8 Mobile Agents: The State of the Art

B. Yang[1,2] and J. Liu[1]

1. Hong Kong Baptist University, Kowloon Tong, Kowloon, Hong Kong
2. Jilin University, Changchun, China

8.1 Introduction

This chapter will lead the reader into the world of mobile agents, a promising distributed technology that makes the design, implementation, and maintenance of distributed systems much easier than do traditional development methodologies.

Unlike stationary agents, which run on one host computer during their entire lifecycle, mobile agents can move their code to a new host where they can resume executing. The idea of transporting programs in a network is not new. However, a program whose sole ability is to transport itself from one host to another cannot be properly described as a mobile agent. Generally, mobile agents are a special kind of software – they not only possess basic characteristics (being generally autonomous, reactive, goal driven, and social), but also have the ability to move. Specifically, they can transport themselves from one site in a network to another but are not confined to the systems in which they began to execute.

The ability to travel, which distinguishes mobile agents from other types of agents, allows them to move to a new host and then to take advantage of being in the same environment to interact with each other locally. Lange, the inventor of Aglets, has surveyed the advantages and applications of the mobile agent paradigm [1]. Using mobile agents to develop a distributed system can yield many advantages, such as reducing network load; overcoming network latency; encapsulating protocols; executing asynchronously and autonomously; and adapting dynamically, natural heterogeneity, robustness, and fault tolerance. These advantages lead to improved performance in many distributed applications, and mobile agents are consequently viewed as a general paradigm by which to realize arbitrary distributed applications. These potential beneficiaries include electronic commerce, personal assistance, distributed information retrieval, telecommunication networks services, workflow applications, monitoring and notification, parallel processing, and so on [1].

Many mobile agent systems have been developed to date. These can be classified into two main groups – Java-based and non-Java-based. Aglets [2], Concordia [3], Voyager [4], and Grasshopper [5] are the most famous examples of the former. Telescript [6], Agent Tcl (renamed D'Agent) [7], Ara [8], Messenger [9], and TACOMA [10] are the most widely used non-Java-based systems, using script languages such as Tcl, Python, Perl, or Telescript.

In this chapter, we focus on the kernel technologies required to build a mobile agent system, discussing in detail the facilities, migration and planning, communication and interoperability, and security issues.

8.2 System Facilities

A mobile agent system is a platform that can create, interpret, execute, transfer, and manage agents. The architecture of most existing mobile agent systems is hierarchical, as illustrated in Fig. 8.1. A host can contain one or more of these systems, which themselves are likely to contain a number of places (also called context, location, or environment in some systems), which are the homes of mobile agents and provide various services. To provide security and scalability, the Object Management Group (OMG) introduces the concept of region into mobile agent system architecture [11].

The mobile agent system provides some basic facilities for safely executing local as well as visiting agents. These include naming, lifecycle, serialization/ deserialization, code base, transfer, communication and interoperability, security, and management facilities. A stronger system may provide additional useful facilities such as CORBA and event, transaction, yellow page, and domain name services:

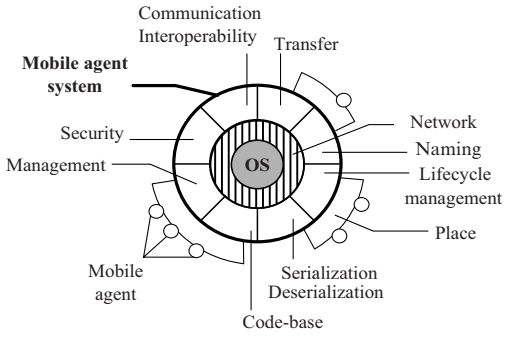

Fig. 8.1. The architecture of mobile agent system

- *Naming facility.* In terms of benefits, a qualified naming facility should at the least be able to (1) generate unique names for agent systems, places, and agents; (2) allow a system to quickly determine whether it can support an incoming agent and (3) enable agents to locate and identify others before communicating with each other by name.
- *Lifecycle facility.* Each mobile agent has its own basic lifecycle model, including some basic behaviors, the implementation and management of which is the responsibility of the lifecycle facility. For example, Aglets