Application Centric Mobile Agent Systems: Bringing the Focus Back to the Applications

Paulo Jorge Marques, Luís Moura Silva, João Gabriel Silva
CISUC, University of Coimbra, Portugal

{pmarques, luis, jgabriel}@dei.uc.pt

1. Introduction and Motivation

The Mobile Agent (MA) paradigm has now been around for some years. Although there are many advantages associated to the use of mobile agents and many agent platforms are currently available from both the research community and industry, they have not reached a wide acceptance. In fact, mobile agents have gained the reputation of being "a solution looking for a problem" [1], and finding a "killer application" is still on the mind of many practitioners of the technology.

Many reasons are typically pointed out on why the technology is not being so successful as it could. Examples include security problems, lack of proper support for fault tolerance and lack of standards. Although these are important problems, we believe that mobile agent systems are not widely deployed fundamentally due to different reasons. The mobile agent community has been mainly focusing on the agent technology and on the mobility issue, rather than on the support needed for real-world application development. The problem can be viewed in two dimensions: the programmer and the user.

The Programmer

From the point of view of the programmer, constructing an application that uses mobile agents is a difficult process. Current mobile agents systems force the development to be centered on the agents, many times requiring the applications themselves to be coded as a special type of agents – *stationary agents*. When this does not happen, special interface agents (*service agents*) have to be setup between the application and the incoming agents. These agents must know how to speak with the mobile agents and with the application. Although the mobile agent concept – *a thread that can move to another node*, is a very useful structuring primitive, all the currently required setup is an overkill that prevents acceptance by the developers.

Since basically anything that can be done with mobile agents can be done using simple client/server remote method evocations, the reasoning goes: "Mobile agents do not give me any fundamentally different (and needed) mechanism, and at the same time force me to develop systems in a completely different way. Why should I bother to use them?"

The problems include: the mobile agent concept is not readily available at the language level; the applications have to be centered on the mobile agents; and a complicated interface between the agents and the applications must be written. The programmers want to develop their applications as they currently do. Agents will typically play only a small role on the application (90-10 rule: 90% traditional development, 10% mobile agents). Current systems force exactly the opposite.

The User

From the viewpoint of the user, if an application will make use of mobile agents then it is necessary to first install an agent platform. The security permissions given to the incoming agents must also be configured and the necessary hooks to allow the communication between the agents and the application must be setup. While some of these tasks can be automated using installation scripts, this entire setup package is too much of a burden for the average user. Usually, the user is not concerned with mobile agents nor wants to configure and manage mobile agent platforms. The user is much more concerned with the applications than with the middleware they are using in the background. In the currently available mobile agent systems, the agents are central and widely visible. They are not the background middleware but the foreground applications.

Another important issue is that the term "*mobile agents*" has very strong negative connotations that make the dissemination of the technology difficult. The user is afraid of installing a platform capable of receiving and executing code without his permission. This happens even though the existence of mobile code is present in technologies like Java, in particular in RMI and JINI. The fundamental difference is that in those cases, the user is shielded from the middleware being used. In many cases, using mobile agents does not pose an increased security threat, especially if proper authentication and authorization mechanisms are in place. However, because the current agent platforms do not shield the user from the middleware, the risk associated with the technology is perceived as being higher, which causes users to back away from applications that make use of mobile agents.

2. The M&M Approach

In the M&M project at the University of Coimbra, we are working on an approach to overcome some of the problems previously identified. We are developing flexible lightweight JavaBeans software components that when incorporated into an application gives it the capability of sending and receiving agents. The application is developed using industry OO development best-practices and can additionally become agent-enabled by the simple drag-and-drop of mobility components. In our approach, the emphasis is put on the development of applications, not on the agents. Each agent arrives and departs from the application that it is specific. The application knows the interface of the agents and the agents know how to interact with the applications (Figure 1).

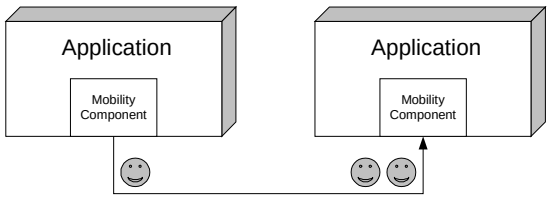

Figure 1 – Applications become agent-enabled by using mobility components.

From the programmer's perspective, all he has to do is to use the mobility components and write the agents. The agents arrive and departure directly from the application without the needing a fully blown agent platform. From the user point of view, he just sees an ordinary application. The usage of mobile agents is completely shielded from him. This is a step forward over the traditional development approaches used in the platform-based MA systems. Because in this approach there is no mobile-agent platform and because the emphasis is put on the application and not on the agents, we call this approach *ACMAS – Application-Centric Mobile Agent System*.

M&M is being implemented using the JavaBeans component model. Nevertheless, we have also decided to support the ActiveX component model, that is based on Microsoft's COM [2]. We have taken this decision because of the wide adoption of this component model in the software industry and in research. We believe that by supporting ActiveX, a much wider usage base and a much richer set of environments where to use our framework will be available. Another important point is that any language that has the necessary mechanisms to make use of COM and ActiveX is able to use the mobility components. At the present, most of the languages that available for the Windows [3] operating system have that capability. By supporting ActiveX, we are no longer limited to develop the applications in Java. We can program the applications in languages like Visual C++, Visual Basic and Delphi. This happens without having to implement any special layers between our components and the target languages. During our experiments, we developed a simple instant messaging application in Visual C++, Visual Basic and Java. The agents migrated and executed in all the client applications independently of the language in which they were written.

At this stage of the project, we already have a rich component palette that in a modular way supports many of the features found in traditionally MA platforms. We have components for mobility support, inter-agent communication, security, agent tracking and infrastructure management. We are currently expanding the component palette into two different application domains: accessing information systems in disconnected computing environments [4] and network management. Our vision is to build an extensive framework where the programmer combines components available off-the-shelf from third-party software producers, components for supporting mobile agents, and components designed specifically for the application domains being considered.

3. Conclusion

We believe that the factors that are preventing a wide adoption of the mobile agent paradigm are not only related to the technological problems that still exist, but to the great overhead that its usage imposes on the programmers and users. When weighting the benefits obtained from using mobile agents against the implications of having them in terms of software development and utilization, the programmers and users choose not to use them.

In the M&M project, we are trying to overcome the limitations identified on the traditional platform-based model by using binary software components. The idea is not to use traditional mobile agent platforms but to embed sufficient support for mobility inside of the applications.

References

[1] *The Future of Software Agents*, Volume 1, Number 4, IEEE Internet Computing, available at http://computer.org/internet/v1n4/round.htm, 1997.

[2] D. Rogerson, *Inside COM*, Microsoft Press, 1996.

[3] "Microsoft Windows Products Homepage," http://www.microsoft.com/windows/default.asp.

[4] P. Marques, L. Silva, J. Silva, *A Flexible Mobile-Agent Framework for Accessing Information Systems in Disconnected Computing Environments*, to be presented at the Third International Workshop on Mobility in Databases and Distributed Systems (MDDS'2000), Greenwich, London, September 2000.