Mobile Agents

Intelligent Assistants on the Internet

Parineeth M Reddy

Parineeth M Reddy is a computer science BE graduate currently doing a project on peer-to-peer computing at SERC, Indian Institute of Science. His research interests include networking and distributed computing.

A software agent is an intelligent program that acts as a user's personal assistant. Software agents endowed with the property of mobility are called mobile agents. Mobile agents perform a user's task by migrating and executing on several hosts connected to the network.

Introduction

An agent is defined as "a person whose job is to act for, or manage the affairs of, other people". In the context of computers, software agents refer to programs that perform certain tasks on behalf of the user. Imagine that you want to go on a trip to a new holiday destination. You contact your travel agent program and describe your preferences and your constraints (such as how much money you are willing to spend, when you want to travel, etc.). The travel agent program suggests where you can spend your holidays after consulting several information sources such as tourist guides and flight schedules and verifying the availability of airline tickets and hotel rooms. When you confirm your destination, the program books the flight tickets and reserves the hotel rooms for you. Thus the software agent acts as your personal assistant.

Software agents have the following properties, which distinguish them from other programs:

Intelligence: Software agents employ techniques from the field of artificial intelligence, which empower them with a fair degree of intelligence and common sense. For example, the travel agent program should realize that people generally do not prefer travelling by flights that depart or arrive at the airport late in the night and the agent should avoid booking tickets on such flights. The travel agent program should be smart enough to

Keywords
Software agent, static agent, mobile agent.

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

bargain and arrange the trip so that the overall expenditure for the trip is as low as possible without compromising on the user's preferences.

Autonomy: The agents themselves decide the sequence of actions to be performed to achieve the user's task. This autonomy enables agents to operate without requiring human intervention. Once the specifications are given to the travel agent, it should be able to proceed on its own to arrange the trip for the user without requiring the user to constantly monitor the agent.

Responsiveness: Agents perceive their environment (which may be the Internet, a collection of other agents, etc.) and respond in a timely fashion to changes that occur in it. At the same time, agents should not simply act in response to their environment, they should be able to exhibit opportunistic, goal-oriented behaviour and take the initiative when appropriate.

Communicative Ability: Software agents should provide a user-friendly interface so that the user can easily interact with the agent. Agents are social entities and often communicate and collaborate with one another in order to complete their tasks. For example, the travel agent program of one user must be able to communicate with other travel agents to find out about hotels which customers disliked and avoid such hotels.

Adaptability: Agents learn about the user's behavior and adapt themselves to suit the user. Consider a search agent that retrieves information for the user from the Internet. The search agent should be able to learn about the type of information the user is interested in and adapt itself to deliver only the relevant information to the user.

Mobile agents migrate from one computer to another in the network and execute on several machines.

Software agents can be classified as *static agents* and *mobile agents*. Static agents achieve their goal by executing on a single machine. On the other hand, mobile agents migrate from one computer to another in the network and execute on several machines. Mobility increases the functionality of the mobile

agent and allows the mobile agent to perform tasks beyond the scope of static agents. For example, a travel agent program may visit several hosts such as an airline reservation system, a hotel resources host, etc. in order to achieve its function. This eliminates the need to have all the relevant databases on one host. This article gives the architecture of mobile agents and tradeoffs involved in using them.

A mobile agent consists of the program code and the program execution state.

Working of Mobile Agents

A mobile agent consists of the program code and the program execution state (the current values of variables, next instruction to be executed, etc.). Initially a mobile agent resides on a computer called the home machine. The agent is then dispatched to execute on a remote computer called a mobile agent host (a mobile agent host is also called mobile agent platform or mobile agent server). When a mobile agent is dispatched the entire code of the mobile agent and the execution state of the mobile agent is transferred to the host.

The host provides a suitable execution environment for the mobile agent to execute. The mobile agent uses resources (CPU, memory, etc.) of the host to perform its task. After completing its task on the host, the mobile agent migrates to another computer. Since the state information is also transferred to the host, mobile agents can resume the execution of the code from where they left off in the previous host instead of having to restart execution from the beginning. This continues until the mobile agent returns to its home machine after completing execution on the last machine in its itinerary.

