Mobile Agent Coordination: MARS

Miss. Rashmi R. Atkare  
ME(CSE) 1st Year  
P.R. Patil College Of Engineering, Amravati

Prof. V. B. Bhagat  
P.R. Patil College Of Engineering, Amravati

Abstract

Mobile agents represent a secure technology. Mobile active computational entities introduce peculiar problems in the coordination of distributed application components. The paper surveys several coordination models for mobile agent applications. On this base, the paper presents the MARS system, a coordination tool for Java-based mobile agents. MARS defines Linda-like tuple spaces that can be programmed to react with specific actions to the accesses made by mobile agents. Several examples in the context of a WWW information retrieval application show the effectiveness of the MARS approach.

Keywords: Mobile Agents, Coordination, Java, WWW Information Retrieval

1. Introduction

Traditional distributed applications are designed as a set of processes statically assigned to given execution environments and cooperating in a (mostly) network-unaware fashion[1]. The mobile agent paradigm, instead, defines applications composed by network-aware entities (agents) capable of changing their execution environment by transferring themselves while executing (mobility) [2].

The mobile agents is broadly justified by the advantages their presence provides in Internet applications[3]: (i) mobile agents can dramatically save bandwidth, by moving locally to the resources they need, instead of requiring the transfer of possibly large amounts of data; (ii) mobile agents can carry the code to manage remote resources and do not need the remote availability of a specific server, thus leading to a more flexible application scenario; (iii) mobile agents do not require continuous network connection; as a consequence (iv) mobile agents intrinsically suit mobile computing systems.

2. Coordination Models

During its nomadic life, an agent is in need to coordinate its activities with other entities, let them be other agents or resources on hosting execution environments. More in particular:

- an application may be composed of several mobile agents that cooperatively perform a task and, then, are in need of coordinating their activities.
- a mobile agent is usually in need to roam across remote sites to access to resources and services there allocated.

Let us consider a simple WWW information retrieval application. An agent is sent to a remote site to analyse WWW pages and to come back with the URLs of the pages that contain a specific keyword. The agent clones itself for any remote link found in the pages of interest and sends the clones to the found sites, to recursively continue the searching work.

Two main characteristics distinguish different coordination models: spatial and temporal coupling.

- spatially coupled coordination models require the involved entities to share a common name space; conversely, spatially uncoupled models enforce anonymous interactions;
- temporally coupled coordination models imply synchronisation of the involved entities; conversely, temporally uncoupled coordination models achieve asynchronous interactions.

Therefore, four categories of coordination models can be derived (see figure 1).
Most of the Java-based agent-systems may supply HTML pages to agents in the environment resources, coordination usually occurs one. In the case of the access to the hosting communication protocol, typically a inter-agent coordination, two agents must agree on a synchronization (temporal coupling). In the case of partners (spatial coupling). This usually implies their communication by explicitly naming the involved partners. Interactions occur in the context of object systems, and can also exploit low-level message-passing via TCP/IP.

2.1 Direct Coordination
In direct coordination models, agents initiate a communication by explicitly naming the involved partners (spatial coupling). This usually implies their synchronization (temporal coupling). In the case of inter-agent coordination, two agents must agree on a communication protocol, typically a peer to peer one. In the case of the access to the hosting environment resources, coordination usually occurs in a client-server way [1,4], local WWW servers may supply HTML pages to agents. Most of the Java-based agent-systems – like Sumatra and Odyssey [5] – adopt the client-server style typical of object systems, and can also exploit low-level message-passing via TCP/IP.

2.2 Meeting-Oriented Coordination
In meeting-oriented coordination, agents can interact with no need of explicitly naming the involved partners. Interactions occur in the context of known meeting points that agents join, either explicitly or implicitly, to communicate and synchronise with each other. Meeting oriented coordination is implemented in Ara [6]: one agent can assume the role of meeting server announcing a meeting point at one hosting environment; incoming agents can enter the meeting to coordinate each other.

2.3 Blackboard-Based Coordination
In blackboard-based coordination, interactions occur via shared data spaces, local to each hosting environment, used by agents as common repositories to store and retrieve messages. As far as agents must agree on a common message identifier to communicate and exchange data via a blackboard, they are not spatially uncoupled.

The most significant advantage of this coordination model derives from fully temporal uncoupling: messages can be left on blackboards without needing to know, neither where the corresponding receivers are nor when they will read the messages.

2.4 Linda-like Coordination
In Linda-like coordination, the accesses to a local blackboard are based on associative mechanisms[7]: information is organised in tuples, retrieved in an associative way via a pattern-matching mechanism. Associative blackboards enforce full uncoupling, requiring neither temporal agreement nor mutual knowledge to let agents coordinate.

3. The MARS System
The MARS (Mobile Agent Reactive Spaces) system, developed at the University of Modena in the context of the MOON research project [8], implements a portable reactive Linda like coordination architecture for Java-based mobile agents.

4. MARS Architecture
The MARS system is conceived for the coordination of Java-based mobile agent applications in a wide heterogeneous network (possibly Internet). We assume that each node of the network hosts a mobile agent server in charge of accepting and executing incoming Java agents. Agents are supposed to move across nodes in a network-aware fashion, by explicitly asking for their migration. MARS defines a reactive tuple space model in which the effects of the operations on the tuple space can be dynamically modified. Agents access to the tuple space always with the same basic set of operations (Linda-like). The MARS system to implement reactions in the presented WWW information retrieval application. local execution environments provide access to the HTML pages via tuples that identify file general characteristics, such as, pathname, extension, dimension and modification time, and that contain a reference to a File object to be used to access the content of the file.

5. THE SECURITY MODEL
Security is a very important issue in the context of mobile agent applications. In MARS, security concerns protecting the tuple spaces from unauthorized or malicious accesses. MARS assumes that the mobile agent system – like most of the available systems – provides for agent identification and authentication before allowing any agent to execute on one node.

6. CONCLUSION
Coordination models that force some form of strict coupling between interacting entities are not suitable for the development of widely distributed applications based on mobile agents. As the paper shows in the context of a WWW information retrieval application, Linda-like coordination models have to be preferred because of their capacity of enforcing anonymous and asynchronous interactions. The additional capability of embodying programmable reactivity into the coordination model
permits to tune the semantics of the interactions and to lead to more flexible and secure coordination. In this context, the paper shows the MARS system, whose aim is to define a general coordination architecture based on reactive tuple spaces for the coordination of Java-based mobile agent applications.

7. REFERENCES:


