Mobile Agent Systems: From Technology To Applications

Paulo Marques, Paulo Simões, Luís Silva, Fernando Boavida, João Gabriel
CISUC, University of Coimbra, Portugal
{pmarques, psimoes, luis, boavida, jgabriel}@dei.uc.pt

Abstract

Over the last couple of years we have been working on the development of mobile agents systems and its application to the areas of telecommunications and network management. This work path produced positive results: a competitive mobile agent platform was built, the run-time benefits of mobile agents were proved, and our industrial partners have developed practical applications that are being integrated into commercial products.

However, despite the positive results, we feel that mobile agent technology is still not ready to enter the path of mainstream software development. This is the reason why this technology has not yet gain large acceptance by the developers of real applications, despite the large number of interesting experiences already available.

In our perspective, one of the main reasons for this situation arises from the traditional approach to mobile agent technology. This approach, based on the familiar concept of mobile agent distributed platforms as extensions of the operating system, focuses too much on mobile agents and associated issues (mobility, agent lifecycle, security, coordination, etc.) and provides poor support for the development of applications where mobile agents are just one of several available technologies.

Learning from past experience, we are now working on a new approach where the focus is brought back to the applications and mobile agents become just one the tools available to develop distributed systems. This provides a much lighter framework for application-based mobile agents

This paper presents the lessons learned from our previous project and discusses this new concept we are developing: application-centric mobile agent systems.

1. Introduction

Over the last few years we have observed the proliferation of available mobile agent (MA) systems, currently reaching the impressive number of seventy-two known platforms [1]. This is a lot more than available RPC [2] or CORBA [3] implementations. Nevertheless, this technology is still far from the mainstream programmer and, oddly as it seems, there are now much more MA platforms than MA-based applications.

In the last couple of years we were involved in the JAMES project [4]. This project was held in consortium with Siemens Portugal SA and Siemens AG, and the objective was to develop a new MA platform specifically tuned and customized for the area of telecommunications and network management (NM). Our industrial partners used this platform to produce a few MA-based applications that are now being integrated into commercial products. These applications use mobile agents to perform management tasks (accounting, performance management, system monitoring and detailed user profiling) that deal with very large amounts of data, distributed over the nodes of GSM networks.

With this project we have learned that MA-technology, when appropriately used, provides significant competitive advantages to distributed management applications. However, we have also realized that the current status of MA technology is still not appropriate to take those advantages to mainstream application development.

In order to overcome some of the limitations found, we are now working on a new approach that abandons the classic concept of MA platforms as extensions of the operating system. Instead, it focuses on providing agent mobility within application boundaries, rather than within system boundaries.

In this paper we present the lessons learned from the JAMES project and discuss our on-going work to address the identified problems.

2. Platform-based MA Systems: A Critic Perspective

The JAMES project took place from March 1998 to December 1999, and involved two teams. The first team developed a MA infrastructure specifically tuned for NM applications. Like other known MA implementations, the idea was to have a set of platforms providing an execution environment for mobile agents, controlling their migration and lifecycle. These platforms are seen as an extension of the host's operating system (typically one platform per host) where mobile agents from different applications coexist. Following this *platform-centered* design we focused on issues like performance [5], robustness, integration with legacy management technologies [6], infrastructure manageability [7], agent coordination and security.

The second team used this platform to develop management applications. Four prototypes were produced and evaluated, and two of them were selected for integration into commercial products.

This separation between platform and applications, as well as the desire to produce commercial products, provided a good perspective on the reasons why there are so many mobile agent infrastructures and so few real world applications. This perspective can be described in three dimensions: the programmer, the user (costumer) and the application field (in our specific case network management).

2.1. 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 to use it 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?"

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 a small part on the application (90-10 rule: 90% traditional development, 10% mobile agents). Current systems force exactly the opposite.

The point is that mobile agent technology should not be an exclusive of agent-based software design. It should be available, as well, to traditional object-oriented software developers.

2.2. The User

From the viewpoint of the user, if an application will make use of mobile agents it is necessary to first install an agent platform. The security permissions given to the incoming agents must also be configured and the proper hooks necessary to allow the communication between the agents and the application must also be setup. While some of these tasks can be automated using installation scripts, this entire setup package is too much of a burden. 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.

The term mobile code also has very strong negative connotations that make the dissemination of the MA 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 this technology is perceived as being higher, which causes users to back away from applications that make use of mobile agents.

