An Open Secure Mobile Agent Framework for Systems Management

Paolo Bellavista and Antonio Corradi

Dipartimento di Elettronica, Informatica e Sistemistica
Università di Bologna
Viale Risorgimento, 2 - 40136 Bologna - ITALY
Ph.: +39-051-6443001 - Fax: +39-051-6443073
e-mail: {pbellavista, acorradi}@deis.unibo.it

Cesare Stefanelli

Dipartimento di Ingegneria
Università di Ferrara
Via Saragat, 1 - 44100 Ferrara - ITALY
Ph.: +39-0532-293831; Fax: +39-0532-768602
e-mail: cstefanelli@ing.unife.it

An Open Secure Mobile Agent Framework for Systems Management

Abstract

The Mobile Agent (MA) technology is gaining importance in the distributed management of networks and services for heterogeneous environments. MA-based management systems could represent an interesting alternative to traditional tools built upon the client/server model, either SNMP- or CMIP- based. The acceptance of MA solutions for management is currently limited by two main requirements: the need of interoperability and the request for security. Without security, management systems can not suit global untrusted environments, such as the Internet; without interoperability, they can not interact with existing tools and legacy systems. The paper describes an MA-based management system which considers security and interoperability as the two main design objectives. It is an open management framework that grants interoperability by providing compliance with CORBA, the most diffused standard in the area of Object-Oriented components. In addition, it is based on a thorough security model and provides a wide range of tools and mechanisms to build and enforce flexible security policies.

Keywords: Network and Systems Management, Web-based Management, Interoperability, Security, CORBA, MASIF, Java.

Suggested Running Head:
An Open Secure Mobile Agent Framework for Systems Management

1 Introduction

The increasing complexity of global distributed systems, from a set of resources connected by network infrastructures to a set of network-centric coordinated services, has motivated the evolution of traditional management models. These models are generally shaped after the client/server (C/S) model, which applies to both IETF SNMP [1] and OSI CMIP [2] standards. The interaction between managers and agents is usually decided in a static way, where roles are assigned *a priori* and clients and servers can only exchange predefined data. In the basic C/S solution, the key role of the central manager can cause inefficiencies and overhead: the need of accuracy in control and the large size of controlled systems can induce an intolerable traffic of information exchange. In addition, the traffic tends to increase when there are some anomalous events on the managed resources: the manager is likely to be overwhelmed and cannot keep up with her duties.

New programming paradigms based on mobile entities [3] have suggested novel approaches to network and systems management more suitable for the increasing complexity of current global and heterogeneous systems [4]. The basic idea is that the network itself can play an active role in service provision, evolving from a simple transport layer toward a coordinated and distributed processing environment [5,6]. The new management approaches facilitate the delegation and automation of control actions, thus reducing network load and relieving the central manager duties.

Among the new paradigms, the Mobile Agent (MA) one is obtaining more and more interest in the area of network, systems and service management [7,8]. The adoption of the MA technology is currently limited by the lack of security and interoperability. New management environments should operate in global and untrusted environments, and should provide flexible security mechanisms to grant the level of security required by different organizations. In addition, they should interoperate with legacy systems, based on either SNMP or CMIP, and even with emerging management tools that are built upon the Common Object Request Broker Architecture (CORBA) [9] or upon the MA technology.

This paper describes a secure and open MA environment for the management of networks, services and systems, called **MAMAS*** (**M**obile **A**gents for the **M**anagement of **A**pplications and **S**ystems). Security and interoperability have been taken into account since the first phase of the project: without either of them, the MA technology cannot be accepted because it lacks of either the *closing property* or the *opening* one. The *closing property* is the possibility of constraining the system in such a way to identify and exclude any malicious attempt, while the *opening property* is the necessity of overcoming system boundaries in order to access to any necessary external component and to allow any external recognized usage. The *closing property* is granted by *security* policies and the *opening* one by *interoperability* considerations.

The rest of the paper is structured as follows. Section 2 is a brief overview of traditional and emerging solutions in the management area. Section 3 depicts MAMAS general architecture and section 4 presents several implemented MAMAS agents and their functionality. Section 5 gives some details about the MAMAS implementation, especially from the point of view of security and interoperability.

* The MAMAS environment and its MA support are available from: `http://www-lia.deis.unibo.it/Software/MA/`

2 Traditional and Emerging Management Solutions

Traditional solutions to systems management based on SNMP and CMIP protocols have been largely used in many commercial products, but have shown some of their limits. Several environments attribute a central role to one manager that controls the state of the distributed system: OpenView [10], NetSP, SunNet [11], give the operator full information about system resources, by using SNMP. A limited possibility of automated actions is available: most of the installations require the presence of an operator capable of real-time decisions. In addition, the lack of security features makes not possible to achieve the *closing property*.

These tools are good design examples, but they are based on the C/S model of SNMP and CMIP: a central manager provides a user interface to the system administrator and interacts with remote managers (called agents in OSI and IETF terminology) that run on network nodes and manage remote access to their local information base. The central manager interacts with remote managers via a fine-grained C/S management protocol. This form of interaction can introduce the so called micro-management problem, caused by the high traffic around the central manager node, already overloaded because it must perform the needed computation.

There are several projects for the management of Unix-based environments that, instead of using a standard protocol, start from scratch to design the management features [12]. The goal is to furnish one central operator a view of the current system state, with a limited possibility of automatic and manual intervention. These projects create a scenario for rapid development: typical implementation languages are Perl [13] and Tcl/Tk

[14]. While the stress is on the rapid application development side of the proposals, the usage of shell languages make difficult to give an answer to the *closing property*.

