MOBILE AGENT

Mohit Mittal[1]
[1]*Assistant Professor,*
Anand College of Engineering & Management, Kapurthala.

Tarun Bhalla[2]
[2]*Assistant Professor,*
Anand College of Engineering & Management, Kapurthala.

Abstract:*Mobile agent paradigm is an emerging and exciting paradigm for mobile computing applications. Reasons are the inefficiencies associated with more traditionally distributed systems such as client-server applications in terms of latency, bandwidth, vulnerability to network disconnection, mobility etc. Mobile agent technology helps design wide range of adaptive, flexible applications with non-permanent connections by adding mobility to code, machine based intelligence, improved network and database possibilities. This paper provides a comprehensive overview of mobile agents in mobile computing. Mobile agents, their usage, applications, architectures, languages, existing technologies and implementation challenges such as security, portability, scalability, standardization etc. are addressed.*

Keywords:-Introduction, architecture, configuration, advantages, challenges and conclusion.

1. Introduction

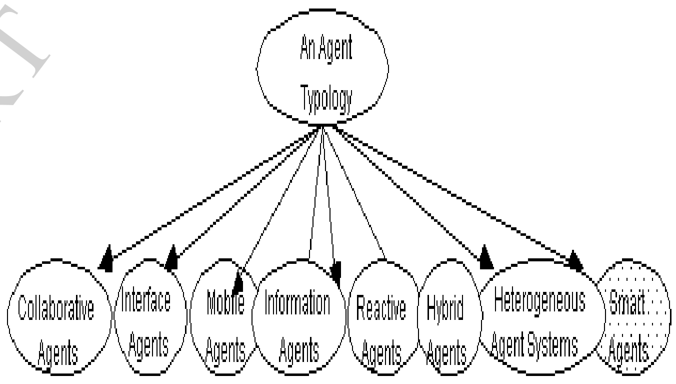
1.1 Introduction:As the trend towards wide-area open networks like the Internet and intranets grows larger, an increasing number of people are expected to use partially connected mobile devices such as laptops, mobile phones, personal digital assistants (PDA) home and business computers etc. to realize the benefits of having their electronic work available from anywhere and at any time. [2]

However, developing distributed applications that can function in a mobile computing environment is difficult for various reasons. First, mobile devices are not permanentlyconnected to the network and often disconnect from it for long periods of time. Second, even when connected, the connection often has low bandwidth, high latency and prone to sudden network failures. Third, every time the mobile device reconnects, the network address assigned to it may change. Any distributed application that need to effectively use internet resources from a mobile platform must deal with the environmental changes and intolerant network conditions.[2]

Mobile agents because of their features such as autonomy, mobility, asynchronous communication, flexible query processing, intelligence, cooperation, reactivity etc. not only support mobile computers and disconnected operations, but provide a convenient, efficient and robust programming paradigm for implementing distributed applications. [2]

This paper will provide a comprehensive overview of mobile agents in mobile computing. In particular, the paper will address what mobile agents are, their usage, advantages, applications, existing technologies, languages, architectures and implementation challenges such as security, portability etc.[2]

Classification of Agents



1.2 Common Characteristics of Agents[2]:

1. Autonomy*:*Agents should be able to perform the majority of their problem solving tasks thedirect intervention of humans or other agents and they should have a degree of control over their own actions and their own internal state.

2. Social ability*:*Agents should be able to interact, with other s/w agents and humans in order to complete their own problem solving.

3. Responsiveness:Agents should perceive their environment and respond in a timely fashion to changes that occur to it.

4. Temporal continuity*:* Agents are continuously running not once only computations or scripts that map a single input to a single output and then terminate.

5. Goal- Orienteers*:*An agent is capable of handling complex high level tasks. The decision how such a task is best split up in smaller sub-tasks and in which order and in which way these sub-tasks should be best performed, should be made by the agent itself.

6. Mobility:It is the ability of an agent to move around an electronic network.

7. Benevolence:It is the assumption that agents do not have conflicting goals and that every agent will therefore always try to do what is asked of it.

