among the network nodes, MAs can overcome this problem by moving to remote hosts to establish channels based on proprietary protocols. Execute asynchronously and synchronously, in mobile computing arena where network connection inexpensive (small bandwidth). Some tasks require open connection between mobile devices and fixed networks; this might not be economical and practical. Therefore, MAs can solve this problem by embedding a mobile task into the mobile agent and dispatch the mobile agent to the destination and disconnect the link between the mobile device and the

---

Copy Right © INDIACom-2011 ISSN 0973-7529 ISBN 978-93-80544-00-7
fixed network. However, when the task is finished, MAs can reconnect and return to the mobile device with the results. Heterogeneous, computer networks are heterogeneous in nature from software and hardware perspectives. MAs can work in this kind of environment perfectly due to their independence property. MAs are only depending on their execution environment. Robust and fault-tolerant, the event-based model makes MAs react dynamically to any critical situation within the environment. This makes it easier to build robust distributed systems based on mobile agent technology.

5. MOBILE AGENT REQUIREMENTS

There are three main requirements: the mobile agent programmed (code) itself, mobile agent platforms (execution environment) and mobile interfaces or mobile agent creators. Currently, the traditional programming languages are facing real challenges in implementing the new way of coding (e.g. mobility task) but there are some conventional programming languages which have the potential to be used for this purpose. Developers can build powerful distributed applications by using Java due to its independence platform feature. In addition, mobility tasks can be implemented in Java via Serialization Interface whereby a mobile agent can be serialized throughout a communication channel and deserialized at the other end. However, when developing and implementing the mobile agent code by using any capable language (e.g. Java), MAs should be able to perform the following operations in order to build a robust system: Creation, Cloning, Dispatching or Migration, Retraction, Activation, Deactivation and finally Disposal. Creation is the first period in the mobile agent life cycle. For example, when a system requests service to be done by the mobile agent, the system must create a mobile agent instance in the first place before any work can be done. During, this creation stage the mobile agent will be equipped with the desired parameters in order to achieve its goal. Cloning basically creates a copy of the original mobile Agent object. This operation is used when the need for another agent (with the same features and properties to do the same or other job in conjunction with the original one arises. The dispatching or migrating is another main functioning the mobile agent life cycle. This function is responsible for moving the agent from one node to another within the network environment by specifying the destination address (e.g. URL) to the agent. There are two main types of migration: strong and weak. Strong migration means that the mobile agent can carry itself, its data and its state while weak migration means that mobile agent can carry only itself and its data (e.g. mobile object). The retraction function is done where the agent’s source code requires its agent to be returned to the original host or node. Activation and deactivation are operations done when the mobile agent is required to start or to stop only within certain time of its lifetime. Finally, the dispose operation is done where the agent life comes to the end. Fig. 3 explains the above mobile agent operations as suggested by an Aglet system the second component of the mobile agent requirements is the execution environment. In order to program and run the mobile agent application, mobile agent platform must be implemented and exist. MAs’ special characteristics and behaviors require the host to know how to speak with the incoming agents and with their applications. The host will then provide the execution needs to that arriving agent in order to achieve their goals. Mobile agent platforms should provide the following important requirements: platform independence, authentication, secure execution, dynamic class loading, network connectivity, and resources control. Over the past few years, many mobile agent systems or platforms have been proposed and developed by different industrial and academic organizations. There are a number of well-known academic mobile agent systems such as Agents, Mole, Are and Tacoma as in and industrial mobile agent systems such as Typescript, Aglets, Concordia and Voyager. Developers have designed and implemented the mobile agent based on two main models, one as an extension to the operating system functions and the second as a compound based model. The idea of the first model is to have platforms controlling the mobile agent lifecycle (typically one platform per host). This does not give much flexibility to the mobile agent to operate independently. The idea of the second is to separate the platform from the application. This separation may give a boost to a wide range of developers to start building mobile agent applications seamlessly. The entire list of the mobile agent systems that has been stated above and other systems can be seen at. The third component in the requirements is the interface. This component has two types; the first component can be a text or GUI-based program at the user-end. In this the user may need to set up some agent’s parameters before leaving the client ground to land at the server. [7][11]

![Fig. 3. Mobile agent life cycle (Aglet perspective)](image)

6. TYPES OF MOBILE AGENTS

Mobile Agents can be classified into three main categories:
1. User interface agents
2. Process agents
3. Wrapper agents

User interface agents receive requests from users (humans and other systems), package the requests and send them to appropriate process agents. Receiving and the shipping agents
are examples of user interface agents. User interface agents can be passive or proactive. Proactive form is more suitable for intelligent systems. The passive form is widely used as it produces less ambiguous direction for processing requests. 

Process agents translate the users’ requests that they receive through user interface agents into a series of tasks; determine the best sources to obtain the information required for processing the requests, and make decisions among alternatives. These agents act as surrogates for users in managing and executing the process. Tracking agents is examples of process agents. Wrapper agents represent legacy systems in the mobile-agent-based architecture. Each legacy system is represented by a wrapper agent component (a.k.a. wrapper agent). A wrapper agent represents a legacy system’s interfaces, announces the legacy system’s capabilities in agent compatible ways, acts as the proxy of the legacy system, handles all service requests intended for the system, and manages the system interactions.

