Multi-Management Schemes for MAF Platforms

Rui Pedro Lopes[1], José Luis Oliveira[2]

[1] Polytechnic Institute of Bragança, ESTiG, 5300 Bragança, Portugal
rlopes@ipb.pt
[2] University of Aveiro, DET, 3810 Aveiro, Portugal
jlo@det.ua.pt

Abstract. Due to the crescent complexity of networks and systems, network management solutions are being pushed towards more distributed and cooperative models. Several specifications promoted by the IETF DISMAN charter already allow strong distribution of management tasks. Unfortunately, they are not adequate to achieve cooperative models. According to recent research, mobile agents provide a good platform to back cooperative models but several lacks are still identified – interoperability between different platforms and SNMP integration. Based on Mobile Agent Facility specification from the OMG, we propose a MIB to better integrate mobile agent models into SNMP frameworks and a GUI tool to monitor and control mobile agents platforms.

1 Introduction

There are several different ways to look at the term mobile agents. Usually it is used to name a mobile code paradigm where processes are allowed to migrate autonomously to another host and resume execution seamlessly. However, in the distributed artificial intelligence field, the term mobile agent refers to an intelligent agent with an additional property: mobility. As such, it encompasses a lot more than simply mobile code. In the context of this paper, and according to several positions about this field [1][2], we use the term mobile agents as the former although allowing a mobile agent to become, at some point, intelligent. We also consider that code mobility in the context of mobile agents also implies state mobility. If the platform does not allow strong mobility – transparent migration of state and code – the programmer can always explicitly preserve the agent state (weak mobility) [3].

Mobile agents require a runtime platform (agent system) that provides them with the resources they need to operate. The agents' platform is typically constructed within a single organization in a closed way, and it usually supports agents that are strictly developed for that specific platform. The absence of standards for an agent system API limits or even eliminates the possibility for interoperability between different vendors' products. This also means that the monitoring and management operations over any kind of agent element are performed with proprietary tools through proprietary methods.

The Mobile Agent Facility (MAF) specification is the first attempt to standardize agent system actions, thus aiming to achieve interoperability between different manu-

facturers' agent systems [4]. MAF is a collection of definitions and interfaces designed as simple and generic as possible to allow future development in mobile agent systems. MAF uses OMG's IDL to declare two interfaces, which are the base to all the operations on the agent system and on the region: the `MAFAgentSystem` and the `MAFFinder`. MAF interfaces specify a common access method to mobile agent platforms. So, any MAF platform can be managed by following the same procedures with the same tools.

Concepts such as agent, place, agent system and region are the base of MAF nomenclature. A place is a logical entity where agents operate. Places can provide differentiated environments for agents depending, for instance, on agent goal, on organizational policies or on access control. The agent system (or agency for some) is the environment for several places, typically corresponding to a host. The region is more or less an organizational classifier that allows mimicking, in this environment, the domain concept of the IP-based LANs.

On the other hand, the SNMP is, so far, the framework of choice from a large number of equipment (hardware, software and technical knowledge), so any near upcoming management architecture have to deal with SNMP standards. Thus, it is natural the coexistence of mobile agent and SNMP.

Why should mobile agents be managed by SNMP? First of all, SNMP is specifically designed for management operations. Moreover, there are available several SNMP based powerful applications and quite a number of installed systems rely on them. This way it is possible to integrate both the mobile agent technology and the SNMP management architecture thus getting the advantages of both [1][5].

Management systems in general and SNMP in particular define a tool, usually called network management station, which provides the user with views of the network current and past state. With this tool, the user may be acquainted of the eventual problems and the current running condition.

The continuous introduction of technological waves in modern networks forces these applications to cope with different models, access methods or security mechanisms. We recently proposed the use of URI (Uniform Resource Identifier) schemes [6] to identify the service tools to use in a common context interface, just like in Internet browsers. The resource is specified in the address field by the URI and, according to its grammar, it calls the appropriate tool for processing and presentation [7]. This approach allows using the same tool for managing different network elements as well as using different access mechanisms. The work presented in this paper follows some previous work related to the management of mobile agents and agent systems through MAF interfaces and SNMP. This work started by defining a specific MIB to convert between SNMP commands and MAF interfaces calls [8].