1.3 Nwana[2]

Nwana (1996) proposes a typology of agents that identifies other dimensions of classification. Agents may thus be classified according to:-

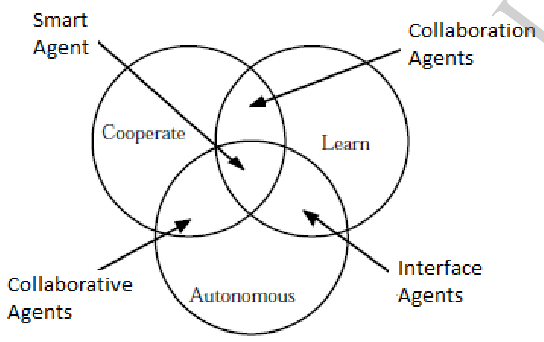
• Mobility, as static or mobile

• Presence of a symbolic reasoning model, as deliberative or reactive. Exhibition of ideal and primary attributes, such as autonomy, cooperation, and learning. From these characteristics, Nwana derives four agent types: collaborative, collaborative learning, interface, and smart.

• Roles, as information or Internet

• Hybrid philosophies, which combine two or more approaches in a single agent

After developing this typology, Nwana goes on to describe ongoing research in seven categories: collaborative agents, interface agents, mobile agents, information/Internet agents, reactive agents, hybrid agents, and smart agents. [2]



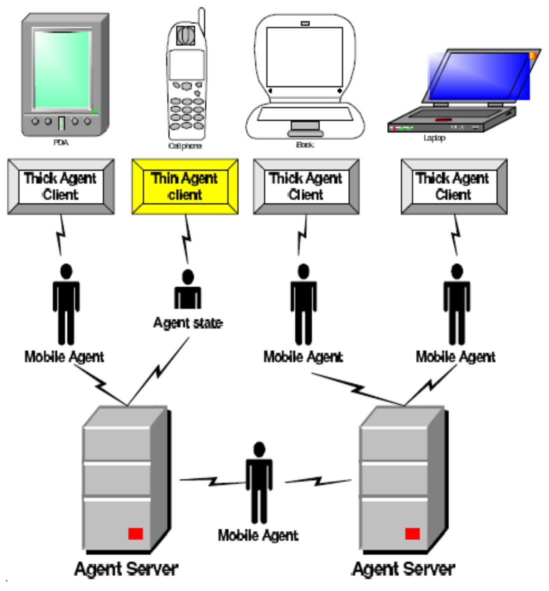
“Figure 1: Typology based on Nwana’s (Nwana 1996) primary attribute dimension.” [2]

2. Mobile Agent

2.1 What are Mobile Agents?

A mobile agent is a program that is autonomous and can move through a heterogeneous network under its own control, migrating from a host to host and interacting with other agents. It decides when and where to migrate. It can execute at any point or suspend its execution, move to another host and continue its execution on that host. Mobile agents have certain features such as autonomy, mobility, goal driven, temporarily continuous, intelligence, cooperation, learning, reactivity etc. Because of these features, they are well adapted to the domain of mobile computing. [5]
For instance, a mobile agent can move from a PDA to Internet to collect interested information for the user. Since it is on the network and does not have to transfer the multiple requests/responses across the low bandwidth connection, it can access necessary resources efficiently. Further, sudden connection losses will not affect the agent since it is not in continuous contact with the mobile device. An agent can perform its tasks even if the mobile device is disconnected from the network. Upon the reconnection of mobile device to the network, agent will return to it with results. Alternatively, a network application can dispatch a mobile agent onto the mobile device. The agent acting on behalf of the application interacts with the user regardless of whether or not mobile device is connected. [5]

Mobile agents simplify the development, testing and implementation of distributed applications because of their ability to hide the communication channels and show the computation logic. They can distribute and redistribute them throughout the network and can act as either clients or servers depending on their goals. They can also increase the scalability of the applications because of their ability to move work to an appropriate location. [7]

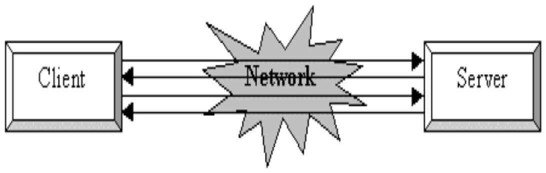


“Figure 2: A heterogeneous mobile agent system.” [7]

A software agent is a mobile software agent if it is able to migrate from host to host to work in a heterogeneous network environment. This means we must also consider the software environment in which mobile agents exist. This is called the mobile agent environment, which is a software system distributed over a network of heterogeneous computers and its primary task is to provide an environment in which mobile agents can run. Note that not only an agent transports itself, but also its state. When it reaches the new host, the agent should be able to perform appropriately in the new environment.

