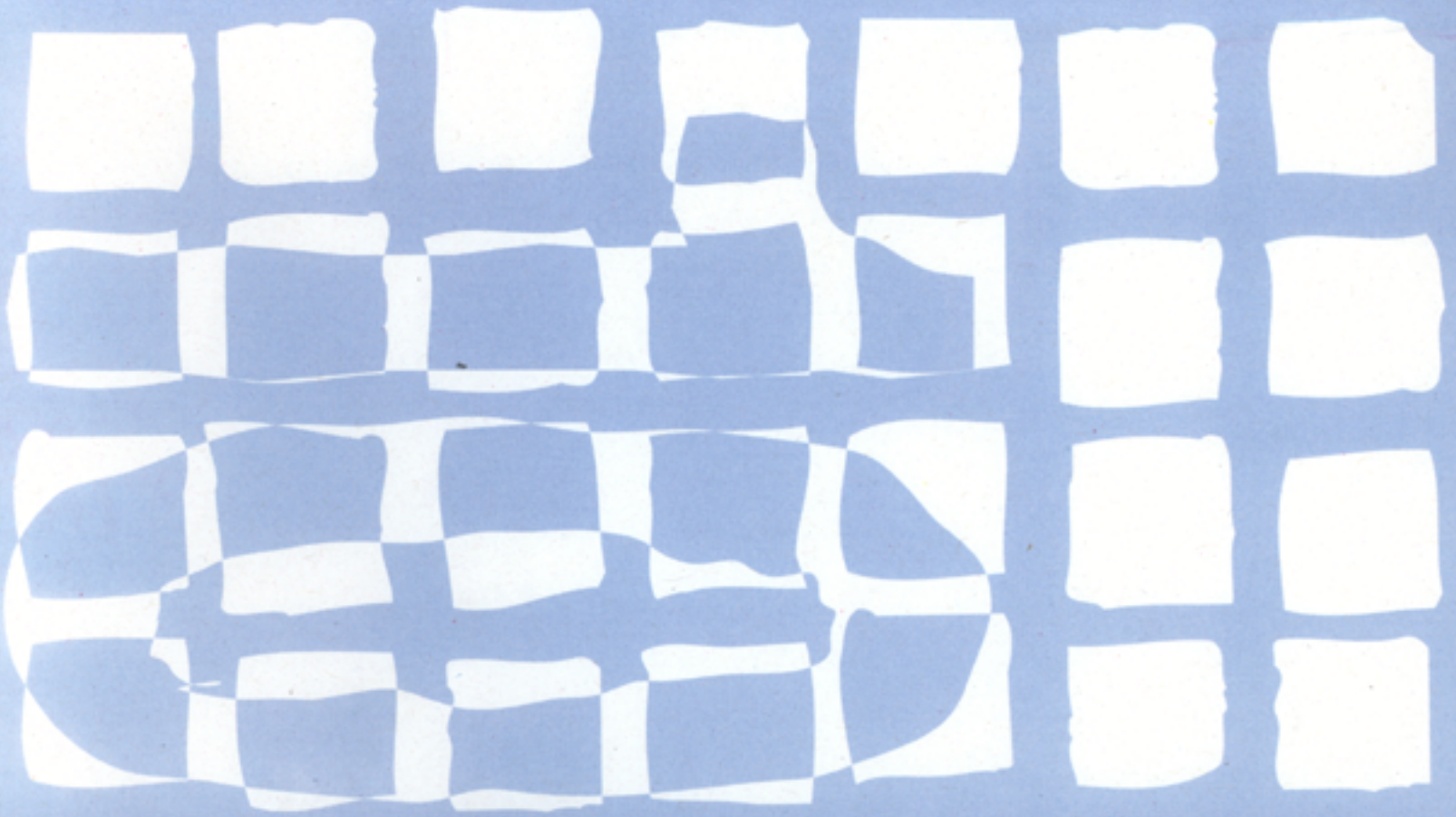

Associação Portuguesa de Sistemas de Informação

Sistemas de Informação

revista da associação portuguesa de sistemas de informação 1999

ISSN: 0872-7031

SOFTWARE AGENTS IN NETWORK MANAGEMENT

Rui Pedro Lopes
ESTiG – IPB,Campus de St. Apolónia – P-5300 Bragança, rlopes@ipb.pt