This paper is structured as follows: section II describes the MAF-MIB structure and its main functionality; section III describes some usage scenarios for the SNMP-MAF association; section IV reveals the URI based mobile agent management station; the paper ends with some conclusions.

2 The Mobile Agent Facility MIB

The Mobile Agent Facility specification (MAF) is a first attempt to standardize agent system, i.e. the middleware that allows agents to stop, to move and to restart execution in another place. MAF uses OMG IDL to declare two interfaces, which are the base to all the operations on the agent system and on the region (some sort of domain in mobile agent technology): the `MAFAgentSystem` and the `MAFFinder`.

The `MAFFinder` interface consists on an access point to a naming service for agents systems, places and agents. Catalogue and lookup are the main functionalities of this interface. The `MAFAgentSystem` interface defines methods and objects that support agent management tasks such as fetching an agent system name and receiving an agent.

Following the MAF specification, the MAF-MIB is structured in two groups[1]: *mafObjects* and *mafLookup*. The *mafObjects* group is related to agent system information, retrieved through the `MAFAgentSystem` interface. The *mafLookup* group allows directory service queries through `MAFFinder` interface (Figure 1).

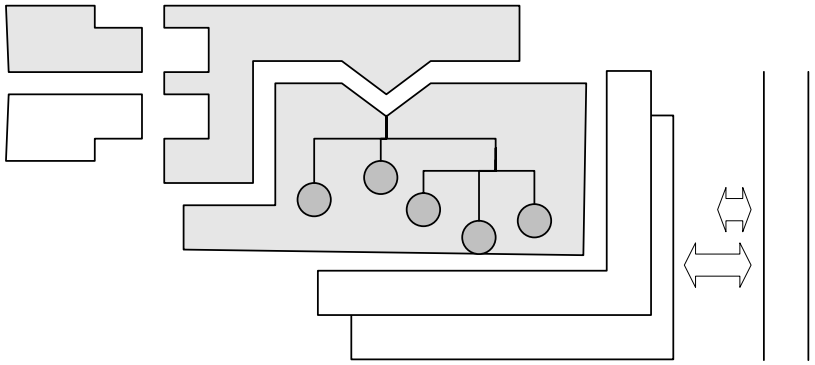


Figure 1 – Architecture of the MAF-MIB SNMP agent.

Related work around SNMP management of mobile agents' platforms suggests using specific MIB to manage specific platforms [9]. The advantage of this approach is the possibility to explore all the characteristics of the platform but it disallows using it with other agent platforms. MAF allows achieving agent platform independence.

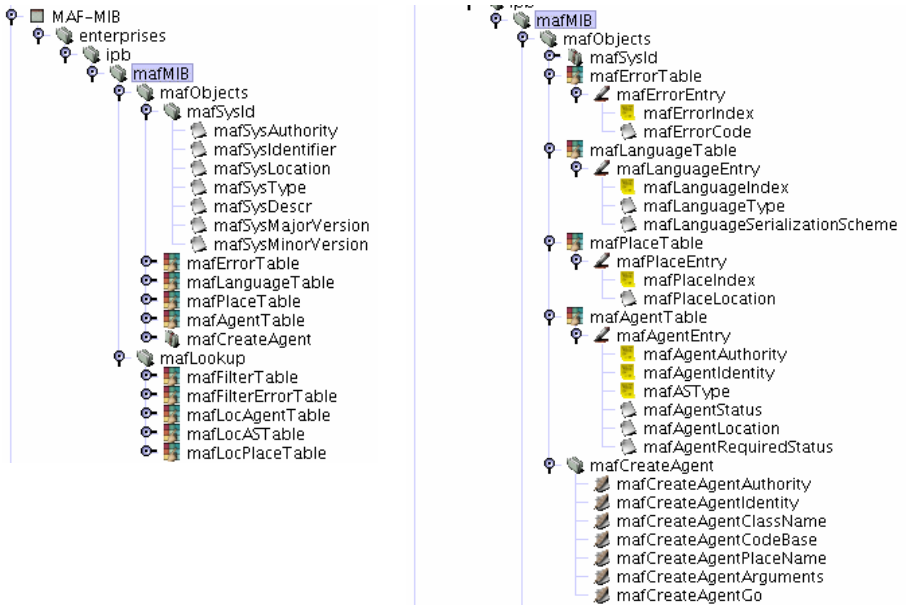
2.1 The *mafObjects* group

This group is mainly based on the `MAFAgentSystem` interface. It starts with a *mafSysId* section, which has information about the agent system authority, identification, type, description and version (Figure 2). While the agents travel, several errors can occur. The table *mafErrorTable*, maintains a record of these errors.