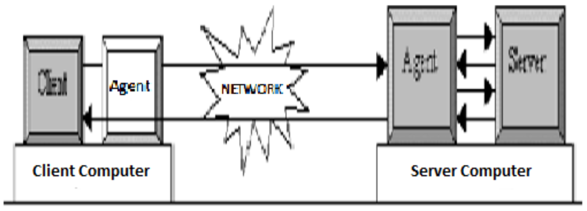
2.2 A New Paradigm for Distributed Computing:

The central principle of today's distributed programming is remote procedure calling (RPC). The RPC approach, which was conceived in the 1970s, views computer-to-computer communication as enabling one computer to call a procedure in another. In RPC, all messages go through the network; each either requests or acknowledges a procedure's actions. This approach, however, has its own limitations. Most notably, all interactions between the client and server must go through the network.[1]



"Figure 3: RCP-based Client/Server Computing Paradigm." [1]

Another approach that is forming a new paradigm for distributed computing is one that employs mobile agents. Initially this approach was known as Remote Programming. The Remote Programming approach views computer-to computer communication as one computer not only to call procedures in another, but also to supply the procedures to be performed. Each message that goes through the network comprises a procedure that the receiving computer is to perform and data that are its arguments. The procedure and its state are termed a mobile agent as they represent the sending computer even while they are in the receiving computer.[1]



"Figure 4: Mobile agents-based computing paradigm." [1]

This approach is attractive since the reliability of the network is not crucial for the following reasons:

- Mobile agents do not consume much network bandwidth. They only consume bandwidth when they move.
- They continue to execute after they move, even if they lose network connectivity with their creators.

Therefore, if a client requires extensive communications with a particular server somewhere on the network, then implementing such a system using mobile agents is attractive. This is due to the fact that an agent can move closer to the remote server, reducing the network traffic, performs all tasks and comes back. During that period the client machine doe not have to be switched on. It will have to be switched on only when it is time to welcome back the agent. At this point, someone may say that this is exactly what process migrationis all about and this has been done in the 60's. That someone would be absolutely right. However, mobile agents are different in the sense that they exhibit the characteristics of an agent.[1]