In the last few years, several researches have examined the intrinsic nature of distributed systems, i.e., their capacity of hosting mobile and dynamic entities. Different answers can come from new execution models [3,15], which can contribute with new solutions to systems management [16,17]. Management by Delegation (MbD) represents a clean effort toward decentralization and increased flexibility of management functionality [4,18]. MbD dynamically distributes network management components to “elastic” remote managers, that can learn new modes to locally handle resources. Again, the security requirement has been neglected. The goal of the Active Networks (ANs) project at MIT is to build programmable networks [5]. In ANs, applications can inject customized programs into remote nodes to perform ad hoc computations on the packet content. By pushing programmability down to the network layer, ANs have already shown their capacity of achieving significant results in terms of performance, scalability and QoS provision [19]; however, there are typical management issues, such as security, that seem not suitable to be solved with a network layer approach [20]. Intelligent Networks (INs) are based on a similar approach, but IN service deployment is performed with the goal of permitting the expression of the service control in an outband channel, separated from the bandwidth devoted to service processing [21]. INs, at least in their first realizations, seem to neglect the *opening property*.

The MA technology is suitable for the implementation of both ANs and INs architectures [8,22] because agents can move to managed resources while in execution and according to specific run-time conditions, and can even be installed for a limited time duration.

A large part of the recent research in the systems management field has been dedicated to CORBA. CORBA is used as a way to provide high abstraction levels for resources and services [23]; CORBA gateways permit protocol translation and service emulation for interworking with SNMP and CMIP legacy systems [24,25]. In addition, CORBA can act as the middleware framework to build new integrated management environments [26].

3 MAMAS: an MA Environment for Systems Management

Many organizations face the problem of managing their distributed and heterogeneous systems composed of a large number of multi-vendor computing resources. In addition to the interoperability requirement to face heterogeneity, the complexity of the management problem stems from many other factors, such as the adequate level of security, or the integration of different administration schemas.

From the network architecture point of view, one organization may be composed of several departments, even geographically distributed. When different departments of the same organization have to communicate within the Internet, the system should grant the same levels of security and QoS as in intranet communication. In an administration perspective, while simple traditional approaches tend to identify a central administrator in charge of managing all resources, many organizations are better suited to distributed and coordinated strategies, with several administrators with different responsibilities for different resources.

One of the most promising technology for dealing with the complexity of an open network-centric scenario is the MA one, based on the idea of moving execution entities to the part of the system to be controlled, overcoming the restriction of the traditional C/S model of interaction.

The paper describes MAMAS, an open and secure MA environment for the management of networks, services and systems. MAMAS can adapt to very different organizations, in terms of both network architectures and administration policies. It can be configured for a range of architectures, from one single LAN to the interconnection of several LANs. In addition, MAMAS can be used by a single administrator, but also for distributed and coordinated strategies with teams of administrators. MAMAS mobile agents can migrate at run-time and can face dynamic situations where any static decision can become inefficient. Mobile agents act on behalf of administrators, can be configured to monitor the whole system and can introduce dynamic and automatic correction actions to several management problems.

3.1 MAMAS Architecture

MAMAS faces the complexity of management problems by introducing two abstractions, one at the physical level, the other at the logical one: the *network locality* models physical resources, the *administration locality* represents the responsibility scope of system administrators (see Figure 1).

The *place* abstraction, where agents can execute, is the network locality that represents the physical machine. The *domain* abstraction encloses a set of places; it typically represents a LAN and includes a default place that embeds the *gateway* abstraction responsible of the interconnections among different domains.

MAMAS models any kind of administration locality by grouping resources in relationship with administrators; one administration locality determines who is in charge of managing actions, which permissions are granted to her and to what extent, thus embodying an explicit model of trust for management purposes [20]. Several administration localities may overlap, to model the joint administration of resources by several managers, even with different permissions over the same resources.

The MAMAS ability of modeling both network and administration localities offers flexibility in implementing several management policies, from a simple centralized management scheme to distributed and coordinated strategies with several administrators in charge of controlling multiple network localities, i.e., networks interconnected by gateways and firewalls. At the moment, MAMAS does not give the possibility of hierarchies of domains: this architecture constraint permits an efficient implementation of the locality abstractions and does not introduce any limit in real modeling capacity in the area of systems management.

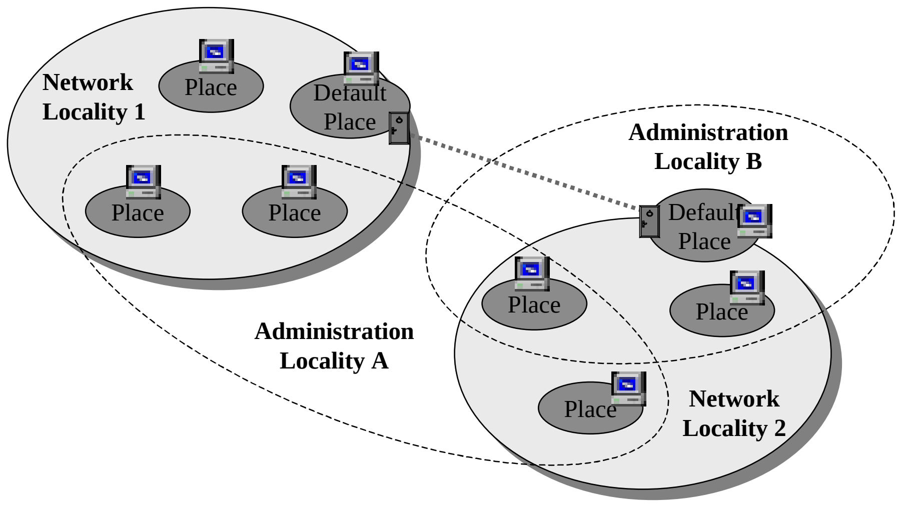


Figure 1. MAMAS abstractions for network and administration localities.