[1] This MIB is available on http://nms.estig.ipb.pt/

SNMP Message processi

Other comm.



a) b)

Figure 2 – MAF-MIB managed objects structure. a) global structure. b) detailed view of the *mafObjects* group.

The agent system supports agents written in several programming languages, such as Java, TCL and others. A manager that wants to know which language is supported may "look" at *mafLanguageTable*.

The following two tables, *mafPlaceTable* and *mafAgentTable*, provide a list of currently existing places and agents in this particular agent system. Moreover, the agent table allows the user to suspend, resume or terminate an agent by setting the required agent status (*mafAgentRequiredStatus*). The index to *mafAgentTable* is composed of the agent authority, its identity and its agent system type. This triplet ensures that every agent has a globally unique name.

The last section, *mafCreateAgent*, holds the necessary objects for creating an agent. The first two objects, *Authority* and *Identity*, will be ignored if the user does not have naming responsibility. In this situation, the authority and identity will be set by the agent system. The *Name* and *CodeBase* provides the agent system with the necessary information to download the code and construct the agent. It thus follows the pull model [10]. The agent will be created on the provided place name (*mafCreateAgentPlaceName*) or on a default place if no place has been provided. It is also possible to define arguments to be passed to the agent constructor (*mafCreateAgentArguments*). After setting the required values, the agent is created by setting the *mafCreateAgentGo* to 1 (`create`).

2.2 The *mafLookup* group

To suspend or terminate the agent the user has to know its location. This information can be retrieved from the *mafLookup* group (Figure 3). It focuses on lookup operations, performed mainly with `MAFFinder` interface method calls and is not restricted to a specific agent system. It can be used to perform lookup operations anywhere up to the region limits.
This service has some special characteristics and, as such, some factors conditioned the MIB structure:

- Search flexibility – Internet search engines use special keys to specify the search terms. The same approach, already used by the agent platform Grasshopper [11], should be followed for increasing the search flexibility.
- Concurrent access – several search operations may be started simultaneously by different management applications. The search system has to deal with this kind of access.
- Simplicity of usage – the search system has to be simple to use.

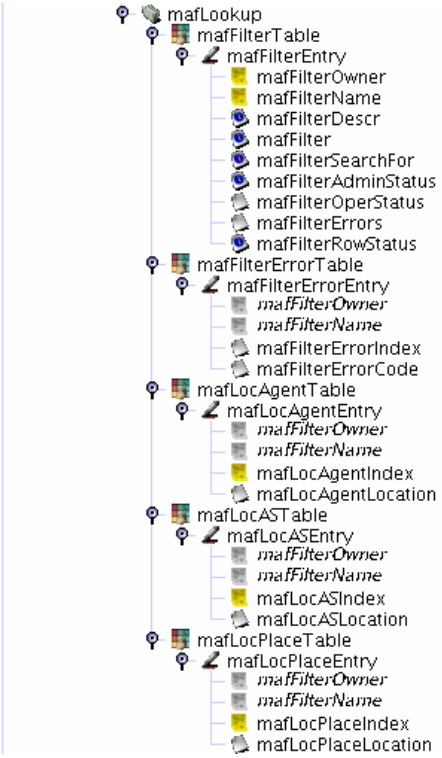


Figure 3 – The *mafLookup* group.

These factors resulted in the definition of search filters associated with a search table. The table *mafFilterTable* allows keeping several search filters, indexed by man-

agement application (*mafFilterOwner*) and by a locally unique name (*mafFilterName*). The filter has a description to detail its function and the type of elements to search for.

