AGENTS - MOBILE AGENTS IN JAVA

M. Dönszelmann
CERN, Geneva, Switzerland

Abstract
The CERN School of Computing provided an excellent opportunity to try out mobile agents on physics analysis. The course explained the general concepts of mobile agents and their applicability to the field of High Energy Physics. In particular the students took a look at the merits of using mobile agents for physics analysis.
The course consisted mainly of lab works which provided both inexperienced and experienced students in the fields of Object-Oriented programming and Java an interesting introduction into the working of Mobile Agent Systems. A full Distributed Analysis System was put together.

1. MOBILE AGENTS

Mobile agents[1] are programs that can move around on the network, while performing their duty, which may be a calculation, a database lookup or some other service. They keep their state as they move along from one machine to the next, thereby taking with them the result they have obtained so far. Machines not only provide a place for agents to work, but also a place for agents to meet and exchange information. A mobile agent may change its itinerary depending on the information it receives from another agent. Mobile software agents very much resemble the way people act, work and meet in the real world.

The fact that mobile agents can move themselves with their information across the network can reduce network traffic. For example, to discover information on another node, we would normally interrogate a server on that node via a mechanism such as remote procedure calls (RPC). We may have a set of questions, where later questions depend on the result of earlier ones. Using RPC every question (and its answer) would be separately transferred across the network. For mobile agents we use the term remote programming (RP). We instruct an agent with a set of questions and send it (with its questions) over to the remote node. The agent asks its questions locally and comes up with the final answer. It then travels back, only taking the final answer with it. Using RP, the network bandwidth consumed is the one of the agent and its final result, rather than all intermediate information of all the questions. We use the locality of information, to reduce the network bandwidth.

TeleScript[1], one of the oldest mobile agent systems, defines the concept of agent and place. An agent stays in a place and can do some work there. It may travel to another place, either by instructing itself, or being instructed by some other agent. Agents can meet other agents and communicate across the network to other agents. A place provides an environment for a agents to stay, to meet and to communicate. It provides security to protect the host system from hostile agents and to protect agents from each other. A set of collaborating agents, which can safely travel from one place to another and communicate with each other, may provide for a higher level service then the sum of the services provided by each of the agents individually.

Mobile agents have been around for some time, and many systems have been built to deploy them, such as TeleScript. The Java language[2] and its virtual machine seem to be very appropriate to implement an agent system. Java is platform independent, allows for serialization and persistency and comes with security built into the virtual machine. These are just some of the features needed to create an agent system. Several Java implementations exist today, one being the Aglet system[3] from IBM, which follows best the concepts of TeleScript.

2. COURSE SUMMARY

The course consisted of three parts: an introduction into agent system, implementations and examples, and an explanation of the exercises.

The introduction into agents mainly followed the book by W.T. Cockayne and M. Zyda[1]. The concept of mobile agents was explained using 21 students, who played a small game on stage. In this game 7 students were persons with some characteristics. 7 other students were tasks which had to calculate some number based on the characteristics of the persons. The final 7 volunteers were agents, which moved around on stage, gathering information from the persons, doing the calculation and bringing the result back to their task.

The implementations of agent systems using the Java platform was discussed in depth, including some smaller examples such as a distributed web-search, which uses mobile agents for its search strategy and its results. In the area of High Energy Physics three examples were given: a smart e-mail system, in which e-mail is in fact a mobile agent gathering updates on a physics paper, a slow controls system in which alarms of different severity are sent around as mobile agents, who then meet and decide if a real alarm should be triggered, and a distributed physics analysis system, which the students were supposed to built during the lab works.

The distributed physics analysis system uses mobile agents for physics jobs. This enables us to make the job travel from one data set to another, without moving the actual data. The data sets are assumed to be in different places. The job travels with its set of histograms (results), and fills them with information from each data set. Note that this is different from a job submission system, in which jobs cannot pack up and move to the next data set, and results have to be merged afterwards. An optimization could take place having agent jobs run in parallel on different data sets. Results would still have to be merged, but that responsibility is now with the agent job.

3. LAB WORK SUMMARY

The exercises consisted of three parts. You could choose between Part-A or Part-B, followed at all times by Part-C. The documents which describe the exercises are available from the agents web site [4].

3.1 Part - A

Part-A was meant for the non-java and non-object-oriented programmers. No programming knowledge was required. Some explanation on Object Oriented Programming was provided. The description in the document for part A was fairly explicit. A step by step description on what to do, using examples, would tell the student how to reach his end goal.

As a first exercise one created a set of three simple classes, and made a small calculation. As a second exercise an Analysis Job was programmed, thereby making use of pre-fabricated classes from an Analysis library. It was this Job which was to be handed to the group doing Part-B, who would run it in their Distributed Analysis System. Optionally the exercise was extended to create some agents which travelled around and did some calculation.

3.2 Part - B

Part-B was meant for the java or c++ experienced. The goal of part B was to write a distributed analysis system, which would take a job of part A and move it around on the network. An Agent System (Aglets) was provided, but some Agent classes had to be written. The description in the document for part B was in the form of requirements for classes and hints on how to implement these. There was no step by step description provided, assuming the students would know how to go about.

3.3 Part - C

At the end of the course, the whole system would be exercised on the machines in the lab. One would actually try to measure network traffic.

4. CONCLUSIONS

A lot of things went right during the exercises, the most important part of the course. Most people started doing part-A, some with little knowledge on object-oriented programming, learning Java, understanding Agents and most groups finished it, including the optional extension.

Part-B was done by a smaller group, including some who finished early with part-A. Here is where some things went wrong. Two problems in the agent system were discovered, one to deal with naming, the other with security. It was thought that the Aglet system was still pretty fragile to use for production.

Only a few groups managed to get as far as Part-C, mainly due to the bugs discovered. They run their system over several machines analyzing data by moving agents across the network, rather than moving data. No network measurement was performed though.

In a feedback session, held at the end of the course, it was thought it would be a nice idea to try the same exercise, running agents world-wide, on the machines of the students at their work-place. A decision was taken to set up a web site for this, to wait for an upgrade of the Aglet system, so that the bugs found earlier would be fixed and to try this out before the next school. Information can be found on the agents web site [4].

5. ACKNOWLEDGMENTS

I would like to thank those people who helped making the course on Agent System a success. In particular I thank Paolo Palazzi, my former boss, for giving me the idea to teach about agent systems at the school and Gottfried Kellner for letting me spend time on the preparation of it. A team of people, Arash Khodabandeh, Alberto Aimar, Klaus Ackerstaff and Eberhard Wolff were present during the exercises and lectures and helped out with the logistics and organization. Mario Ruggier and Bertrand Rousseau were both helping behind the set in providing document templates and logistics at CERN. Andreu Pacheco and Fabien Collin provided a stable set of machines on which we could do our exercises. Thanks to all of them. However, I should not forget to thank all the students of the school for their active participation, including the summer students of 1998 at CERN, who tried out the course in a mini version. You all made this course a great success.

6. REFERENCES

[1] W.T. Cockayne and M. Zyda, *Mobile Agents,* Manning Publications Co., 1998.

[2] M. Campione and K. Walrath, *The Java Tutorial: Object-Oriented Programming for the Internet,* Addison-Wesley, 1998, http://www.javasoft.com/docs/books/tutorial

[3] D.B. Lange and M. Oshima, *Programming and Deploying Java Mobile Agents with Aglets,* Addison-Wesley Publishing Co., 1998.

[4] CSC’98 - Mobile Agents Web Site, http://iptnt.cern.ch/csc98agents