The life cycle of a mobile agent

1. The mobile agent is *created* in the Home Machine.
2. The mobile agent is *dispatched* to the Host Machine A for execution.
3. The agent executes on Host Machine A.
4. After execution the agent is *cloned* to create two copies. One

The mobile agent uses resources (CPU, memory etc.) of the host to perform its task.

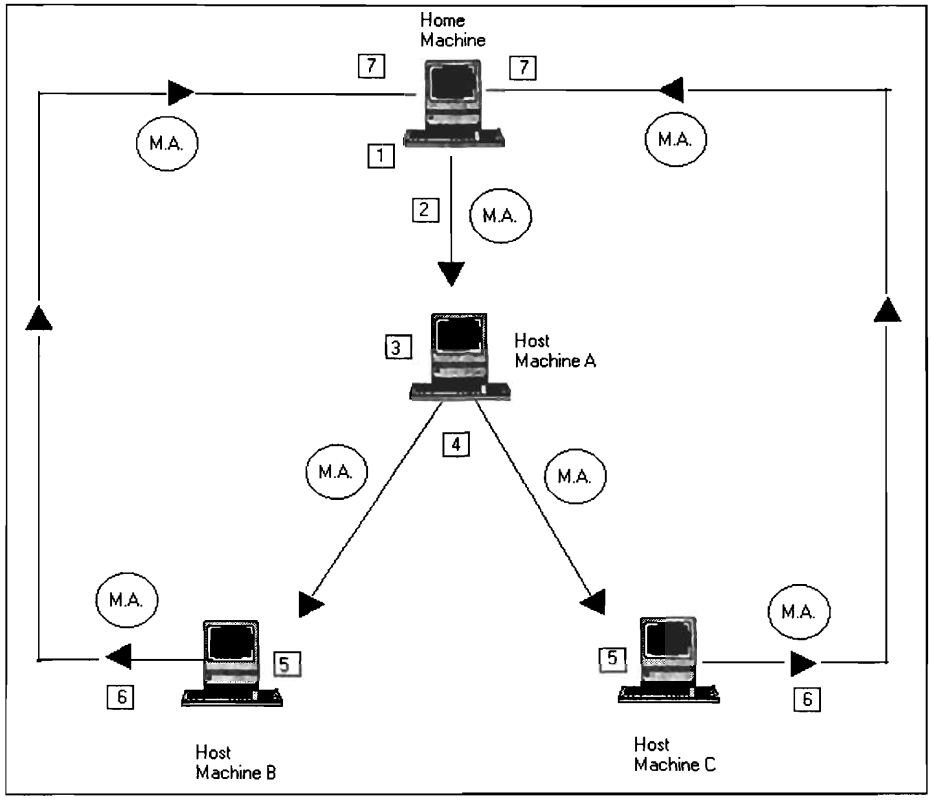

Figure 1. The life cycle of a mobile agent.

copy is dispatched to Host Machine B and the other is dispatched to Host Machine C.

5. The cloned copies execute on their respective hosts.

6. After execution, Host Machine B and C send the mobile agent received by them back to the Home Machine.

7. The Home Machine *retracts* the agents and the data brought by the agents is analyzed. The agents are then *disposed*.

From this we observe that a mobile agent experiences the following events in its life cycle:

Creation: a brand new agent is born and its state is initialized.

Dispatch: an agent travels to a new host.

Cloning: a twin agent is born and the current state of the original is duplicated in the clone.

Deactivation: an agent is put to sleep and its state is stored on a disk of the host.

Activation: a deactivated agent is brought back to life and its state is restored from disk.

Retraction: an agent is brought back from a remote host along with its state to the home machine.

Disposal: an agent is terminated and its state is lost forever.

Mobile agents can learn from experiences and adapt themselves to the environment.

Properties of Mobile Agents

Mobile agents have the following unique properties.

Adaptive Learning: Mobile agents can learn from experiences and adapt themselves to the environment. They can monitor traffic in large networks and learn about the trouble spots in the network. Based on the experiences of the agent in the network the agent can choose better routes to reach the next host.

Autonomy: Mobile agents can take some decisions on its own. For example, mobile agents are free to choose the next host and when to migrate to the next host. These decisions are transparent to the user and the decisions are taken in the interest of the user.

Mobility: Mobile agents have the ability to move from one host to another in the network.

Advantages of Mobile Agents

Reduction in Network Load

The interactions in a distributed system are often achieved using communication protocols. These protocols involve transfer of large volumes of data stored at remote hosts over the network to a central processing site resulting in high network traffic. An alternative to using communication protocols is the use of mobile agents. Mobile agents are dispatched to the remote hosts containing the data. The agents perform the computations at the remote hosts and return back with the results. Since computa-

An alternative to using communication protocols is the use of mobile agents.

Mobile agents operate asynchronously. Once a mobile agent is dispatched from the home machine, the home machine can disconnect from the network.

tions are moved to the data storage location instead of moving data to the computing location, network load is reduced.

Overcome Network Latency

Consider a manufacturing plant in which many critical real time systems are controlled through a network. Controlling many systems through a network involves significant delays, which are not acceptable for critical real time systems. To overcome this problem, mobile agents can be directly dispatched from the central controller in the manufacturing plant to the real time systems. The agents act locally and directly execute the controller's directions.

Protocol Encapsulation

Protocols enable components of a distributed system to communicate and co-ordinate their activities. However, protocols evolve over a period of time and new features such as better security may be introduced in the protocol. It is a cumbersome task to upgrade the protocol code at all locations in the distributed system. Mobile agents offer a solution to this problem. The mobile agent code can encapsulate the protocol. When a protocol is upgraded, only the mobile agent has to be altered.

Asynchronous and Autonomous Execution

Mobile agents operate asynchronously. Once a mobile agent is dispatched from the home machine, the home machine can disconnect from the network. The mobile agent executes autonomously without the intervention of the home machine. The home machine can reconnect at a later time and collect the agent.

Mobile agents react dynamically and autonomously to the changes in their environment, which makes them robust, and fault tolerant.

Fault Tolerance

Mobile agents react dynamically and autonomously to the changes in their environment, which makes them robust, and fault tolerant. They have the ability to distribute themselves in the network in such a way as to maintain the optimal configuration for solving the particular problem. If a host is being shut down, all agents executing on that machine will be warned and

given time to dispatch themselves and continue their operation on another host in the network.

Disadvantages of Mobile Agents

The main drawback of mobile agents is the security risk involved in using mobile agents. Security risks in a mobile computing environment are two fold.

The main drawback of mobile agents is the security risk involved in using mobile agents.

Firstly a malicious mobile agent can damage a host. For example a virus can be disguised as a mobile agent and distributed in the network causing damage to the host machines that execute the agent.

On the other hand a malicious host can tamper with the functioning of the mobile agent. Most experts suggest that this risk is far more difficult to deal with. To illustrate this scenario, consider a mobile agent that visits the servers of several airlines to buy a ticket for the lowest price. A malicious airline server can try to obtain sensitive price information from the mobile agent (such as the prices quoted at the servers previously visited by the mobile agent). The malicious server may tamper with the mobile agent and increase the prices quoted by other airlines thereby giving it an unfair advantage. Some servers may even try to steal the credit card number from the mobile agent.

The defense mechanisms suggested try to prevent malicious actions in the first place. These include safe programming languages that prevent mobile agents from performing malicious actions on the hosts. If a malicious action does occur then the defense mechanisms detect it as soon as possible and take remedial action. One of the schemes proposed is to introduce a tracing mechanism that records the execution of the mobile agent at each host. When the agent is dispatched to the next host, the trace is also sent. Using this trace, malicious actions can be detected and the malicious host can be identified. However, in spite of significant development in the field of cryptography there still exist many security issues that need to be addressed in mobile agents.

A virus can be disguised as a mobile agent and distributed in the network causing damage to the host machines that execute the agent.

Suggested Reading

[1] Michael Knapik and Jay Johnson, *Developing Intelligent Agents for Distributed Systems: Exploring Architectures, Techniques, and Applications*, McGraw-Hill Professional Publishing, 1997.

[2] Joseph P Bigus, Jennifer Bigus and Joe Bigus, *Constructing Intelligent Agents Using Java: Professional Developer's Guide*, 2nd Edition, John Wiley & Sons, 2001.

[3] William R Cockayne and Michael Zyda, *Mobile Agents*, Prentice Hall, 1998.