3.2 MAMAS Security

The mobile agents employed in systems management can attempt very sensible operations. They usually have system duties and the risk of an incorrect action, due to erroneous or malicious reasons, is intolerable. The *closing property* should be considered a key property of management environments.

In MAMAS, security is integrated at any system layer, because only this pervasive approach can achieve a level of quality different from the minimal one obtained by considering the security requirement a posteriori. The management environment makes available a wide range of security mechanisms and tools, in order to prevent any malicious action: it provides protection for both hosts and agents, and ensures integrity and secrecy for agent migration and communication. In particular, place protection is achieved by verifying the integrity of incoming agents, by authenticating them, and by controlling agent actions on resources.

The security infrastructure is based on layered security policies: the definition of different network locality abstractions allows to enforce security policies in which actions are controlled at both place and domain levels. The domain defines a global security policy which imposes general authorizations and prohibitions; each place can restrict the domain-level set of permissions.

Figure 2 depicts a concrete scenario in which MAMAS agents move from place to place: any inter-domain movement is solved as a movement from the default place of the sender to the default place of the receiver domain. The identification of the principal responsible for one agent derives from the agent signature [27]. When entering the place, the authentication check verifies agent signatures and the authorization phase grants agents the right to access local resources according to their responsible administrators. To ensure secrecy, management agents can be encrypted when traversing an insecure path; the detection of any agent modification occurs via the use of secure hash function verified in the integrity check phase [27].

Let us note that security has a cost in terms of performances. For this reason, the MAMAS security infrastructure offers a wide range of security mechanisms and policies: administrators can decide a suitable trade-off between security needs and required performances, tailored to specific cases. Agents from internally trusted domains (e.g., the intranet of one organization) can directly proceed to the authorization check, while agents from untrusted ones are subject to all the secrecy, integrity, authentication and authorization steps [28] (see Figure 2).

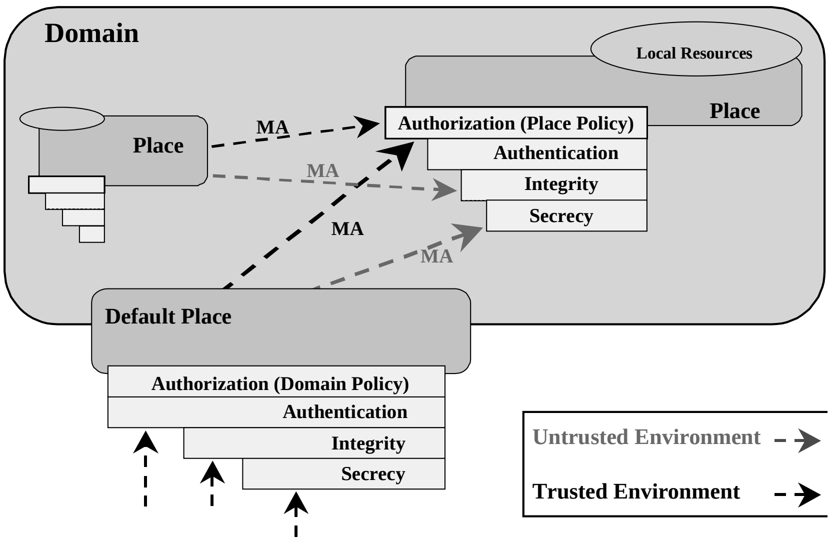


Figure 2. Different security checks for agents in trusted/untrusted environments.

3.3 MAMAS Interoperability

Security is an important property of management environments for global and open systems, but we consider fundamental also the *opening property*. First, management tools should interact with other external components, in order to require/offer services and facilities; secondly, they should be able to command and control any possible resource, independent of the supported management protocol; finally, they should integrate with legacy management systems.

Interoperability is greatly simplified when standardization has already imposed or produced large acceptance of interoperable interfaces and guidelines. The principal recognized effort in the standardization of object-oriented components is CORBA, promoted by the Object Management Group (OMG). The CORBA standard has aroused increasing interest, permitting users to make dynamic communications in open environments, to enclose legacy systems, and to achieve very advanced name services. Apart from these basic features, the OMG has gone on in defining a standard framework to solve the problems of the distributed systems design area: the need of additional functionality has led to the specification of standard interfaces for the CORBA Object Services and Common Facilities (in particular, in the context of systems management, the specifications for the Security Services (SSs) [29] and for the Systems Management Common Facilities (SMCFs) [30]).

At present, architectures of management systems tend to move from the OSI manager-agent model to the distributed object-based one. However, SNMP, with its general acceptance in the Internet environment, and CMIP pervasively used in the telecommunications industry to manage equipment networks and services, are likely to remain the most diffused standards for some time to come, because of the amount of investment in existing applications. CORBA can play a very significant role for the integration between these traditional management technologies and the emerging ones.

The design of the MAMAS environment has aimed to achieve a wide interoperability with different management frameworks to ensure the *opening property*. The integration with legacy systems is considered a fundamental requirement. It is straightforward with Java wrappers to encapsulate existing protocols and to provide an MAMAS-specific *application gateway*, according to the definition in [31], toward either CMIP or SNMP.