7. MOBILE AGENT APPLICATIONS

There are number of applications of mobile agents [16][19]. Few of them are as follows:

1. Resource availability, discovery, monitoring: agents are very useful in monitoring distributed data sources. They are useful in such application as they don’t amplify network load, they can check the resource availability on network, they can monitor the system load etc. also the agents can be dispatched to the location where the user has no normal access to.

2. Information Management Tasks: Information management tasks like searching for information, information filtering, information monitoring. Search agents contain domain knowledge about various information sources. This knowledge includes the types of information available at each source; search agents are useful as they do their work of searching the relevant information from each source. It also deals with the problem of information overload to a user by limiting the information coming to a user.

3. Dynamic software deployment: the deployment and installation of any application in a distributed environment is a complex task, particularly for unmanaged nodes in large scale deployment scenarios. Mobile agents can do this job more effectively as they do not require particular user intervention and its autonomous behavior helps in such situation very well.

Most of the agent systems are developed using:

1. Programming language: Java or C++
2. Communication language: KQML or FIPA ACL
3. Java serialization is used for the migration of the agents.

The following table shows few of the mobile agent system available:

<table>
<thead>
<tr>
<th>Product</th>
<th>Company</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent Building</td>
<td>IBM</td>
<td>C++, Java</td>
</tr>
</tbody>
</table>

8. SECURITY SERVICES FOR MOBILE AGENTS

Security services are important for the protection of mobile agents and server from the attacks. The following is the list of commonly available services for securing agent system.

1. Authentication: before accepting an incoming agent, you want to know who its sender is. In this case, you need authenticate the agent. This process includes the verification of the developer who created the agent or before sending the agent to some host you may wish to know who the host is and what its credentials are:
   a) Authentication of user: the user needs to authenticate himself to a given server. Public-key encryption or a password can be used for this purpose.
   b) Authentication of host: before a server starts to communicate with another server or client, it needs to know with whom it is communicating.
   c) Authentication of code: before executing an incoming agent, the host needs to know who created the agent. Digital signatures are typically used for this purpose.
   d) Authentication of agent: before executing an incoming agent, the server needs to know who is responsible for this agent or who its owner is.

2. Integrity: to trust an agent, you need to make sure that no one has tampered with its code and data. Checking the integrity of the agent is the technique we use to make sure that no manipulation is done with its code and data. Confidentiality: an agent may carry confidential information that should be readable only by intended server or agent. Such information should be kept secret form other servers and agents. [12]

3. Authorization: authorization or access control is the way to specify and enforce an agent’s capability to access information or to use services provided by a server.

4. Non repudiation: an agent or sever cannot deny that a given communication exchange or transaction has taken place.

5. Auditing: auditing service records security-related activities of an agent for later inspection.

9. TECHNICAL OBSTACLES

The Mobile agent paradigm is a promising technology and new method of communication amongst network nodes. Despite a number of successful mobile agent applications, still there are some barriers preventing this technology from spreading out to a wider range of enterprise and individual users. This is due to many reasons such as, lack of standard in both software and hardware products (e.g. programming languages, protocols and
A Multi-Paradigm Approach for Mobile Agents

devices). To overcome this, number of initiatives is underway which may help developers in building their applications based on mobile agent technology as in. The lack of understanding among the researchers and the developers in defining the real concept of mobile agent technology and the tasks that MA should performs is another contributing problems. Furthermore, the current infrastructures not ready to support and integrate with mobile agent technology. Another major concern made by researchers is the security issue, for example, when using MAs whether in E-commerce or M-commerce fields, to act on behalf of their users to handle transactions over the net. Other unresolved issues include privacy, trust and integrity. Privacy is lost since the agent must have access to the user profile, which may contain sensitive information about the user, and may be shared with other agents in the working environment. In addition, this information may be modified during the transaction (by a hacker for example). Hence, methodology is needed whereby the information should be kept and sent without changes to its contents. These issues are trivial but it has to be considered in the mobile agent applications. In other words, MAs need to be protected against hosts, and hosts need to be protected against MAs. Thus, in order to apply the mobile agent approach to these models, a rethink in design and development of the current models of business is necessary to adopt the new paradigm (mobile agent technology).

10. CONCLUSION
Mobile agent technology can offer a new paradigm for communication over heterogeneous network channels. Number of advantages of using mobile agent computing paradigms have been proposed and identified. These advantages include: overcoming network latency, reducing network load, executing asynchronously and autonomously, adapting dynamically, operating in heterogeneous environments, and having robust and fault-tolerant behavior.

11. FUTURE SCOPE
The area of mobile agent technology is still in a somewhat immature state. The traditional host orientation toward security persists, and the focus of protection mechanisms within the mobile agent paradigm remains on protecting the agent platform. However, emphasis is moving toward developing techniques that are directed towards protecting the agent, a much more difficult problem. Fortunately, there are a number of applications for agents where conventional and recently introduced security techniques should prove adequate until further progress can be made.

REFERENCES
[10]. I. Brusic, V. Hassler, W. Lugmayr, Deployment of mobile Agents in the mobile telephone network management on System Sciences, 2000, Proc. 33rd Annual Hawaii Int. Conf. on, Jan 4-7, 2000 Page(s): 3108 -3116
[18]. S.Adman, J. Detain, P. Yalamanchili, a survey of mobile agent systems<URL:http://www.cs.ucsd.edu/classes/sp00/cse221/reports/dat-yal-adn.pdf>