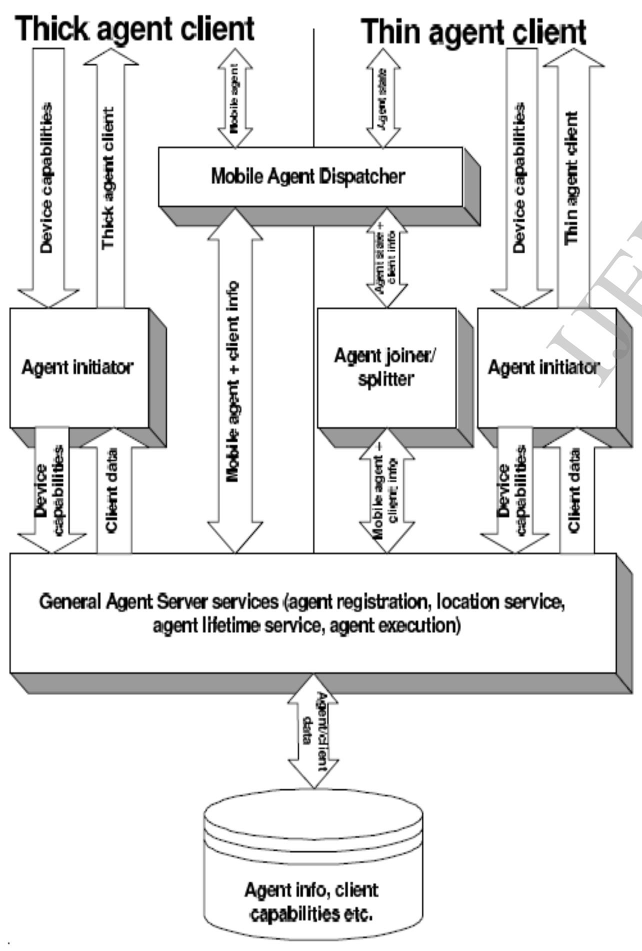
3. ARCHITECTURES

3.1 Single Agent System: As the name suggests, a single agent system employs only one agent. This type of a system can become more complex than multiple agent systems, if the job assigned to an agent is complex. For instance, in multiple agent systems, control can be distributed and a single agent does not have to complete the given job. On the other hand, in a single agent system, an agent can be unnecessarily burdened if it has to complete all the tasks. Single agent architecture is better suited for those systems whose domain requires centralized control. [3]

3.2 Multi-agent systems: Deployment of multi-agent architecture benefits a system in many ways. In these systems, various independent agent handle separate tasks. Provision of parallel processing capabilities and the presence of redundant agents increase system's operations and robustness. Even if one or more agents fail, a system can still work, since agents share responsibilities. Another advantage of multi-agent system is scalability. It is easier to add new agents to a multi-agent system than to single agent system Programmers can easily decompose a system into multiple tasks and assign these tasks to different agents. Hence from the programmer's perspective, it is relatively easy to program in these systems. Multi-agent system architecture suits those systems such as ecommerce, where criterion changes across agents or over time. Multi-agent systems can be of different forms. [3]

3.3 High-level architecture: In Logical view architecture of our heterogeneous mobile agent system. A central part of the architecture is the agent system repository where agent information and agents can be stored along with client information such as client device capabilities. We have defined a thin agent client that only deal with the agent state, where the code of the agent will be stored on the agent server. The thin agent client contains code for transmitting and receiving agent data and a simple GUI for manipulating this data. For devices with sufficient memory and CPU, mobile agents will run locally on the device. Above the repository the general agent server services are located that controls the essentials agent services such as registration of agents, locating agents, management of agent lifetimes (initiate agents, clone agents, kill agents) etc. The rest of the architecture is split into two main parts; one for thick agent clients, and one for thin agent clients. The thick agent client is able to execute mobile agents locally, while the thin agent client only is able to manipulate the agent data (state). There is only one part that distinguishes the architecture of the thick and thin agent client; the Agent joiner/splitter. The Agent joiner/splitter makes it possible to split the data and code of the mobile agents before dispatching the agent to the thin client. This makes it possible to access mobile agents on less capable devices. [7]

3.3.1 The Agent Initiator: When a client connects to the agent server for the first time, it must state the client device capabilities in terms of CPU speed, execution memory, storage, and Java Virtual Machine edition and version. Although a client device has enough memory and fast enough CPU, there could e.g. be no Java virtual machine available that support serialization for the device. This is the reason that Java Virtual machine is also a part of the client capabilities. The client capabilities described in XML consist of five main parts: Device, CPU, Memory, Java and Download. The Device part is used to describe the device in terms of name and operating system running. The next two parts CPU and Memory describe the device capabilities in terms of processing power and executing memory and storage space. The Java part describes the available Java environments on the device in terms of edition and versions. The last part, Download, makes it possible to specify how the agent client should be downloaded to the device. [7]



"Figure 5: A Logical view of mobile agent architecture." [7]

When a mobile agent client is initialized the first time, it follows the following process:

1. Install agent initiator -The agent initiator is a simple Java program that must be installed on the mobile device before installing the agent client. This program is used for extracting device capabilities on the client. The devices capabilities that cannot automatically be detected must be entered manually by the user. In addition, the agent initiator contains software for communicating with an agent server.

2. Configure the agent initiator -The user should look through the device capabilities detected, and add missing, or change faulty information.

3. Initiate the agent initiator -The agent initiator sends the device capability information to the agent server.

4. Install agent client -Based on the device capabilities, a suited agent client will be sent to the device and then installed. The agent client can be sent directly using the agent initiator, or indirectly using transport channels such as email or HTTP-download.

