Role of mobile agents in mobile computing environment

Naveen Ramamurthy
Dept of Computer Science and Engineering
University of Texas at Arlington
ramamurt@cse.uta.edu

Abstract

Mobile Agents are an effective paradigm for distributed applications, and are particularly attractive in a dynamic network environment involving intermittently connected devices. One of the strengths of mobile agent computing is the possibility of moving complex processing functions to the location where huge amounts of data are to be processed. This paper gives a brief introduction to mobile agents and their suitability in mobile computing environment.

Introduction

Mobile agents can be considered as an incremental evolution of the earlier idea of "process migration". Mobile Agents are autonomous programs that can move from host to host in a network, interacting with resources and other agents. The state of the running program is saved and transported to the new host, allowing the program to continue execution from where it left off before migration. Mobile agent technology offers several potential benefits over conventional client-server computing that could help improve classic distributed system designs, which are usually based on the well-known remote procedure call (RPC) or its object-oriented equivalent, remote method invocation (RMI).

The client-server paradigm establishes a tight coupling between the client and the server. It assumes that there is no limitation in the bandwidth available between the machines as well as the presence and reliability of the machines. Mobile computing is characterized by low bandwidth and the resource constrained computer they operate on. The intrinsic properties of mobile agents technology like mobility and autonomy present solutions to this scenario. Agent mobility permits the optimization of the connection time required between mobile terminals and the network and autonomy permits asynchronicity between user actions and agent performed tasks. The mobile agents can continue executing while the user is temporarily disconnected, complete their task and wait for the user to reconnect to yield back results. The advantage here lies in the ability of an agent to perform both information retrieval and filtering at a server, and to return to the client only the relevant information. Thus the information transmitted to the device is minimized and the device need not perform filtering.

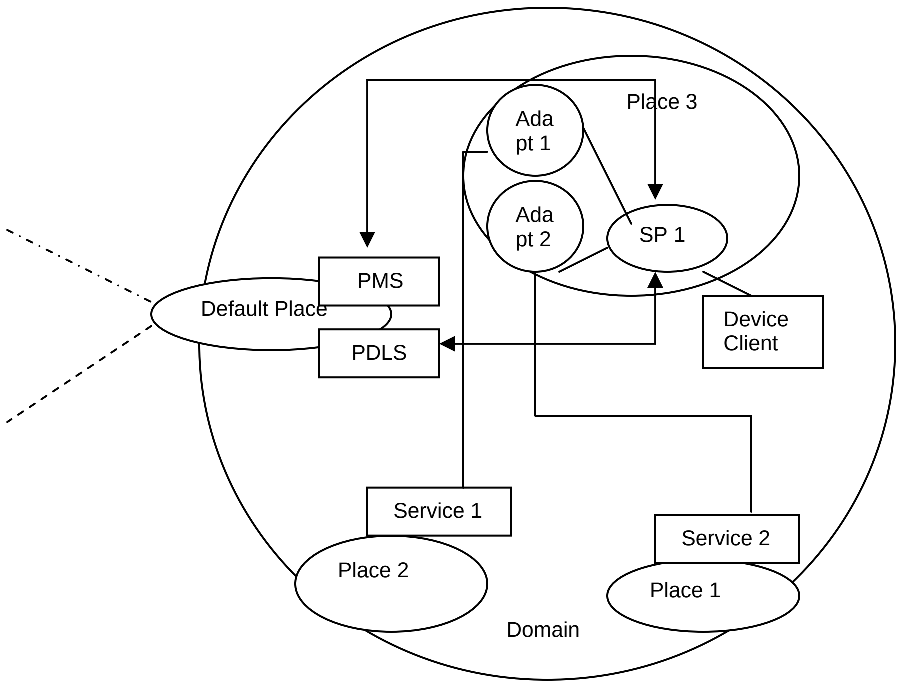
Providing Middleware Services to Mobile Devices using Mobile Agents

This scenario describes the use of mobile agents in providing middleware services using the Soma platform. Soma is a Java-based MA platform with a rich set of facilities for communication, migration, naming, persistency, security, and interoperability. It provides portable devices with both traditional and new location-dependent Web services. It provides any portable device with an MA-based companion, called the *shadow proxy*, and with application-specific MA-based processors, called *service adapters*.

Architecture

Any node hosts at least one place for agent execution; domain abstractions group several places and correspond to either fixed or wireless network localities. In each domain, a default place is in charge of interdomain routing. Each domain is provided with a local discovery service with intradomain visibility scope to enhance scalability in large-scale deployment scenarios.