The search filter is a character string stored in the column *mafFilter* and its syntax depends on the kind of element to look for. For example, the search for places is only performed by name. There are no other relevant characteristics for this element. For agents and agent systems the filter may support other terms, for example, the supported language or the serialization method. For this reason, the filter syntax allows performing rather complex search operations.

Following the Grasshopper's syntax, the filter may be defined according to the following rules:

```
filter   =  item (|,&) filter /* or, and */
item     =  key comparator value
comparator  =    (=,^,$,~) /*equal, start, end, contains*/
key      =  string /* Name, Codebase, Lang., Serialization…*/
value    =  string
```

The filter "`Name=pingAgent&Codebase~myHost`" is an example of a search operation for an agent with the name "pingAgent" with the code base "myHost".

By setting the column *mafFilterSearchFor*, the search filter may be applied to look for places, agent systems and/or agents. This column is of type `BITS` so it allows one or more types. The filter may be activated (`searching`), i.e., perform a new search, or deactivated (`stopped`) in the column *mafFilterAdminStatus*. Similarly, its current state may be retrieved from the column *mafFilterOperStatus*: `searching`, `stopped`, `finished`. The column *mafFilterRowStatus* creates or destroys filter entries.

According to the search type defined in *mafFilterSearchFor*, the results are retrieved from *mafLocPlaceTable*, *mafLocASTable* and/or *mafLocAgentTable* for places, agent systems and/or agents. These are related to the filter by the index *mafFilterOwner* and *mafFilterName* and can contain several results for a given search operation (*mafLocPlaceIndex*, *mafLocASIndex* and/or *mafLocAgentIndex*).

In case some error occurs due, for example, to a wrong syntax or to any other problem, this is shown in *mafFilterErrorTable*. If the user destroys the filter entry, all the dependent results are also eliminated from the respective table.

3 MAF-MIB Usage Scenarios

The natural evolution of systems such as network management is towards distribution as its size and complexity increases [12]. The increase in size and complexity of modern networks makes centralized management difficult, if not impossible. It is harder to scale, less flexible and less robust. In fact, with increasing network sizes, the computational load as well as the network load on the central manager can grow to a point where it might no longer be able to monitor all agents. Flexibility is related to the network capacity to respond to new conditions and to accept new technology. Because management functions are typically predefined and limited this is harder to achieve with centralized models. A system may become more robust if some kind of

redundancy is achieved, so that it can resist better to error or fault conditions. This is also contrary to the centralized concept.

The Management by Delegation (MbD) model, proposed by Goldszmidt in the beginning of the 90's and fully specified in 1995 uses code mobility to delegate management tasks dynamically over a distributed set of management agents [13]. This approach uses the remote evaluation (REV) approach, where code is sent near the resources and executed locally [3].

The Distributed Management (DISMAN) charter [14] of the IETF followed this approach in the Script MIB [15]. This module receives management applications as scripts and executes them locally. Script MIB implementations may support different languages simultaneously.

The Script MIB provides a SNMP compatible framework thus easing the integration of mobile code approaches with existing management systems. The possibility of association of mobile code with other SNMP MIB modules is a powerful approach and can reduce or even eliminate some of the SNMP deficiencies [16].

Although capable of receiving and executing code, the Script MIB is not well suited to define script based cooperative management systems. It also does not allow process mobility [10]. Several studies, experiments and authors have been supporting the advantages of using mobile agents in different aspects of network management [17-21]. We believe that such tool may be useful and so it should be considered as mechanism to increase the distribution level up to the cooperative model.

The MAF-MIB presents a smooth integration between mobile agents and SNMP by providing a mechanism to launch, monitor and control mobile agents through SNMP. At the current state, the biggest differences between both approaches are related to security. The Script MIB relies on SNMP security mechanisms so every security detail is fully controllable by SNMP. In MAF-MIB, the security is strongly associated with the runtime environment, i.e. the agent system. As a consequence, it is possible to control by SNMP the launching and controlling restrictions but it is not possible to set the runtime security parameters by SNMP. It will have to be done in a platform specific way. Several comparison aspects between the Script MIB and the MAF MIB are condensed in Table 1.

Table 1 – Script and MAF MIBs reflection table.