3. 3.2 The Agent Joiner/Splitter: For clients that cannot run the mobile agents locally (because of client capabilities), we have designed an agent joiner/splitter for allowing these clients to access the agents anyway. This is done by separating the state (or data) of the agent with the code of the agent. This means that the thin agent client only receives the data part of the mobile agent while the code part resides on agent server. In addition, the client on the mobile device has a graphical user interface (GUI) making it possible to instruct the agents. When the user decides to send an agent from the mobile device, the state of the agent is transmitted to an agent server. This means that both the code and the state of the agent are joined on the agent server. [7]

The agent's lifecycle can be split into respectively client and server states.

<u>The client states are:-</u>

1. Initialize -The state of a new agent is populated by options given by the client.

2. Migrate (C/S) -The state of the agent migrates from the client to the server.

8. Receive -The state of the agent is transmitted from the server to the client.

<u>The server states are:</u>

3. Initialize -The agent performs one-time setup activities by building the initial data structures of the agent. In this state the code and data of the agent are joined.

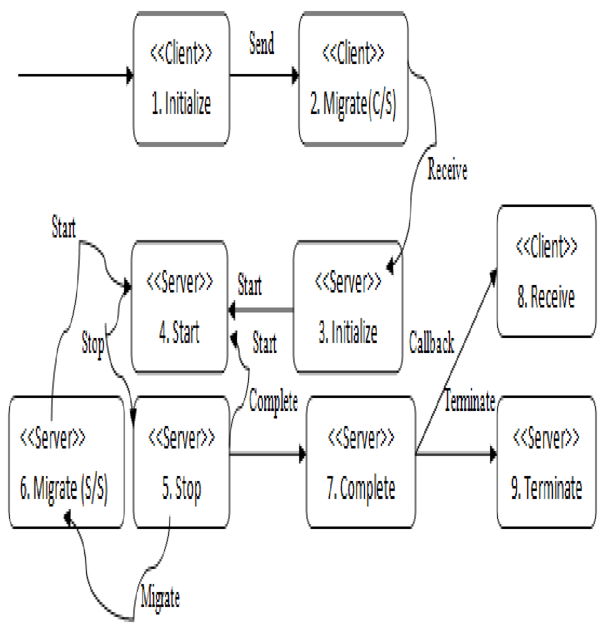
4. Start -The agent is running (and performs its tasks).

5. Stop -The agent stops the execution and saves the state.

6. Migrate (S/S) -The agent migrates from one server to another (both code and state).

7. Complete -The agent is prepared for call-back by the client.

9. Terminate -The agent is removed from the agent server.



"Figure 6: State view of mobile agent architecture." [7]

4. CONFIGRATION[7]

4.1 Configurations of mobile agents: The tasks that an agent can accomplish depend on how the agent is configured in the first place. Some of the configurations of an agent include:

1. Simple reflex agent - In this case, agent does not have memory. It simply perceives the current situation, finds a rule that matches the situation and executes it.

2. A reflex agent with internal state - Agent perceives the current situation and based on its perception and stored internal state finds a rule that matches the situation and executes it.

3. An agent with explicit goals -Agent selects those actions that help it achieve goals. Compared to simple reflex agent, this agent is more flexible.

4. A utility-based agent – While achieving its goal, this agent also maximizes some performance measure.

4.2Languages[7] :A number of new and open technologies such as distributed objects, Java and extensible markup language (XML) can be used to implement mobile agents. One of the popular languages for implementing mobile agents is Java. In fact, Concordia, Odyssey and Voyager are all implemented in Java. Some of the features of Java such as multi-platform support object serialization, networking support which includes sockets, URL communication, distributed object protocol called remote method invocation (RMI) etc. makes it extremely suitable for mobile agent technology. Applets can be used to launch and receive mobile agents. XML documents can be used to publish anything on the internet. It is one of the most popular accepted foundation layers on which to build. It can be used to encode data with meaningful structure and semantics that any agent with proper authorization can easily access, understand and interpret.