[*] Mobile agents do impose more complex security problems in fully open environments where there is no user accountability (for authentication and authorization). Nevertheless, most applications are not deployed in that kind of environments.

2.3. The Application Field

The primary application field of the JAMES project was network management. The mobile agent paradigm fits very well into the conceptual foundations of distributed and delegated network management. This is one of the reasons why NM is so often considered one of the most attractive demonstration fields for MA technology. Nevertheless, in general, the NM community is still reluctant to accept the advantages of mobile agents, even when Active Networks [8], which share the same principles and technologies, are of the most promising NM research areas.

One of the main reasons is that mobile agents tend to be exclusively associated to multi-hop migration, where an agent successively visits several network nodes performing a given task. This multi-hop model is seldom the most appropriate delegation technique for NM applications [9,10]. Nevertheless, there are other models for using mobile agents.

In our experience we have not found many NM applications requiring multi-hop migration. However, we have concluded that the mobile agent concept is a very appropriate paradigm for deploying distributed and delegated management services, successfully competing with other technologies like push/pull mechanisms, RPC, CORBA and the SNMP Script MIB [11].

The already mentioned cost of installing and maintaining the agent support infrastructure is also a very sensitive argument for the NM community, which wants manageable management systems.

Poor interoperability with legacy NM architectures, resulting in poorly integrated applications, and increased security risks are other relevant reasons why mobile agents have not yet gained wide acceptance in the NM field.

This scenario is probably similar to other application fields and highlights some of the major problems with the way mobile agents are perceived:

- When compared with existing solutions, mobile agents tend to be exclusively evaluated for *what more they can do* (multi-hop migration) rather than for *what can they do better*.
- The security constraints are considered as unacceptable, even though currently used technologies (namely SNMP [12] and CORBA) provide poorer or no security at all. Our experience is that in the majority of application environments mobile agent based applications can already provide better security than current solutions.

2.4 Conclusions

We have discussed some of the obstacles to widespread deployment of mobile agent systems. Although these obstacles were identified during one specific R&D project, we feel they are shared by the generality of mobile agent systems, and should be added to the list of most commonly pointed problems of agent technology: the lack of killer applications, security, performance and scalability [13]. Nevertheless, these problems are counterbalanced by the positive conclusions from a number of mobile agent projects (including JAMES):

- It was demonstrated that mobile agents can provide an excellent mechanism for building distributed applications.
- Despite all the problems it is possible to build competitive commercial-level applications that use mobile agent technology.

The fundamental conclusion is that the long list of problems is more related to the way mobile agent technology is being perceived and implemented, rather than with the paradigm itself. Our current line of work addresses these problems by proposing a different perspective on mobile agents.

3. Application-centric Mobile Agents

3.1. Back to Applications

The most distinctive characteristic in our new approach is that there are no agent platforms. Instead, agents arrive and leave from the applications they are part of, not from agent platforms. The applications become agent-enabled by incorporating well-defined binary software components [14] into their code. These components give the applications the capability of sending, receiving and interacting with mobile agents. The applications themselves are developed using the current industry best-practice software methods and become agent-enabled by integrating the mobility components. We call this approach *ACMAS – Application Centric Mobile Agent Systems* ─ since the applications are central and mobile agents are just a part of the system playing specific roles (see Figure 1).

The consequences of ACMAS are as follows:

- In this way it is not necessary to design the whole application around agents. Agents are sent back to middleware, in pair with other distributed programming technologies.
- Security is integrated with the application security framework, rather than being completely generic (end-to-end argument [15]).
- Agents interact directly with the applications from the inside. This eliminates the need to setup interface agents and configure and manage their security policies.
- There is no agent platform to install and maintain. Although there are still distributed applications to install and manage, this is much simpler than managing a separate infrastructure shared by a large number of distributed applications with different policies and requirements.
- The end-user sees applications, not agents. In this way, the acceptance of applications that use mobile agents is increased since what the end user sees is the added value functionality, not the agents.
- It is simple to program. The programmer only needs to visually drag-and-drop the necessary components from a component palette and configure their properties and interconnections.