José Luís Oliveira
Universidade de Aveiro, DET/INESC, P-3810 Aveiro, jlo@ua.pt

Key words: Network Management, Management Information, Agents

Abstract: The globalisation of Internet technology had a strong impact on technology price and availability, which resulted in the emerging of more opportunities and more services in distance learning, electronic commerce, multimedia and many others. From the user perspective, there is a need for more comprehensive interfaces. From the network perspective, the introduction of QoS and the upcoming of new and more complex services led to additional bandwidth and management requirements.
The amount of management information produced by applications, workstations, servers and all kind of network components is increasingly hard to process. All this information can be filtered by an intelligent network management system, capable of proactive management. This paper intents to highlight the role of newer paradigms, such as Software Agents, in Network Management frameworks. The use of intelligent programs that substitutes the user in boring, repetitive or information intensive tasks can help in the resolution of problems such as congestion, reliability, real-time response and many others.

1. INTRODUCTION

Intranet is nowadays a common concept to a large and growing number of enterprises around the world. They are intrinsically open systems, as they constitute an enterprise-wide information utility accessed by anyone, at any time, from anywhere in the world, using any platform. Basically, an intranet is a network connecting a set of computers and devices using Internet protocols, such as the Transmission Control Protocol/Internet Protocol (TCP/IP) and Hypertext Transport Protocol (HTTP), and services (WWW, news groups, e-mail, etc.). In addition, they use open and platform-independent technologies for computing, communications, services and network management. Intranets promote the exchange of information, particularly time critical information, without the need for face-to-face meetings (thus reducing travel expenses), exchange of large printed documents and faxes, which results in substantial savings of time and money. The network management scenario is just an example of extensive use of network resources. As its use grows so do the user dependency and consequently more attention is put on reliability and on quality of service.

This frenetic activity over network resources can easily result in management disorder, due to the number of network elements and to the huge amount of information to process.

Since the last decade international standards consortia have proposed several management models. Nevertheless it remains yet a demand for the appropriate technology that will aid in the self-processing of management raw information.

The second section will present the traditional management architectures based on

the client/server-agent/manager paradigm. The third section will start by giving an overview on software agents and continue with its application on management scenarios. The fourth section presents some advantages of mobile agents relatively to static agents for network management and the fifth section concludes this paper.

2. TRADITIONAL MANAGEMENT

Communication networks management deal with several different aspects: fault, billing, configuration, performance and security (ISO/IEC 7498-4). Management architecture is typically based on the following elements: a) an information model, b) the management model and c) an information transfer protocol.

2.1 Information Model

Each management framework defines different information structures or models. The SNMP framework, for example, defines SMI (Structure of Management Information) (Rose, McCloghrie 1990) based on objects that do not follow the object-oriented paradigm. The OSI management framework defines an object-oriented structure (ISO/IEC 10165-1), where each Managed Object (MO) is a logic representation of physical entities or resources. Other proprietary frameworks, such as WBEM (Web Based Enterprise Management) (WBEM 1996) or JMAPI (Java Management Application Programming Interface) (Sun 1996), define architecture specific information models. All these approaches have a common potential problem: the huge amount of management information involved.

2.2 Management Model

Management models are generally based on distributed instrumentation points (agents) and a centralised information processor (management station) (Figure 1).

The management station issue queries and commands to agents. Each agent gathers working parameter information from network components and publishes it in a virtual information base (MIB).

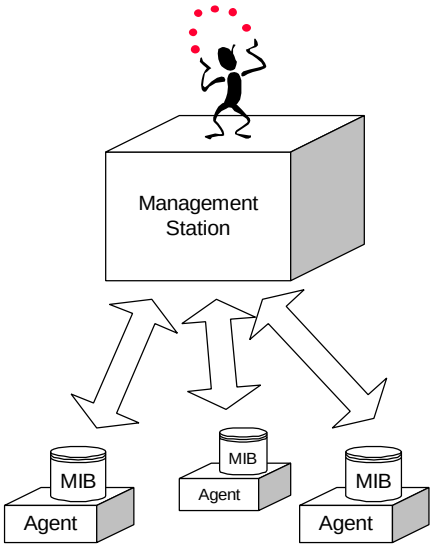

Figure 1 – Generic Network Management Model.

Traditional Network Management Stations (NMS) are simple "windows" to the network, in the sense that they reflect the status of the network itself and of devices attached to it. Most systems provide a graphic description of the network, following some kind of metaphor, such as of the file manager (Vieira & Sá Morais, 1997) or the presentation of topological maps (Oliveira 1995). The graphical description of management information puts any network node or component in reach of a simple mouse click, which gives the user the ability to monitor and configure remote equipment.