The SOMA-based middleware for portable devices consists of the components in Figure 2: shadow proxies, service adapters, device-specific clients, the Portable Device Lookup Service (PDLS), and the Profile Manager Service (PMS).



Shadow Proxies are application-independent middleware components that represent portable devices on the fixed network. Each mobile device is associated with a shadow proxy. Shadow proxies are mobile agents running on a place in the SOMA domain where the device is currently located. Shadow proxies follow their mobile devices in their movements among different SOMA domains. When the portable device moves to a new domain the shadow proxy also migrates to that domain.

Service adapters are application-specific middleware components in charge of performing data transformations, depending on user or device profiles. Several different adapters can concurrently operate for one single shadow proxy to carry on parallel service requests from the same portable device. Service adapters are mobile agents that follow their shadow proxy's movements. Filters and transcoders are the two types of service adapters available in the SOMA domain. Filters can recognize and discard parts of service results when the mobile device cannot support their visualization. Transcoders are involved in transformations on service results, such as HTML-to-WML conversions.

Device-specific clients are components that run in mobile devices. These clients announce when the device enters or exits a network locality, request shadow proxies for services and receive the adapter service results.

Portable Device Lookup Service (PDLS) Each default place has a PDLS component. PDLS uses naming service to find the active shadow proxy for the mobile device and is responsible for triggering the shadow proxy migration to its network locality. When shadow proxies ask for services, PDLS identifies the needed service adapter and triggers the adapter migration to the requesting shadow proxy's location. The PDLS then provides a reference to the adapter that acts as the intermediate between the shadow proxy and the actual service component. If no adapters are compatible with the specified profiles then the PDLS sends a service unavailability message to the device client.

Profile Manager Service (PMS) maintains profiles of supported devices and registered users. It implements a partitioned and partially replicated directory service specialized for profiles. It keeps local copies of profile information and coordinates with PMSs in other SOMA domains to provide shadow proxies with the visibility of any profile registered in the system.

Supporting the WWW in Wireless Communications Through Mobile Agents

In this example mobile agents are used to relocate the cache of a mobile terminal from one base station to the base station of the control area to which the mobile terminal has moved. This allows a single cache to be maintained a mobile terminal and provides for accelerated web browsing.

Architecture

The Customer Premises Network (CPN) consists of a number of base stations (BS), which are interconnected though a LAN. BSs maintain information about all the mobile terminals that are currently under their control. The BS queries the user profile database and retrieves the relevant information. The path prediction algorithm (PPA) at the home agent is invoked after the entrance of the mobile terminal in the current control area. The home agent notifies the current BS with probability values to all the neighbours of the control area. From that point onwards, the current BS relocates its cache (or parts of it) to one of the BSs in the control areas indicated by the PPA. The MT is likely to attach to those control areas in the near future.

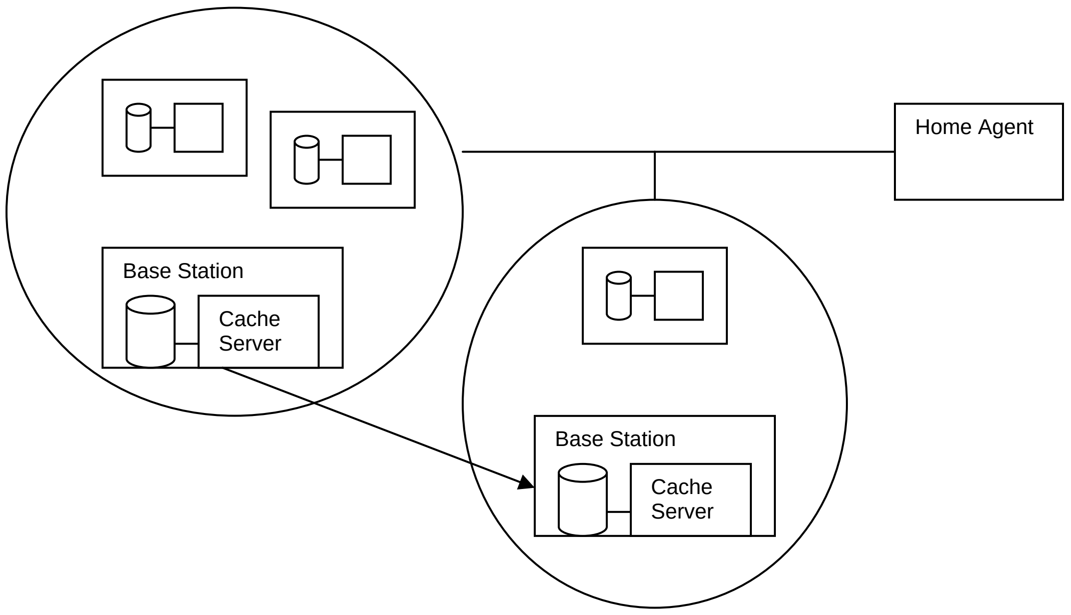


Fig. The moving cache technique

Aglets were used for the implementation of all procedures taking place in the access network. BSs provide the execution environment for the Aglets.