Notable Mobile Agent Systems: Java applets are the most known examples of mobile code. However, java applets are not really mobile agents because they only migrate once and that too upon the user's request before they get executed. Mobile agents are much more powerful than that as they can migrate and execute at their own will. True mobile agent systems include Tele-script, IBMAg lets, Odyssey, Concordia, Voyager etc, Agent Tcl, Tacoma, Mobile Service Agents, Ara etc.[7]

Currently, Tele-script agents are used in managing networks, active e-mail, electronic commerce, business process management etc. A Tele-script agent in network management may bring a software upgrade onto a user machine, install the code without any intervention from user and disappear. A tele-script agent in e-commerce may leave the user's mobile device, roam on internet, acting on user's behalf, collect the material and return to the user's mobile device with the results.[7]

4.3 Advantages of Mobile Agent[7]:Mobile agents have several advantages over traditional client/server model. Some of these include:-

1. Reduction of network traffic -In distributed systems, performing a simple job involves multiple interactions, resulting in increased network traffic. In mobile agent paradigm, the objective is to move the computation to the data rather than the data to the computation, thus consuming fewer network resources and thereby increasing efficiency.

2. Overcome network latency -Management of critical real time systems with substantial size networks creates latencies, which are unacceptable. Deployment of mobile agents will overcome this problem since these agents can execute locally upon the central controller's directions.

3. Encapsulation of protocols -Often, as new communication protocols which improve efficiency and security emerge, businesses need to upgrade their protocols; otherwise they turn into a legacy problem. This whole process is cumbersome. Mobile agents will resolve this problem, as they can migrate to a remote host and establish channels based on proprietary protocols.

4. Asynchronous & autonomous execution - For jobs that require a continuous open connection, deployment of mobile agents will result in cost savings. Mobile agents with embedded tasks can be dispatched into the network where they operate independently and asynchronously.

5. Dynamic adoption -Mobile agents can perceive the surrounding environment and can act dynamically.

6. Seamless system integration -Both from the perspectives of hardware and software, networking computing are heterogeneous. Mobile agents can provide seamless system integration since they are dependent only on those environments in which they execute.

7. Robust and fault-tolerant -Since mobile agents can act and react dynamically in presence of unfavorable conditions, it is easy to build a robust and fault-tolerant distributed system with mobile agents.

4.4 Challenges in implementing Mobile Agents[7]:

1. Security -Security is one of the major issues that need to be considered in the implementation of mobile agents. One of the properties of a mobile agent is that it can roam on a heterogeneous network and can execute its code on a foreign server system. This property also makes it vulnerable to malicious attacks from other agents and servers.

2. Protection of hosts from malicious agents - Since mobile agent system is an open system, it can easily be attacked. Attacks can be in the form of leakage, tampering, stealing of resources and vandalism. The host executes mobile agent's code. Therefore a mobile agent has access to the resources of a host. This access gives the mobile agent the power to attack other local agents, generate viruses, worms or deny the services to other agents etc. Several researchers in this area have provided only a partial solution. Some of the solutions provided include authentication, verification, authorization, digital signatures etc.

3. Protection of Agents from malicious hosts -It is much more difficult to protect an agent from malicious hosts than to protect a host from malicious agents. Since hosts execute the mobile agents, they can see the agent's code and data. Hence it makes easier for the host to tamper the agent's code or terminate them. Some of the ways to protect the agents from the host include limiting the amount of confidential data supplied to the agent, routing them in secure zones, use of cryptographic algorithm or tamper proof hardware, enforcing good host behavior etc.

4. Portability -Mobile agent technology allows program to migrate freely from one host to another among heterogeneous machines. The code compiled into some sort of platform independent representation, such as java byte codes is either executed inside an interpreter or compiled into native code, when it arrives at the target machine. This code if not, portable across mobile-code systems, will limit the usage of mobile agents.

5. Performance and scalability -For portability and security reasons, most current agents are written in a slow interpreted language. As a result, these agents save network latency and bandwidth at the expense of high loads on service machines. For instance, while mobile agents execute faster in network inconsistencies, the opposite is not true. When there are no networks disconnections, mobile agents usually take much longer time to perform a task than their traditional counterparts, since the time that is saved from avoiding intermediate traffic is much less than the time involved in slow execution and migration overhead. With the surfacing of just-in-time compilation code such as java, software fault isolation etc. Some of the problems with the slower execution of mobile agent code are alleviated. However, several other problems involving migration overhead, slow execution etc. still remain to be addressed.