Traditional NMS are also capable of some basic automatic tasks. The degree of automation is, unfortunately, very low, resuming to the listening of extraordinary network events, in the form of trap messages. If the message reports some critical event, the system may warn the manager, either by e-mail messages or pager contacts. Usually, received traps are logged for latter inspection.

2.3 Agents

Agents are remote programs responsible for collecting status and working parameters in each network component. Each component (a router, hub, printer or workstation, for example) usually supports one or more agents. They work as an interface between the management system and the component. Traditional agents do not filter information, which results in growing network traffic and growing difficulty to process information in a centralised way (Figure 2).

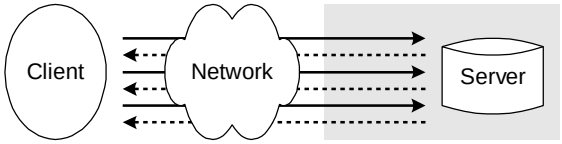

Figure 2 – Communication between a client and a server.

2.4 Management Protocol

The main task of the management protocol is the transfer of information and commands between agents and the management station, even when performed by remote invocation of code (as in JMAPI). To avoid as much as possible the overload caused by management systems on agents simple protocols are generally used. The complexity is thus concentrated in higher level services and on the NMS side.

In TCP/IP based LANs, the most used protocol is the Simple Network Management Protocol (SNMP) (Stallings, 1993) but its coexistence with proprietary solutions is a common reality. The simplicity of management protocols must be complemented by powerful network management systems, able to perform management operations in a multiprotocol management environment.

2.5 Drawbacks

This architecture presents several pitfalls related to its centralised organisation and to its communication intensive nature (Goldszmidt & Yemini 1998):

- **Network traffic increase** – the communication between the management station and agents introduces new traffic on the network.
- **Dealing with vast volumes of data** – the processing of large volumes of data can be dependent of a single system.
- **Robustness and fault tolerance** – when the management station fails, the entire system fails. When the communication between manager and agents is interrupted, the system fails.
- **Efficiency** – The information is firstly sent to the NMS, it must be processed locally and then commands must be issued to the agent. There is a relatively high consume of network resources.
- **Real-time response** – controlling entities across a network is asynchronously made.

3. SOFTWARE AGENTS FOR NETWORK MANAGEMENT

The growing diversity and extension of corporate LANs are increasing the management difficulty, particularly in simple but repetitive tasks. In addition, the number and diversity of operating systems and network applications spread all over the network is changing the management scenario towards information chaos. Some information could be processed by automatic procedures, away from user domain, resulting in a considerable decrease of information to be viewed (Figure 3).

One of the problems identified on SNMP framework is related with the lack of scalability of the model on very large networks. This constraint results from the inability of a centralised manager to handle huge amounts of management information and also because centralised polling across geographically distributed sites is infeasible and expensive.

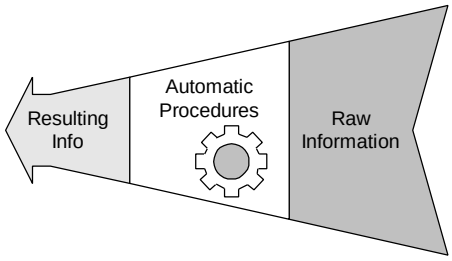

Figure 3 - Automatic processing of raw information.

This kind of problems has been addressed by the IETF' Distributed Management (disman) WG. This work group was chartered to define a architecture where a main manager can delegate control above several distributed management stations (DISMAN-framework). Instead of one centralised and usually very large application that concentrates all the intelligence of the system, a number of relatively small agents are involved in a co-operative effort to solve a problem. Considering all the diversity of choices to manage a network it is possible to ask which one is best? What kind of technology should be used?

3.1 Agent Definition

There has been some controversy about the definition of software agents, mainly due to its association with various expectations and contexts. One generic definition that seems to be consensual is that software agents are autonomous processes capable of performing actions with some specific goal, usually on behalf of some user or on behalf of another process (Finin et al., 1998). Other authors claim that an agent must also "perform its actions with some level of proactivity and/or reactiveness" and "exhibits some level of the key attributes of learning, co-operation and mobility" to meet the conditions to become a software agent (Green et al., 1997).