However, we have chosen an open way to achieve interoperability via compliance with CORBA. The choice stems from many reasons: first of all, many management environments can employ CORBA [32,33], and much research work has recently investigated the possibility of integration between CORBA and the management of legacy systems [23-26]. In addition, our solution to interoperability permits to exploit all the functionality offered by the CORBA middleware, in particular CORBA SSs and SMCFs. MAMAS also answers the issue of interoperability among different MA programming frameworks by considering full compliance with the OMG Mobile Agent System Interoperability Facility (MASIF) [34], an emerging standard to support agent mobility and management, which integrates CORBA distributed objects and mobile agents. The more MA-based proposals for network and systems management are becoming common, the more the role played by MASIF is gaining relevance [6,8,35].

MASIF proposes a standardization for agent and agent system names, for agent system types and for location syntax; it defines two interfaces (`MAFAgentSystem` and `MAFFinder`) with the typical sets of functionality respectively for agent management and for agent tracking. Agent management allows an external system to control agents of a MASIF-compliant MA system (actions such as suspending/resuming/terminating agents or moving agents between type-compatible MA platforms). Agent tracking permits the tracing of agents registered with `MAFFinders`, which essentially provide an MA name service, since the CORBA Naming Service is not suitable for entities like agents, which are mobile by nature.

The implementation of the *opening property* gives MAMAS the capacity of interoperating in different contexts (see Figure 3):

1. an MAMAS application can perform management operations on legacy systems via third-party CORBA gateways to SNMP/CMIP;
2. an MAMAS application may call external CORBA objects, either CORBA Services/Facilities or other systems management frameworks which offer a CORBA interface (MAMAS agents as CORBA clients);
3. an MAMAS application may register its interface on an ORB and offer the implemented services to any recognized external CORBA client (MAMAS agents as CORBA servers);
4. any external entity, MA-based or not, may ask MAMAS agents for agent management and tracing services defined by the MASIF standard;
5. mobile agents can be moved between different type-compatible MA-based management environments compliant with MASIF.

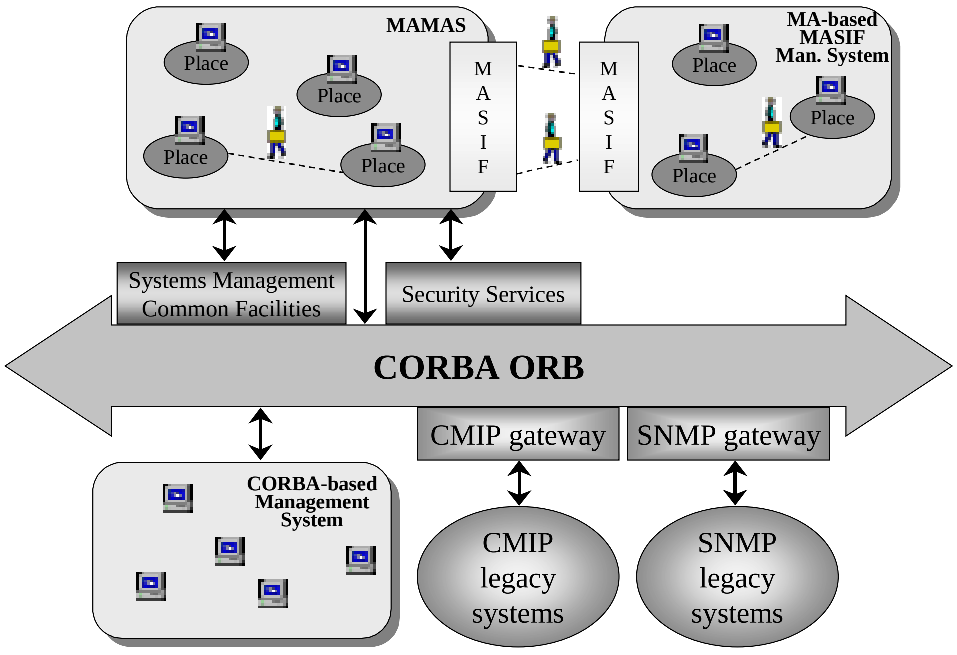


Figure 3. MAMAS interoperability via CORBA.

Opening the system to the external world requires the interaction of MAMAS security mechanisms and policies with the ones of different environments. CORBA SSs propose a solution framework in which the traditional security mechanisms can be employed to implement the needed security policies.

4 MAMAS Mobile Agents Components

In MAMAS, agents act on behalf of administrators and fulfil administration needs by moving and executing on different nodes. MAMAS makes possible to delegate management actions to mobile agents, relieving the administrator duty and automating control actions. Any administrator can implement her policy by using mobile agents. System policies can be propagated at run-time, with no need to shutdown the system: a new agent can bring the new policy everywhere.