	Script MIB	MAF-MIB
Integration with SNMP	yes	yes
Association with other MIB modules	yes	yes
Remote control of management tasks	yes	yes
Remote monitoring of management tasks	yes	yes
SNMP security mechanism integration	fully	partially
Process mobility	no	yes
Cooperative model	no	yes
Multi-language support	yes (implementation)	yes (MAF)
Access to intermediate results	By SNMP	Platform dependent
Access to final results	By SNMP	Platform dependent

Another particularity is the access to intermediate and final results. The Script MIB allows accessing results at any time by SNMP. The MAF-MIB only provides tools for searching for mobile agents and controlling its life cycle. For example, adding a

supplementary column (*mafAgentResult*) to the *mafAgentTable* may eliminate this problem.

4 The MAF Explorer

In this context, we developed also a prototype of a GUI tool for the management of mobile agents supporting two simultaneous access methods, SNMP and CORBA, in a single interface. This tool differentiates the access method based on the provided URI. At the moment it accepts two URI schemes, namely, one for SNMP [7] and one for MAF.

We defined a MAF URL with the format:

```
                [scheme:][//host:port][/path]
scheme      = “maf”
host        = IP address or equivalent Internet name
port        = integer
path        = the path registered in the CORBA name server
```

An example for MAF URLs: maf://localhost:1050/CfMAF.

An example for SNMP URLs: snmp://rlopes@jupiter.ipb.pt/?v3.

The common syntax, although semantically different, allows defining a list of URLs (bookmarks or favorites) regardless of the management model. According to the URI scheme, the tool loads the appropriate module and proceeds along with the user commands (Figure 4).

The *MAF Explorer* follows the file manager paradigm with a tree view on the left and the content panel on the right. The previous figure shows a tree view of the mobile agents’ platform resources and was built after setting the URL in the address field (upper right corner). If the user sets an SNMP URL, the tree is modified according to the information from the SNMP agent.

On the right side, it shows the details of the node “CopyAgent”. The grayed labels indicate that the parameter is read only. As an example, the user cannot modify the agent location, although the agent may move autonomously. A black label indicates that the user may also change its value. The agent status may be altered to suspend, resume or terminate its running status.

This tool is used as a “proof of concept” to validate the MAF MIB and yet both MAF and SNMP access methods to agent systems. It has a close correspondence with MAF methods and MAF-MIB objects and it can manage any MAF compatible mobile agent platform either directly, through the CORBA interfaces, or by SNMP, through the MAF-MIB.

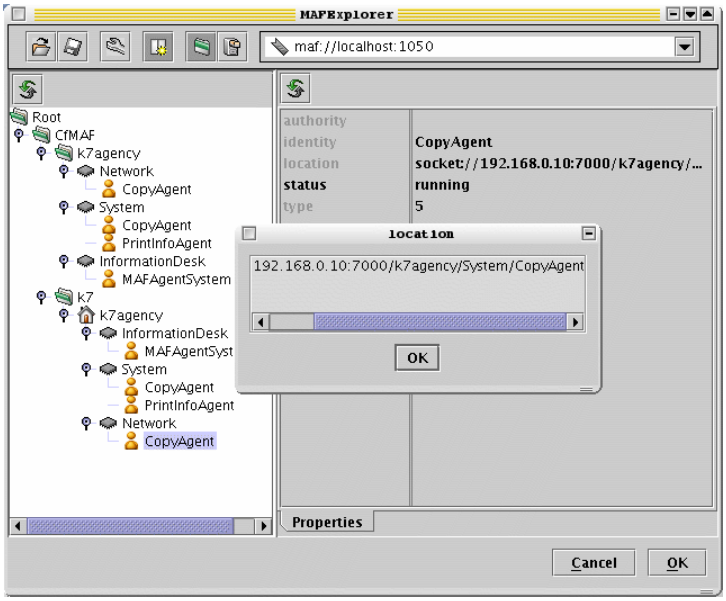


Figure 4 – URI based mobile agent management tool.

5 Conclusions

Mobile agents are supported by a platform, which provides the resources that they need to operate. It is typically constructed within a single organization in a closed way and interoperability between different vendors' platforms is not generally available.

Sensible to this problem, OMG introduced the MAF specification based on CORBA interfaces, a first attempt to standardize agent system actions. MAF defines common access methods to mobile agent platforms, promoting the adoption of a uniform set of management tools and procedures.