4.5Applications of Mobile Agents[5]:

Several applications benefit from the use of mobile agent technology. Some of these include:-

1. Electronic Commerce -Many commercial transactions require access to resources in real time. The ability of a mobile agent to personify their creator's intentions and to act and negotiate on behalf of them makes it well suited for electronic commerce.

2. Personal Assistance -An agent can act as a personal assistant to the user and perform tasks for user on a remote host regardless of whether or not user is connected to the network. For instance, to schedule a meeting, a user can dispatch a mobile agent onto the network to interact with agents belonging to other users. The agent can negotiate with other agents the convenient time for all of the users and can schedule a meeting.

3. Secure brokering -Mobile agent technology is an attractive solution to brokering, particularly in the context of untrustworthy collaborators. In such a situation, the interested parties can let the agents meet and negotiate at a mutually agreed impartial secure host and form alliance.

4. Distributed information retrieval -Mobile agent technology provides efficient information retrieval. When dealing with large amounts of data, rather than moving all the data to search engine to create search indexes, user can simply dispatch mobile agents to remote sources to create those indexes locally and to ship them back later to its origin.

5. Telecommunication networks services -Mobile agents provide an effective and flexible solution to the management of advanced telecommunication services by providing dynamic network reconfiguration and user customization

6. Workflow applications and groupware - Mobile agents because of their features such as autonomy, mobility etc., provide autonomy to the workflow item and support the information flow between co-workers.

7. Monitoring and notification -As a local representative for remote services, an agent can perform tasks on behalf a user irrespective of whether or not user is connected to the network. For instance, a user can dispatch a mobile agent to the internet to monitor the stock prices and to notify him/her only when certain thresholds are reached.

8. Information dissemination -Mobile Agents, exemplifying the Internet Push Model can distribute information such as news and software updates for vendors. The agents carry the upgrades and installation procedures directly to the user's personal computer and without any user's intervention updates and manage the software on the computer.

9. Parallel processing **-**Mobile Agent technology can provide administration for parallel processing tasks. If a computation requires large amount of processing power so as it to distribute it among multiple processors, mobile agents can help get the processes out there.

CONCLUSION

Mobile agent paradigm is considered to be an effective paradigm in the area of distributed programming. Although current trends in internet technologies indicate widespread usage of mobile agents in near future, several technical and non-technical challenges such as security still remain and need to be resolved. However, once these challenges are met, and some internet sites accept mobile agents, the usage of mobile agents will spreads widely, thereby revolutionizing the mobile computing.

REFERENCES

[1] D. C. Smith, A. Cypher and J. Spohrer (1994) "Programming Agents without a programming language" Communications of the ACM 37 (7) pp 55-67.

[2] J. White. "Mobile Agents".In J. M. Bradshaw (editor), Software Agents. AAAI Press/MIT Press, 1997, pp. 437-472.

[3] P. Morreale. "Agents on the move". In IEEE Spectrum, April 1998, pp. 34-41

[4] C. Guilfoyle and E. Warner (1994) "Intelligent Agents: The New Revolution in Software" Ovum

[5] DejanMilojicic, "Mobile agent applications" Hewlett Packard Laboratories.

[6] Robles Sergi,"Mobile agents systems and trust", bellaterra, july 2002.

[7] Alf Inge Wan,Carl-Fredrik Sørensen and Eva Indal "A Mobile Agent Architecture for Heterogeneous Devices" Dept. of Computer and Information Science, Norwegian University of Science and Technology, N-7491 Trondheim, Norway.

Mohit Mittal received his B.Tech and M.Tech degree in Computer Science from Guru Nanak Dev University, in 2011. He is working as Assistant Professor in Anand college of Engineering and Management, Kapurthala. His research areas include image processing, computer networks and Network Security.

Tarun Bhalla received his B.Tech degree in Computer Science from Punjab Technical University. He is currently working as Assistant Professor in Anand College of Engineering and Management, Kapurthala. His research interest area includes Database, Network Security Mobile Computing and adhoc network.