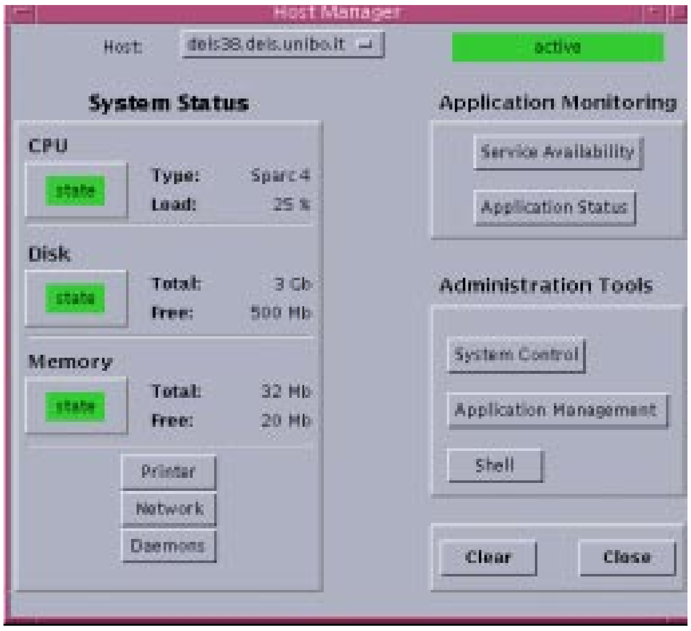
The MAMAS environment already provides a set of mobile agents for systems management. In addition, it is easy to tailor new agents to new specific administration needs. The following list gives a few examples of functionality already available via MAMAS agents:

- monitoring the state of the distributed system;
- controlling and coordinating replicated resources;
- dynamically installing new services;
- helping in the configuration of any new or reinserted node;
- shutting down the whole system by ensuring a minimal survival service level.

As an example of an MAMAS agent, let us consider the monitor agent that reports to one administrator the information about the state of the whole distributed system. The agent gives the situation of each node in terms of system and application indicators. In addition, it gives also network information, such as the collision rate (see Table 1). Figure 4 shows the GUI of the monitoring tool when reporting the state of a specific host. The administrator is given the situation of a node in terms of system and application factors, for instance, the state of physical resources, such as CPU and disk occupation. The application monitoring part permits to create and send new agents where requested in the system.

System indicators		**Application indicators**
CPU load	collision rate	service availability
file system occupation	network connectivity	program versioning
swap space available	firewall state	application processes situation
daemon processes situation	...	local agent states
printers status	...	...
...	...	...

Table 1. Some of the figures available in MAMAS.

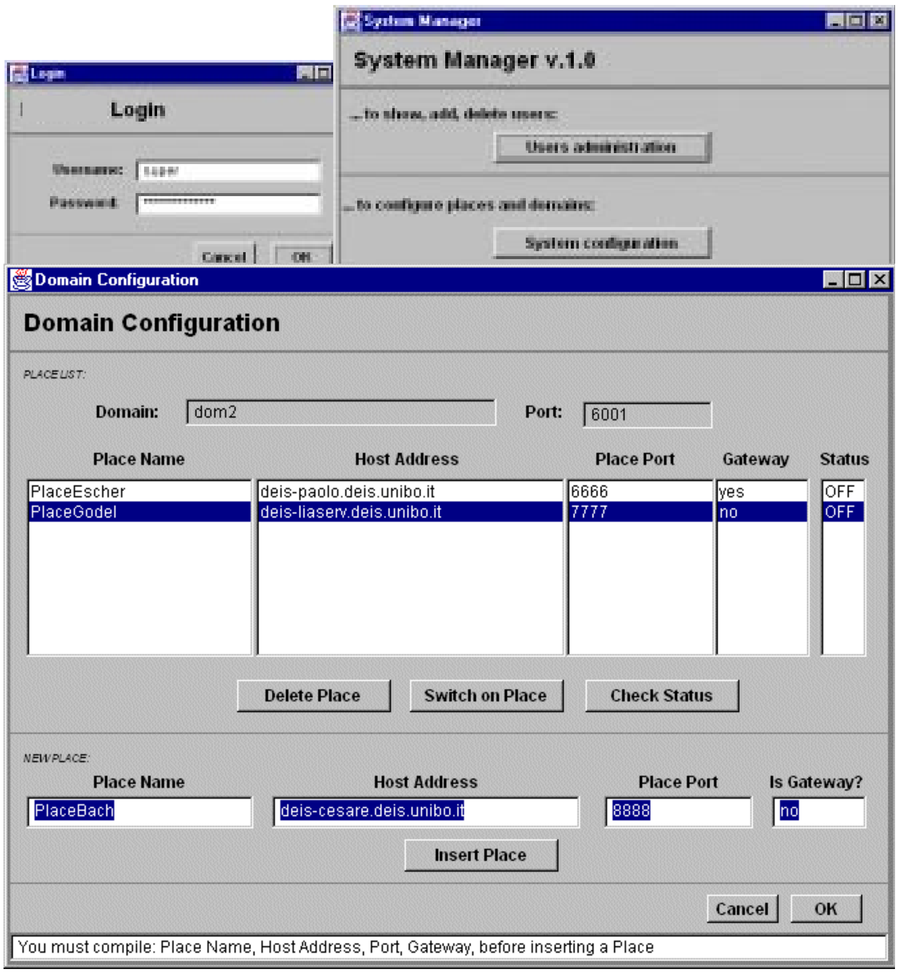


Host: deis38.deis.unibo.it
active

Type:	Sparc 4
Load:	25 %
Total:	3 Gb
Free:	500 Mb
Total:	32 Mb
Free:	20 Mb

Figure 4. The MAMAS graphical interface to monitor distributed systems.

Any administrator can access to the MAMAS environment via a Web browser. In fact, MAMAS provides a user-friendly graphical interface to operate directly on the system. For example, Figure 5 shows how an administrator can control the initial configuration of places and domains, and its modification at run-time. After the authentication phase, she is authorized to perform different management operations, depending on her permissions. She can add/delete/show users, modify security and administration policies, dynamically insert new resources and behavior in the managed system, and perform more complex and coordinated tasks such as multiple parallel installation of new services on a run-time dependent subset of hosts.