We have presented a MAF MIB that provides an SNMP access on MAF platforms. Besides, it establishes common search operations and elementary tools that allow monitoring and modifying mobile agents' life cycle. This MIB may also complement the DISMAN Script MIB to supply a base for developing cooperative network management models and process mobility thus increasing the distribution level.

Finally, in order to validate this concept and also to allow the monitoring and control of MAF platforms and MAF-MIB SNMP agents, a management system was created.

References

1. Goldszmidt G., Yemini Y., "Delegated Agents for Network Management", *IEEE Communications Magazine*, Vol. 36, No. 3 (1998) pp. 66-71.
2. T. Magedanz, T. Eckardt, "Mobile Software Agents: a new Paradigm for Telecommunications Management", *Proc. of the 1996 IEEE Network Operations and Management Symposium – NOMS'96*, Kyoto, Japan, April 1996.
3. A. Fuggetta, G. Picco, G. Vigna, "Understanding Code Mobility", *IEEE Transactions on Software Engineering*, Vol. 24, No. 5 (1998) pp. 342-361.
4. Mobile Agent Facility Specification, Object Management Group, 00-01-02.pdf (ftp://ftp.omg.org/pub/docs/formal/00-01-02.pdf).
5. R. Lopes, J. Oliveira, "On the use of Mobility in Distributed Network Management", *Proc. of the Hawaii International Conference on System Sciences – HICSS*, January 2000.
6. T. Berners-Lee, R. Fielding, U.C. Irvine, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", *Internet Request for Comments 2396*, August 1998.
7. R. Lopes, J. Oliveira, "A Uniform Resource Identifier Scheme for SNMP", submitted to IPOM'2002.
8. R. Lopes, J. Oliveira, "SNMP Management of MASIF Platforms", *Proc. of the IFIP/IEEE International Symposium on Integrated Management – IM'2001*, Seattle, May 2001.
9. P. Simões, L. Silva, F. Boavida, "Mobile Agent Infrastructures: a Solution for Management or a problem to Manage?", *proc. of the Conference of Telecommunications - ConfTele 2001*, Figueira da Foz, Portugal, May 2001.
10. J. Schönwälder, J. Quittek, C. Kappler, "Building Distributed Management Applications with the IETF Script MIB", *IEEE Journal on Selected Areas in Communications, Vol. 18, No. 5*, May 2000.
11. Grasshopper, the agent platform (http://www.grasshopper.de/).
12. J. Martin-Flatin, S. Znaty, J. Hubaux, "A Survey of Distributed Network and Systems Management Paradigms", *Technical Report SSC/1998/024, Swiss Federal Institute of Technology Lausanne*, August 1998.
13. G. Goldszmidt, *Distributed Management by Delegation*, PhD Thesis, Columbia University, New York, December 1995.
14. DISMAN, (http://www.ietf.org/html.charters/disman-charter.html).
15. D. Levi, J. Schoenwaelder, "Definitions of Managed Objects for the Delegation of Management Scripts", *Internet Request for Comments 3165*, August 2001.
16. F. Strauss, "Distribution Models for Network Management", *Project Report* (http://www.ibr.cs.tu-bs.de/projects/jasmin/dismod.ps), June 2000.
17. T. Magedanz, R. Glitho, "Mobile agent-based network and service management", *Journal of Network and Systems Management, vol. 7*, September 1999.
18. G. Knight, R. Hazemi, "Mobile agent-based management in the INSERT project", *Journal of Network and Systems Management, vol. 7*, September 1999.
19. M. Zapf, K. Herrmann, K. Geihs, "Decentralized SNMP management with mobile agents", *Proc. 6th IFIP/IEEE Int. Symp. Integrated Network Management*, Boston, May 1999.
20. P. Simões, L. Silva, F. Boavida, "Integrating SNMP into a Mobile Agent Infrastructure", *Proc. of 10th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management – DSOM99*, Zurich 1999.
21. J. Oliveira, R. Lopes, "Distributed Management based on Mobile Agents", *Proc. of the 1st International Workshop on Mobile Agents for Telecommunications Applications – MATA'99*, Ottawa, Canada, October 1999.