Types of Aglets

HRAgent is the stationary agent residing at the home agent. It manages information concerning users. HRAgent invokes the PPA for each registered user. HRAgent creates Watchers and dispatches them to BSs in control areas adjacent to the one in which the mobile device currently resides.

Traveler is the mobile agent, which represents the mobile terminal within the control area. It undertakes the task of cache relocation and is responsible for supplying the web content when requested. It follows the mobile terminal from control area to control area.

Watchers are agents created by the HRAgent and dispatched to BSs adjacent to the one currently handling the mobile terminal. They manage relocated caches and are responsible for reporting handover completion to the HRAgent.

TransferFSlave are agents created by the Traveler to read the files from the cache that belong to the Traveler's user. It returns the files to the Traveler once it completes its task.

Workers are agents that transfer the cache fragments to the BSs of adjacent control areas prior to handover execution. They communicate with the Watcher agent at the adjacent BS for the management of cache content.

Messengers are agents with the only duty to transfer remote messages between other agents.

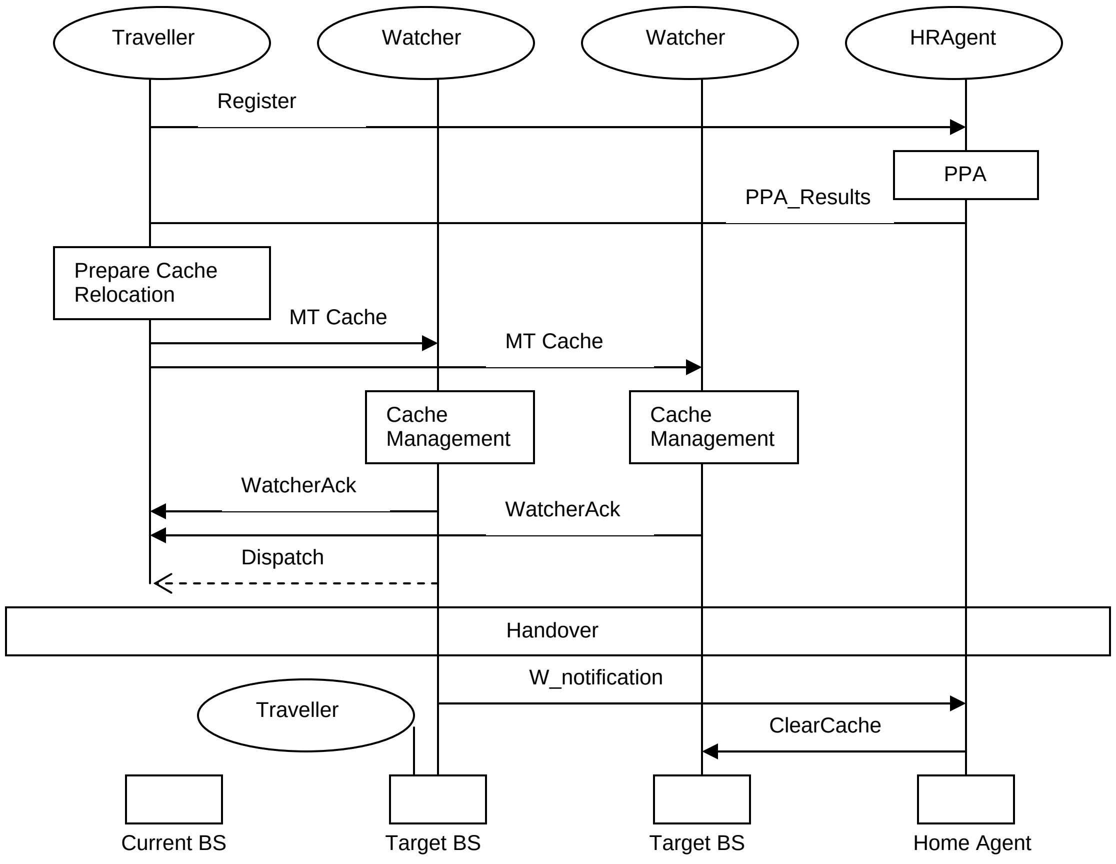


Fig. Message Sequence Chart for Aglet System

Mobile AGENT environment for distributed Applications (MAGENTA)

MAGENTA is a generic mobile agent environment and provides the agents the capabilities of autonomy, reactivity, proactivity and communication. The architecture of MAGENTA comprises of lieus and agents. A lieu is a place where an agent can originate, reside, execute and interact with the system as well as with other agents. The agents move between the lieus and utilize resources at the lieu to perform it's functions.