Username: super
Password: ***************

PLACE LIST:

Domain: dom2 Port: 6001

Place Name	Host Address	Place Port	Gateway	Status
PlaceEscher	deis-paolo.deis.unibo.it	6666	yes	OFF
PlaceGodel	deis-liaserv.deis.unibo.it	7777	no	OFF

NEW PLACE:

Place Name	Host Address	Place Port	Is Gateway?
PlaceBach	deis-cesare.deis.unibo.it	8888	no

You must compile: Place Name, Host Address, Port, Gateway, before inserting a Place

Figure 5. Some MAMAS configuration interfaces.

5 MAMAS Implementation

MAMAS has been implemented as an application layer over our MA programming framework [28], that we have developed from scratch in order to fully support the fundamental properties of flexibility, interoperability and security.

The MA support has been realized in Java JDK 1.2 beta2 [36,37], chosen to exploit Java easy integration with the Web scenario, its intrinsic portability to heterogeneous platforms and its security model which permits to express fine-grained control in resource access. The object-oriented nature of Java has helped the design of the MA support, first, and of MAMAS environment, then: the encapsulation principle suits the abstraction needs of both agents and resources; the classification principle makes possible to inherit behavior from already specified components; garbage collection and error management simplify writing robust code. In addition, Java and CORBA can be seen as two object infrastructures that integrate each other fairly well with complementary goals, the former with implementation transparency, the latter with network transparency [38].

A well-known problem of Java is the lack of full mobility support, especially for Java threads: it is not possible to save the whole state of a thread before its migration to a different node to reestablish its execution there. This restriction can be overcome either by modifying the Java Virtual Machine (JVM) or by providing a new operation at the application level. We have chosen the second solution to preserve portability. Finally, Java is motivated by the fact that the JVM is expected to be available on every network component in the near future, thus providing a universal set of homogeneous resources. Many vendors, including Cisco and 3-Com, have already integrated Java into some of their products.

Interoperability in MAMAS is provided by a software add-on, called *CORBA Bridge*, which extends the functionality of the agent places in charge of interoperating (see Figure 6). The CORBA Bridge extension is composed of two distinct modules: the first one (*CORBA C/S*) simplifies the design of MAMAS applications as CORBA clients/servers; the second one (*MASIF Bridge*) provides the MASIF functionality. The MASIF Bridge implements the `MAFAgentSystem` class with the basic functionality for agent mobility among heterogeneous MA platforms (`create_agent()`, `fetch_class()`, `receive_agent()`, `get_MAFFinder()` methods), and the `MAFFinder` class with the methods for agent naming specified in the MASIF interface.

Since MASIF implementation imposes a heavy load on the execution place, our guideline is that only the default place in the domain should be extended with MASIF Bridge; on the other hand, the CORBA C/S module is lightweight, and many places in the same domain may use it to access to the CORBA bus. Any MAMAS agent, resident on a CORBA C/S place, is able to act as a CORBA client/server both through static invocation/registration (IDL stub/skeleton) and dynamic ones (DII/DSI). Our implementation, based on Inprise VisiBroker ORB [39], exploits only the portable functions provided by its Portable Object Adapter [9], to avoid any closure within a particular ORB realization. Even if there is no conceptual problem for a mobile agent to register itself as a CORBA server, we currently grant this possibility only to MAMAS agents that do not migrate during their lifetime (stationary agents) in order to avoid the overhead of registering/unregistering at the CORBA Naming Service at any migration.

We are currently testing the MAMAS applications capacity of interoperating with CORBA-based management environments such as Tivoli [33], and with mobile agents from other platforms, such as Grasshopper [40], the unique commercial MA system that has already implemented the MASIF interface.

From the security point of view, the MA support provides a wide variety of mechanisms and tools to grant security to the MAMAS entities in a flexible way. Security policies associate agents and administrators with the permissions to operate on local resources; policies are defined at both the place and the domain levels and are stored in encrypted files. MAMAS also provides a graphical management tool for domain/place administration that can add/delete/modify entries in the policy file at run-time, with no need to suspend execution.

The protection of execution environments from malicious agents is solved by authentication and access control mechanisms, through widely accepted certification protocols (DSA algorithm and X.509 certificates). Integrity and secrecy for message exchange and agent transfer are achieved with standard cryptographic mechanisms: it is possible to choose between the DES channel encryption (secret keys exchanged through either the RSA or the DH protocol) and the SSL protocol solution (provided by the iSaSiLk package [41]).

The *opening property* of MAMAS, obtained by its compliance with CORBA and MASIF, forces to address the new security threats introduced by interoperability. On the one hand, sending/receiving CORBA requests/replies requires security techniques to ensure privacy and integrity for exchanged CORBA messages. On the other hand, the possibility for MAMAS agents to act as CORBA servers and for MAMAS places to host MASIF-compliant agents requires mechanisms to authenticate clients and agents, and to control and audit resource access. MAMAS addresses these new security problems by following the guidelines of providing solutions compliant with CORBA SSs.

In addition to the above described properties of interoperability and security, the MA support provides the name system to ensure the message delivery to agents: it is based on a federation of name servers, one per domain, each in charge of answering the requests generated in its locality. We have adopted an initial ad-hoc name solution, with the goal of integrating with accepted standards, such as DNS and CORBA. Finally, the MA support provides a simple communication protocol to serialize/deserialize agents when they are transferred from place to place (possibly in different domains), while waiting for an agreement for the definition of a standard protocol for agent mobility [34,42].

