Mobile Agents for Resource Discovery in a Distributed Computing Virtual Community

F. AGOSTARO, A. GENCO, S. SORCE
DINFO - Dipartimento di Ingegneria Informatica
Università degli Studi di Palermo
Edificio 6 – Viale delle Scienze – 90128 Palermo
ITALY
http://www.genco.unipa.it

Abstract: - In this work we present a mobile agent based tool for arranging and managing communities whose members want to share computing resources. Such a tool will enable community members to suitably arrange their own distributed virtual machine. Mobile Agents are used to detect network addresses and computers that will be part of a virtual machine. We also describe a resource discovery multiple-strategy which allows a member to single out both hardware and software resources.

Keywords: - Mobile agents, distributed computing, resource discovery.

1 Introduction
Grid technology has brought a new way of exploiting computational resources. Nowadays these can be seen as an integrated system made up of a set of interconnected computers, sometimes remotely located. A grid computing common feature is coordinated and controlled resource sharing between the members of a dynamic multi-institutional virtual community. The members of such a community agree on which resources will be shared and which members of the community will be enabled to access these resources, thus defining a set of sharing rules and permissions. Resource sharing can take place by using a suitable protocol architecture, in order to pursue interoperability between the members of a virtual community.

In particular, the grid approach allows the configuration of a distributed machine which uses computing resources located on a network. This topic was widely discussed in literature, and many related problems have been solved through various software packages, like PVM [1], Linda [2] or P4 [3].

There are several existing projects aiming at the design of a framework for integrated management of high performance computation oriented services on distributed systems. Within most of these projects, some tools have been developed to implement the basic functions for communication, authentication, network management and data access.

As for instance in the Globus Project [4], which is the de facto standard for grid environments, computing communities are arranged in “Virtual Organizations” made up of groups who are building experimental and production grid infrastructures for their own purposes. Some low-level tools are provided in Globus to implement the functions listed above and, in particular, there is a module whose mission is the resource allocation and management within a single Virtual Organization.[5]

2 Problem formulation
Apart the software tool used to set up and use the virtual machine, one of the most relevant problem members have to deal with is the location of grid resources. Members with computational needs and available resources, even if somehow connected, do not know each other’s availability or needs.

In the following we propose a resource discovery system, by means of a Distributed Computing Virtual Community (hereafter DCVC) which uses the Mobile Agent technology.

According to a chosen discovery strategy, mobile agents are used to search among a set of addresses and are instructed to single out the ones of those sites fitting the member requirements in terms of hardware and software resources. Mobile Agent technology is used because of its capability to store the execution state, migrate to the next server and continue the execution there. [6]

Members who become members of a Community, publish their availability or needs, so they can interact according to a peer to peer resource provision protocol.
3 DCVC components
The main DCVC components are:
• the client Mobile Agent: this is the community’s key-component and acts for members who need computing resources. It pursues resource discovery mission according to the strategy described in the next section;
• the server application. This resides on the members machine whose resources are available. It receives Mobile Agents incoming from other members sites;
• a distributed database to store and manage the site addresses of the available resources. This extends our previous model, in which we used a centralized database and a different resource discovery strategy. [6]

Users who want to become members of the Community, have to install an agent-based software on their machines to act as client or server.

4 Resource Discovery strategy
Once a Community starts, the steps which lead to members cooperation for distributed virtual machine setup are:
1) When a user starts acting as a Community Member (CM), a copy of the AgentServer is launched and waits for clients’ agent. This way users become active members of the Community. Each member have a Friend User Table, in which a set of couples <resource, address> are listed. Notice that when a member registers within a Community, the Friend User Table is empty.
2) When a member shows his need for computing resources, an agent is generated and instructed to start the resource discovery process.
• First the Agent searches for the requested resources among addresses stored in its Friend User Table. If the search succeeds, the agent migrates to the site where resource are located and acts as a broker to start the cooperation. Now the agent steps into the network to test the current availability of the collected servers. This is necessary because one or more servers which have notified their availability could be not available anymore, or even be disconnected. For each IP address the agent tests the server’s matching to the client’s requirements. If the test succeeds, the agent books the server for a time interval large enough to end the test round. If the test fails, it proceeds its round with the next server. Once the test round ended, if the bookings placed by the agent allow the configuration required by the client, they will be confirmed, and the agent will go back to the client with the IP addresses list. This are used to configure the distributed virtual machine. Some other details are reported which can be used to plan a good load balancing strategy. If the bookings do not fit client requirements, the member will be notified of a request fault.
• If the requested resources are not found in its Friend User Table, the agent migrates to a Default Friend User (DFU), which is selected when a member registers within a Community. If the requested resources are found there, the agent migrates to the sites where resource are located and acts as described above. At its coming back, the agent address list will be used to setup the distributed virtual machine and to update the client’s Friend User Table too. Fig. 1 shows this case, where the Community Member CM$_{1411}$ needs some computing resources which it does not have in its Friend User Table. In this case the Agent migrates to the Default Friend User DFU$_{141}$. By querying the local database, the agent find the community members addresses where requested resources are located and starts its server booking task.
If the resources are not found within the Default Friend User, the agent continues its discovery process by recursive invocation of Default Friend Users, until reaching the community bounds, where a community manager can be invoked to manage the client query. This, which will probably entail a manual insertion of new addresses, aims at community enhancement, does filling the resource gap and providing the last querying member with the address of the requested resource. Fig. 2 shows this case, where red arrows are for querying agent route and green arrows are for coming back agent route.

Notice that an Agent can obtain some addresses from its Friend User Table, some others from the Default Friend User and some others by querying the community manager. At the end of its search, the agent will return an address list of available and suitable machines, or a message telling the unavailability of what requested. In the first case, if the address list has been obtained by recursive invocation of Default Friend User, the list is used to update the client’s Friend User Table.

Furthermore we want to point out that an agent is the key component of the system, because it takes care of the server availability test and of the decision whether to include them into the distributed virtual machine or not, according to member requirements.

4.1 A Resource Discovery strategy variant

In the case that one of the requested resources is a software one, as for instance a Dynamic Linking Library, the client mobile agent is instructed to use a different strategy.

If an Agent finds the requested software resource within a Default Friend User’s local database or by querying the community manager user, it can take the whole software resource and carry it with itself. The Agent can then bring the resource to the requesting member or to the Default User Friends already visited, depending on their settings. In the first case, while the Agent comes back to its owner, the previously visited Default User Friends update their local database with the new information. At the Agent arrival the requesting member can start using the software resource locally, and the whole Community knows that it has the resource (Fig. 3).
If the querying member are not enabled to directly download a software resource, the Agent can upload it to one or more of the Default User Friends visited while its coming back. At the Agent arrival, the requesting member can start using the software resource through the Default User Friend in which it has been uploaded, and the whole Community knows who has the resource (Fig. 4).

**Fig. 4 - Software lookup strategy 2**

<table>
<thead>
<tr>
<th>5 Conclusions</th>
</tr>
</thead>
<tbody>
<tr>
<td>In this work we presented a mobile agent-based system for resource discovery in a distributed computing virtual community for hardware and software resources sharing. The proposed strategy and its variant turned out to be efficient, fault tolerant, and capable of dynamically adapting themselves to environment changes. Future works on these topics will include security and authentication features and enhanced booked server management.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6 References</th>
</tr>
</thead>
</table>