[4] Danny B Lange and Mitsuru Oshima, *Programming and Deploying Java (TM) Mobile Agents with Aglets(TM)*, Addison-Wesley Pub Co., 1998.

Web Sites

1. Introduction to Agents
agents.umbc.edu/introduction/
2. IBM Aglets developed at IBM, Tokyo Research Laboratory.
www.trl.ibm.com/aglets index_e.htm
3. D' Agent developed at University of Dartmouth
agent.cs.dartmouth.edu/
4. Mole developed at the University of Stuttgart
mole.informatik.uni-stuttgart.de/

Applications of Mobile Agents

Although no universally used application (normally called killer application) has been developed for them, mobile agents are suitable for the following applications.

Parallel Computing: Solving a complex problem on a single computer takes a lot of time. To overcome this, mobile agents can be written to solve the problem. These agents migrate to computers on the network, which have the required resources and use them to solve the problem in parallel thereby reducing the time required to solve the problem.

Data Collection: Consider a case wherein, data from many clients has to be processed. In the traditional client-server model, all the clients have to send their data to the server for processing resulting in high network traffic. Instead mobile agents can be sent to the individual clients to process data and send back results to the server, thereby reducing the network load.

E-commerce: Mobile agents can travel to different trading sites and help to locate the most appropriate deal, negotiate the deal and even finalize business transactions on behalf of their owners. A mobile agent can be programmed to bid in an online auction on behalf of the user. The user himself need not be online during the auction.

Mobile Computing: Wireless Internet access is likely to stay slow and expensive. Power consumption of wireless devices and high connection fee deter users from staying online while some complicated query is handled on behalf of the user. Users can dispatch a mobile agent, which embodies their queries, and log off, and the results can be picked up at a later time.

Conclusions

Distributed computing involving several computers in a network can be achieved using message passing or remote procedure calls (RPC). The recently developed mobile agent technology adds a new dimension to distributed computing. Experts

suggest that mobile agents will be used in many Internet applications in the years to come. However there still exist many technical hurdles that need to be tackled, the most important of them being security. Only when security issues are properly addressed, will the mobile agent technology be widely accepted.

Address for Correspondence
Parineeth M Reddy
3529, I Cross, 13-H Main
HAL 2nd Stage,
Bangalore 560 008, India.
Email: parineeth@yahoo.com

Refresher Course in Mathematics

(Coding Theory, Cryptography and Discrete Mathematics)
Department of Mathematics, Panjab University, Chandigarh 160 014
December 2-14, 2002

Organised by Panjab University, Chandigarh
sponspored by Indian Academy of Sciences, Bangalore

Applications are invited from University/College teachers and Research Fellows for participation in the Refresher Course on coding theory, cryptography and discrete mathematics. About 30 persons will be selected to attend.

Outline of the Course:

1. Error-correcting codes, linear codes, generator matrix, parity check matrix, dual codes, syndrome decoding, Hamming codes, perfect codes, BCH codes, cyclic codes, idempotents of cyclic codes, quadratic residue codes, Golay codes.

2. Simple cryptosystems, block ciphers, enciphering matrices, elementary cryptoanalysis, public key cryptography, RSA, discrete log and knapsack cryptosystems, primality testing and factoring, introduction to elliptic curve cryptography.

3. Pigeon hole principle, basic counting arguments, inclusion exclusion principle, recurrence relations, generating functions, elementary graph theory, Eulerian paths and cycles, trees, planar graphs, chromatic number.

There will be formal lectures, general talks, problem sessions and discussions. Pre-requisites: Knowledge of elementary Number Theory and first course in Algebra.

Selected participants will be provided local hospitality and round trip train fare (first class or three-tier A/C) to and fro by the shortest route between their place of stay and Chandigarh.

Those who wish to participate may send their brief curriculum vitae containing name, date of birth, postal and e-mail address, qualifications, teaching experience, courses taught, positions held etc. to:
Professor Madhu Raka, Convener, Organizing Committee, Centre for Advanced Study in Mathematics, Panjab University, Chandigarh 160 014, India.

Research Fellows who wish to participate should also submit a letter of recommendation from their supervisors.

Last date for receipt of applications: September 7, 2002.