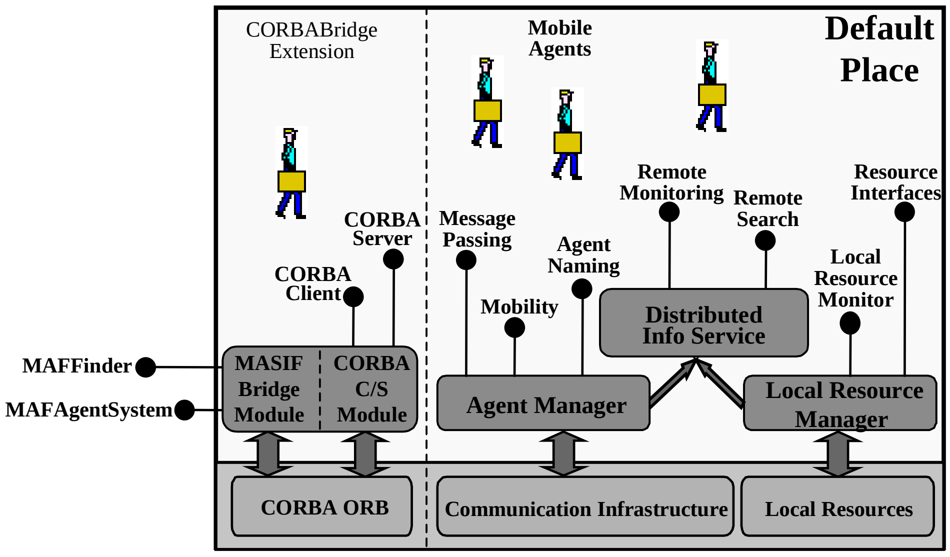


Figure 6. The architecture of a default place in MAMAS.

6 Conclusions

The paper presents MAMAS, a systems management environment based on the MA paradigm. Apart from monitoring the system state and its visualization to operators, it permits the automation of several management actions and the dynamic change of pre-defined system policies. These services are achieved by answering requirements such as flexibility, rapid development and efficiency in presence of hardware and software heterogeneity.

The MAMAS tool stresses two project guidelines to boost acceptance: security and interoperability. Security is integrated at any layer of the project, and the flexibility of the available security tools permits administrators to take into account also the expected performance of services. The decision to strive for interoperability with the CORBA recognized standard is likely to grant the expected durability of the design effort by allowing to interoperate with a wide and enlarging set of management frameworks. At the level of accessibility, MAMAS suggests the usage of Web-based tools. At the level of user services and command, any management function can be embodied into the provided MAMAS components and the use of mobile agents makes possible to dynamically modify their behavior to experiment with different policies.

The Java implementation, apart from the a priori granted portability and rapid prototyping, has been carried out in accord with the possibility of integration within the Web scenario. In addition, CORBA and Java have demonstrated to complement each other well. Their integration can produce the basis to create an open management framework suitable for integrating with legacy systems and capable of easily interacting with new emerging solutions.

Acknowledgements

Work carried out under the financial support of the Italian Ministero dell’Università e della Ricerca Scientifica e Tecnologica (MURST) in the framework of the Project “MOSAICO: Design Methodologies and Tools of High Performance Systems for Distributed Applications”.

References

1. J. D. Case, et al., *Simple Network Management Protocol (RFC 1157)*, DDN Network Information Center, SRI International, May 1990.
2. International Organization for Standardization, *ISO IS9596, Management Information Protocol Specification – Part 2: Common Management Information Protocol*, Jan. 1990.
3. A. Fuggetta, G.P. Picco, and G. Vigna, Understanding Code Mobility, *IEEE Transactions on Software Engineering*, Vol.24, No.5, May 1998.
4. Y. Yemini, and S. da Silva, Towards Programmable Networks, *IFIP/IEEE Int. Workshop on Distributed Systems: Operations and Management*, L'Aquila, Italy, Oct. 1996.
5. D.L. Tennenhouse, et al., A Survey of Active Network Research, *IEEE Communications Magazine*, Vol. 35, N. 1, pp. 80-86, Jan. 1997.
6. M. Breugst, and T. Magedanz, Mobile Agents – Enabling Technology for Active Intelligent Network Implementation, *IEEE Network Magazine*, Vol. 12, No. 3, May-June 1998.
7. A. Bieszczad, B. Pagurek and G. Susilo, Infrastructure for Advanced Network Management based on Mobile Code, *IEEE/IFIP Network Operations and Management Symposium NOMS'98*, New Orleans, Louisiana, Feb. 1998.
8. M. Breugst, L. Hagen, and T. Magedanz, Impacts of Mobile Agent Technology on Mobile Communications System Evolution, *IEEE Personal Communication Magazine*, Aug. 1998.
9. Object Management Group, *CORBA/IIOP Rev 2.2*, OMG Document formal/98-07-01, http://www.omg.org/corba/corbaiiop, Feb. 1998.
10. Hewlett Packard, *Openview User's Guide*, 1992.
11. SunSoft, *SunNet Manager Reference Manual*, 1994.
12. R.A. Finkel, Pulsar: an Extensible Tool for Monitoring Large Unix Sites, *Software Practice and Experience*, Vol. 27, No. 10, pp. 1163-1176, Oct. 1997.
13. L. Wall, and R. Schwartz, *Programming Perl*, O'Reilly, 1990.
14. B. Welch, *Practical Programming in Tcl and Tk*, Prentice Hall, 1997.
15. J.W. Stamos, and D.K. Gifford, Remote Evaluation, *ACM Transaction on Programming Languages and Systems*, Vol. 12, No. 4, Oct. 1990.
16. M. Leppinen, et al., Java- and CORBA-based Network Management, *IEEE Computer*, Vol. 30, No. 6, pp. 83-87, June 1997.
17. M. Baldi, S. Gai and G. Picco, Exploiting Code Mobility in Decentralized and Flexible Management, *Lecture Notes in Computer Science*, No. 1219, Springer-Verlag, Apr. 1997.
18. G. Goldszmidt, and Y. Yemini, Distributed Management by Delegation, *Proc.15th Int. Conference on Distributed Computing Systems*, IEEE Computer Society, Vancouver, British Columbia, June 1995.
19. T. M. Chen, A. W. Jackson (eds.), Special Issue on Active and Programmable Networks, *IEEE Network*, Vol. 12, No. 3, June 1998.
20. R. Oppliger, Security at the Internet Layer, *IEEE Computer*, Vol. 31, No. 9, Sep. 1998.
21. T. Magedanz, and R. Popescu-Zeletin (eds.), *Intelligent Networks – Basic Technology, Standards and Evolution*, Int. Thomson Computer Press, London, June 1996.
22. S. Krause, T. Magedanz, and K. Rothermel, Intelligent Agents: An Emerging Technology for Next Generation Telecommunications?, *Proc. IEEE INFOCOM'96*, IEEE Press, pp. 464-472, 1996.
23. G. Holliman, Q. Kong, and M. Neville, The Integration of CORBA-based Management with OpenView DM, *Proc. Int. OpenView Forum Conf.*, St. Louis, USA, June 1996.
24. S. Mazumdar, and K. Swanson, WEB Based Management - CORBA/SNMP Gateway Approach, *7th IFIP/IEEE Int. Workshop on Distributed Systems: Operations & Management*, L'Aquila, Italy, Oct. 1996.
25. UH Communications ApS, *The UHC CORBA/CMIP Gateway product*, http://login/dknet.dk/˜uh/.