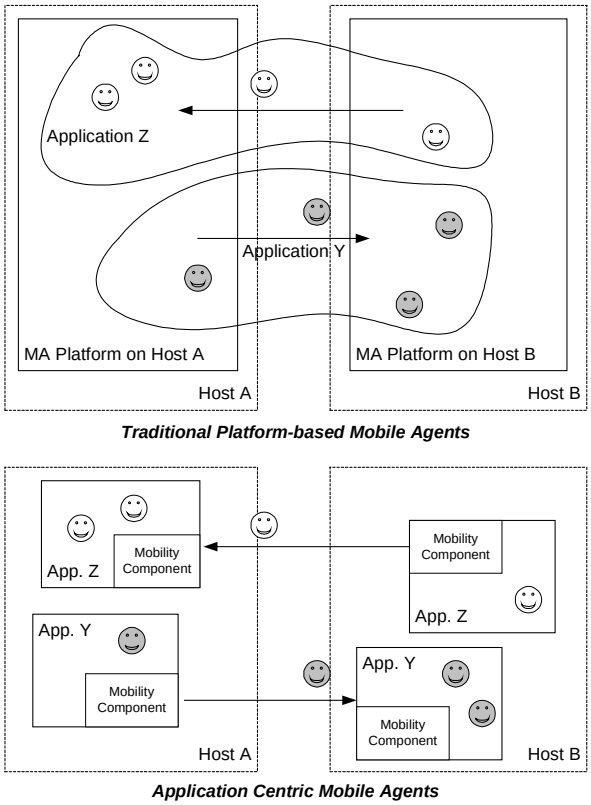

Figure 1 – Platform Based and Application Centric MAs

In ACMAS, the applications are developed using three different kinds of components (see Figure 2):

- Domain specific components.
- Third-party off-the-shelf components.
- Mobile-agent support components.

Mobile-agent support components provide the basic needs in terms of mobile-agent infrastructure. We currently have components for supporting the migration of agents between applications, components for supporting different inter-agent communication mechanisms, components for agent tracking, security and others.

Third-party off-the-shelf components are components that are commercially available from software makers and can be used for building the system. Currently there is a large variety of components available for the most different

things, like accessing databases, designing graphical user interfaces, messaging and others. All these components can be used for building the applications without having to re-implement the required functionalities.

Domain specific components are modules that must be written in the context of the application domain being considered, providing functionalities not readily available off-the-shelf. For instance, while implementing a particular application it may be necessary to write special parsers for extracting information from files, or to write supporting services for allowing agents to monitor the hardware of a machine. These modules can be coded as components and incorporated into the application.

One important point in ACMAS is that while developing an application only the components required for that particular application domain have to be included. Also, because the features available to the programmer are implemented in separate binary components, which have well-defined boundaries, it is possible to expand the package without influencing already developed applications. Each time a new feature is required or a new service implemented this can be done by creating a new component. This allows a high degree of flexibility, since the component palette is constantly being enriched with new components. At the same time, the new features do not force the applications to become heavier or bulkier since only required functionalities are introduced in each application.

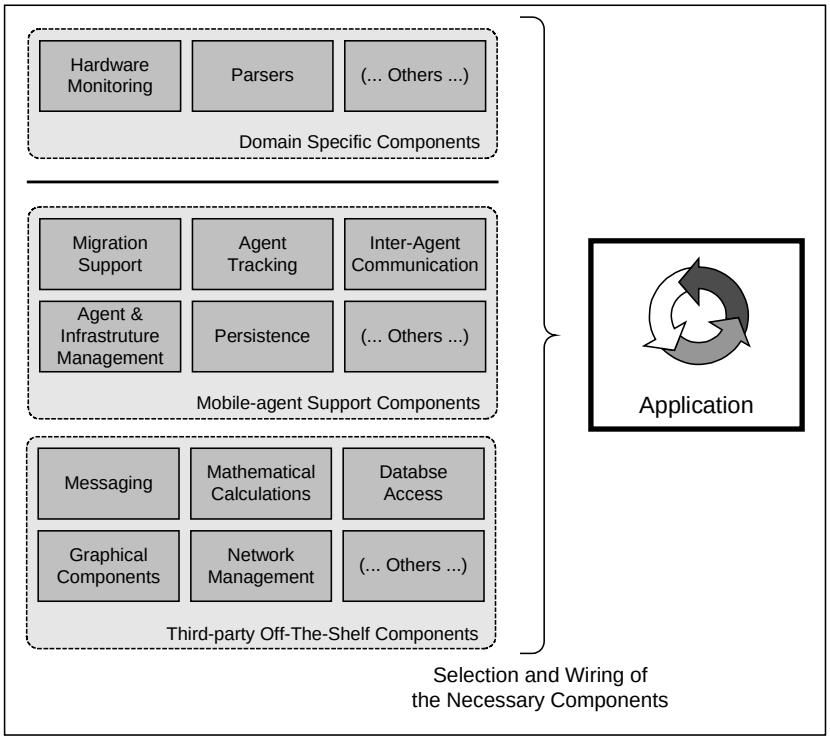

Figure 2 –Applications Developed By Wiring Different Kinds of Software Components