The agents have a unique name, have a purpose that defines its type and behave autonomously. They carry their results in a *folder* and can *meet* with other agents and exchange *notes*. The tasks performed by the agents are saved as *history* and traverses lieus using its *itinerary*. The agent gathers information about the failed sites in *knowledge*.

Lieus have unique names and provide an execution environment for the incoming agents. A lieu, which is executing on a stable machine and connected to the wired network, acts like a repository of information about other lieus. It maintains a table of <LN,IP,PN> tuples which correlates the lieu names (LN), their IP address (IP) and their port numbers (PN). The lieus register with the FirstLieu and are provided with a table containing the tuples about other lieus. The agents are allowed to access the services available at the lieu after they are authenticated. Lieus also facilitate agents to interact and move to other lieus. They also allow the agents to reside. When a mobile device hosting a lieu becomes unavailable, the agents destined for such lieus reside at the last lieus of their itinerary waiting for such lieus to reappear. Lieus maintain a repository of all the agents created and dispatched by it and also of all the agents executing on it. Lieus can launch an agent on a remote sites behalf upon request. The remote site can collect the results of the computation performed by the agent. The requesting site can be a lieu or a non-MAGENTA site. MAGENTA thus provides extremely resource-constrained devices like PDAs the facility to utilize the mobile agent functionalities and obtain the results of the computation without executing a lieu. In MAGENTA the lieus can appear dynamically and disappear dynamically. The lieu informs all the other lieus about its imminent disappearance and then quits gracefully.

A lieu executing on a mobile device might disappear abruptly after launching an agent. The agent after finishing its itinerary moves to the last destination lieu on the wired network and tries to move to the mobile lieu and fails. The agent then waits for the mobile lieu to connect back.

Agents and lieus may also disappear because of the abrupt disconnection of the mobile device or the crashing of the system executing the lieu. This necessitates a fault tolerance scheme to avoid the loss of the agents. In MAGENTA backup copies are maintained at a lieu before an agent migrates to the next lieu. The next lieu after sending the agent to another lieu informs the lieu with the backup to remove the backup. If this message does not arrive within a give timeout period the agent is recreated from the backup copy.

Conclusion

This paper discussed some of the scenarios where mobile agents can be used to overcome the resource constraints faced by the mobile devices. Mobile agent computing is still a young and immature technology. Several issues have not been resolved satisfactorily to date, among them the problem of malicious hosts, systems interoperability, and the question of performance measurement.

References

[1] D. Lange and M. Oshima. “Seven Good Reasons for Agents”. Communications of the ACM, March 1999

[2] C. Harris, D. Chess and A. Kershenbaum. “Mobile Agents: Are they a good idea?” IBM Research Report, March 1995

[3] W. Cockayne and M. Zyda. “Mobile Agents”. Manning Publications. ISBN 1884777368

[4] A. Shahai and C. Morin. “Mobile Agents for Enabling Mobile User Aware Applications”. Proceedings of the second international conference on Autonomous agents. May 1998

[5] P. Bellavista, A. Corradi and C. Stefanelli. “A Secure and Open Mobile Agent Programming Environment”. Proc ISADS99, March 1999.

[6] P. Bellavista, A. Corradi and C. Stefanelli. “A Mobile Agent Infrastructure for the Mobility Support”. Proceedings of the 2000 ACM symposium on Applied computing 2000 March 2000.

[7] P. Bellavista, A. Corradi and C. Stefanelli. “Mobile Agent Middleware for Mobile Computing”. Computer, Volume: 34 Issue: 3 , March 2001

[8] A. Sahai and C. Morin. “Mobile Agents for Location Independent Computing”. Proceedings of the ACM Symposium on Applied Computing, 1999

[9] D. Kotz and R. S. Gray. “Mobile Agents and the Future of the Internet”. ACM Operating Systems Review, August 1999

[10] S. Lipperts and A. Park. “An Agent-Based Middleware: A Solution for Terminal and User Mobility”. Computer Networks, August 1999

[11] V. Pham and A. Kharmouch. “Mobile Software Agents: An Overview”. IEEE Communication Magazine, July 1998