26. Object Management Group, *CORBA-based Telecommunication Network Management System*, OMG White Paper, May 1996.
27. W. Stallings, *Network and Internetwork Security: Principle and Practice*, Prentice Hall, 1995.
28. A. Corradi, M. Cremonini and C. Stefanelli, Security Models and Abstractions in a Mobile Agent Environment, *Proc. WETICE98 Workshop on Collaboration in Presence of Mobility*, Stanford, USA, 1998.
29. Object Management Group, *CORBA Security Services*, OMG Document formal/97-12-22, ftp://www.omg.org/pub/docs/formal/97-12-22.pdf, Dec. 1997.
30. Object Management Group, *System Management Common Facilities*, OMG Document formal/97-06-08, ftp://www.omg.org/pub/docs/formal/97-06-08.pdf, June 1997.
31. P. Kalyanasundaram, A. S. Sethi, Interoperability Issues in Heterogeneous Network Management, *Journal of Network and Systems Management*, Vol. 2, No. 2, pp. 169-193, June 1994.
32. IONA Technologies - OrbixManager, http://www.iona.com/products/sysman/.
33. Tivoli Systems Inc. - Tivoli, http://www.tivoli.com/.
34. GMD FOKUS, IBM Corp., *Mobile Agent Facility Specification*, Joint Submission supported by Crystaliz Inc., General Magic Inc., the Open Group, OMG TC Document orbos/97-10-05, ftp://ftp.omg.org/docs/orbos/97-10-05.pdf, Feb. 1998.
35. M. Breugst, and T. Magedanz, On the Usage of Standard Mobile Agent Platforms in Telecommunication Environments, *Proc. Intelligence in Services and Networks (IS&N'98)*, May 1998.
36. J. Gosling, B. Joy and G. Steele, *The Java Language Specification*, Addison-Wesley, Menlo Park, California, Aug. 1996.
37. Sun Microsystems, *Java Development Kit Version 1.2 (beta 2)*, http://java.sun.com/ products/index.html.
38. S.M. Lewandowski, Frameworks for Component-Based Client/Server Computing, *ACM Computing Surveys*, Vol. 30, No. 1, Mar. 1998.
39. Inprise – Visibroker, http://www.inprise.com/visibroker/.
40. IKV++ GmbH - Grasshopper, http://www.ikv.de/products/grasshopper/.
41. IAIK, *iSaSiLk 2.0,* http://jcewww.iaik.tu-graz.ac.at/iSaSiLk/DOC/isasilk_doc.htm.
42. D.B. Lange, and Y. Aridor, *Agent Tranfer Protocol – ATP/0.1,* IBM Tokyo Research Labs, http://www.trl.ibm.co.jp/aglets/atp/atp.html.

Paolo Bellavista received his Laurea in electronic engineering from the University of Bologna, Italy, in 1997. He is currently pursuing a Ph.D. in computer science engineering at the same university. His research interests include distributed computing, distributed objects, mobile agents, network and systems management, multimedia systems for distance learning. He is member of the IEEE.

Antonio Corradi is an associate professor of computer science at the University of Bologna. His scientific interests include distributed systems, object and agent systems, network management, and distributed and parallel architectures. He received his Laurea in electronic engineering from the University of Bologna and his MS in electrical engineering from Cornell University. ACM, AICA (Italian Association for Computing), and IEEE.

Cesare Stefanelli received his Laurea in Electronic Engineering from the University of Bologna, Italy, in 1992 and the Ph.D. degree in Computer Science in 1996. His research interests are in the area of distributed systems, massively parallel systems and programming environments for parallelism. Currently, he is associate professor of Operating Systems at the Faculty of Engineering of the University of Ferrara.