3.2. Application Domains Being Explored

In order to evaluate the strengths and weaknesses of the ACMAS approach, when compared to the traditional platform-based model, two application domains have been selected:

- Accessing Information Systems in Disconnected-Computing Environments
- Network Management

3.2.1. Accessing Information Systems in Disconnected-Computing Environments

Over the last few years, mobile phones, laptops and personal digital assistants (PDAs) have become commonplace. Laptops and mobile phones are already reshaping the way people work. As mobile devices become more powerful, users are starting to expect to have access to information in any place they are, by using such devices.

To complicate maters, today's Corporate Information Systems (CIS) are being deployed using a three-tier architecture [16]. Thus, accessing the databases is no longer enough. It is becoming increasingly important to have mechanisms that allow mobile end-devices to access and interact with the business logic present in the middle-tier.

Mobile agents are a very interesting approach to software development in disconnected computing environments [17,18]. The advantages of using mobile agents in mobile computing include:

- Connections must only be up for receiving and sending the agents.
- Data must not be transferred from the server to the client: the agents just process it at the server.
- Multiple interactions occur locally at the server, between the agents and server processes.

We are currently exploring the ACMAS approach for building systems where mobile agents provide the base mechanisms for allowing client applications to interact with the business logic present on the information systems [19]. ACMAS is especially interesting in this context since the requirements at the client-side are very different from the requirements at the server-side. ACMAS provide a very flexible way of addressing these different requirements by having different components deployed in the applications of the end-devices and on the server.

3.2.2 Network Management

Network Management has always been one of the most privileged demonstration fields for MA technology. In fact, it was during JAMES that we have identified many of previously mentioned problems of the platform-based approach.

While the previous application domain gives us the opportunity to experiment directly at the application level, for network management we are building a domain-specific component palette that gives applications and agents the services that we found to be the most interesting on this application field:

- Multi-level delegation of management support.
- Distribution of management services across unstructured topologies.
- Disconnected or very low bandwidth operation.
- Dynamic service deployment, reconfiguration and relocation.
- On-the-fly extension of installed management services.
- Heuristic data collection over large network domains.

Working on a component palette for the specific domain of network management is giving us the opportunity to assess the limitations and strengths of the implemented extensibility mechanism for supporting new services [20], and it is also allowing us to evaluate ACMAS in the context of building non-hierarchical management meshes supported by mobile agents.

3.3 Lessons Learned

From building the ACMAS component palette and developing some prototype applications, we have already gathered some important lessons.

When developing applications based on mobile agents, it is a lot more easy to use components to agent-enhance the applications than to center all the development around agents, where complicated setups have to be done. This is especially important if it is necessary to use other middleware like CORBA [3] or SNMP [12]. While current MA frameworks do not integrate well with existing middleware, applications using the mobility components can transparently use other middleware solutions.

After presenting some demonstration applications that are mobile-agent enabled, the reaction of the users was very positive. One key point for this was that they were not aware of the mobile agents but only of the results obtained from the applications. This clearly contrasts with our experience on presenting applications based on classical platform-based systems, where the use of agent-technology typically raised concerns and some suspicion.

Developing components for supporting mobile agents can be hard. When it comes to security, it is not trivial to design an approach where the mobility components do not impose restrictions on the application. In addition, when developing distributed network components (e.g. agent tracking support), managing the configuration of the components becomes complicated since there is no central point to address. This typically requires that the configuration must be replicated on the existing components. Technologies like Sun's InfoBus [21] may help to solve some of these problems.

For a more complete account of the experiences we are having while implementing the framework, please refer to [19-20,22].

4. Conclusions

Over the past two years we have been trying to bring mobile agent technology to the mainstream development of management applications. We have gathered a strong experience on the advantages and shortcomings of current mobile agent technology and its deployment on real systems.

The main conclusion from that experience is that although it is complicated to develop systems on the current state of the technology, the mobile agent paradigm provides important advantages in the context of network management and other application domains.

However, a lot of work is still required before the average programmer of distributed applications can use off-the-shelf tools for using mobile agent technology. We believe the ACMAS approach, which focuses strictly on applications and simplifies the usage of mobile agents, is one step towards this vision.

Acknowledgments

The JAMES project was partially supported by ADI (*Agência de Inovação*) and was accepted in the European Eureka Program (Σ!1921). The research on application-centric mobile agents is partially funded by FCT, through programs PRAXIS XXI (scholarship number DB/18353/98) and CISUC (R&D Unit 326/97).

References

[1] Mobile Agent List, http://www.informatik.uni-stuttgart.de/ipvr/vs/projekte/mole/mal/mal.html

[2] R. Orfali, D. Harkey and J. Edwards, "*Client Server Survival Guide - Third Edition*", John Wiley & Sons Inc., 1999.

[3] OMG, "*The Common Object Request Broker Architecture and Specification*", 1995

[4] L. Silva, P. Simoes, G. Soares, P. Martins, V. Batista, C. Renato, L. Almeida, N. Stohr, "*JAMES: A Platform of Mobile Agents for the Management of Telecommunication Networks*", in Proceedings of IATA'99, Stockholm, 1999

[5] L. Silva, G. Soares, P. Martins, V. Batista, L. Santos, "*Comparing the Performance of Mobile Agent Systems: A Study of Benchmarking*", Computer Communications, Volume 23, Issue 8, April 2000

[6] P. Simões, L. Silva, F. Boavida, "*Integrating SNMP into a Mobile Agent Infrastructure*", in Proceedings of DSOM'99, Zurich, 1999

[7] P. Simões, L. Silva, F. Boavida, "*A Generic Management Model for Mobile Agent Infrastructures*", in Proceedings of SCI'2000/ISAS'2000, Orlando, July 2000

[8] D. Tennenhouse, J. Smith, W. Sincoskie, D. Weatherall, G. Minden, "*A Survey of Active Network Research*", IEEE Communications Magazine, Vol. 35, 1997

[9] D. Gavalas, D. Greenwood, M. Ghanbari and M. O'Mahony, "*Advanced network monitoring applications based on mobile/intelligent agent technology*", Computer Communications, Volume 23, Issue 8, April 2000

[10] L. Silva, V. Batista, P.Martins, G. Soares, "*Using Mobile Agents for Parallel Processing*", Proceedings of DOA '99 - Distributed Objects and Applications, Edinburg, 1999

[11] D. Levi, J. Schonwalder, "*Definitions of Managed Objects for the Delegation of Management Scripts*", RFC 2592, 1998

[12] M. Rose, "*The Simple Book - An Introduction to Management of TCP/IP-based Internets, 2nd Edition*", Prentice-Hall International Inc., 1994

[13] D. Kotz, R. Gray, "*Mobile Agents and the Future of the Internet*", in ACM Operating Systems Review, 33(3), 1999

[14] C. Szyperski, "*Component Software, Beyond Object-Oriented Programming*", Addison-Wesley, 1998.

[15] J. Saltzer, D. Reed, D. Clark, "*End-To-End Arguments in System Design*", ACM Transactions in Computer Systems, Vol. 2, No. 4, November 1984

[16] R. Orfali, D. Harkey, J. Edwards, R. Crfali, "*Instant CORBA*", John Wiley & Sons Inc., 1997

[17] D. Kotz, R. Gray, S. Nog, D. Rus, S. Chawla, G. Cybenko, "*AGENT TCL: Targeting the Needs of Mobile Computers*", in IEEE Internet Computing, vol. 1, 1997

[18] A. Sahai, C. Morin, "*Mobile Agents for Enabling Mobile User Aware Applications*", in Proc. Autonomous Agents 98, Minneapolis, USA, 1998

[19] P. Marques, L. Silva, J. Silva, "*A Flexible Mobile Agent Framework for Accessing Information Systems in Disconnected Computing Environments*", to appear in Proc. Third International Workshop on Mobility in Databases and Distributed Systems MDDS'2000, Greenwich, UK, September 2000

[20] P. Marques, L. Silva, J. Silva, "*Addressing the Question of Platform Extensibility in Mobile Agent Systems*", to appear in Proc. International ICSC Symposium on Multi-Agents and Mobile Agents in Virtual Organizations and E-Commerce (MAMA'2000), Wollongong, Australia, December 2000

[21] Sun Microsystems, *Infobus 1.2 Specification*, Sun Microsystems, 1999

[22] P. Marques, L. Silva, J. Silva, "*Building Domain-Specific Mobile-Agent Platforms from Reusable Software Components*", to appear in Proc. 2000 International Conference on Software, Telecommunications and Computer Networks (SoftCom'2000), Split and Dubrovnik (Croatia), Trieste and Venice (Italy), October 2000