Both this definitions also includes intelligent and mobile computer code (mobile agents) (Agent Society).

Agents are used mainly in intelligent or adaptive user interfaces and distributed agent technology. Nowadays, much research is being done in providing intelligence to mobile agents.

3.2 Agents Working Paradigm

Nowadays, considering the Software Agents paradigm, there is a polarisation into two major factions:

- Mobility of agent's code and state (Bieszczad, 1998)(MASIF).
- Static agent classes that rely on agent intelligence and agent co-operation based on high level communication protocols (FIPA).

Each faction claims advantages over the other. In reality they are complementary, so integration is already being considered (FIPA 1998).

There are a number of models that describe an agent operation. The behaviour of an agent is based on the **Agent Model** that defines the internal structure of the intelligent part. Basically it defines the autonomy, learning and co-operative characteristics of an agent, as well as its reactive and proactive nature.

The agent's operation is very sensitive to security attacks. Security risks exist during registration, agent-agent interaction, agent configuration, agent-agent platform interaction, user-agent interaction and agent mobility. The **Security Model** identifies the key security threats in agent management and specifies facilities for securing agent-agent communication. If the agent is mobile, there must also exist addition protection against malicious agents (Stallings, 1995):

- privacy (assures that data is understood only by authorised parties),
- integrity (allows the detection of modified data by third parties),
- authentication (assures that the parties are really who they claim to be),
- access control (defines which resources can be accessed and by whom), and
- availability (assures that facilities are available when requested by authorised parties).

The other way is also valid – the protection of agents from malicious hosts – protecting

mobile agents from hosts which want to scan information, modify agent code or state, or kill the agent.

The communication between agents must follow a **Communication Model**. Agent would be of little use if they were unable to communicate one with each other, with users or any other entities. Currently there is some research on a common language for agents that will provide a high degree of interoperability between different architectures (Finin et al., 1998). As an example, KIF (Knowledge Interchange Format) (KIF) provides a solution to the translation problem and KQML (Knowledge Query and Manipulation Language) was specified as a language for inter-agent communication (KQML).

An agent can, in any time, be created, be running, moving (for mobile code), waiting or dead. The **Life-Cycle Model** (FIPA) describes when to change the execution state and which events cause the transition (Figure 4).

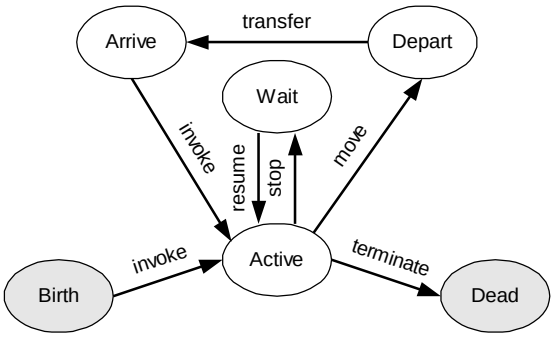

Figure 4 – Agent life-cycle model.

The agent is executed in a computational environment, meaning that it is allocated to a processor (a computer CPU or an abstract processor such as the Java Virtual Machine). The **Computational Model** specifies a set of primitives that defines the computational abilities of an agent, such as data manipulation instructions, thread control instructions and so on.

Concerning specifically mobile agents there are still aspects related to the transport mechanisms that must be considered. The **Navigation Model** defines:

- Naming conventions for all entities in the mobile system.
- Access privileges regarding a certain remote environment.
- How to transit an agent into waiting life-cycle state.
- How to transport a "waiting" mobile agent to other agent environment.
- How to receive "waiting" agents and resume it in the new environment.

Also some relevant issues associated to the navigation model includes directory and referential services to help on the discovery of services (LDAP)(Trader) and network topology information services, to help a mobile agent to make decisions based on the quality of different parts of the network (Ptopo).

4. MOBILE AGENTS FOR NETWORK MANAGEMENT

Since the beginning of the '80, the work related to network management issues were based upon client/server model. Each agent is embedded in network elements and they were created to be as simple as possible. The processing capabilities are left to the management applications and displayed at consoles in Network Operations Centre (NOC).

Ongoing work has already show the potential of intelligent agents in the construction of self-management tools (Lopes & Oliveira, 1997), particularly because they relieve the manager from repetitive and/or boring tasks.

Proposals and serious efforts have been initiated to transform client/server-based network management practices into a distributed and decentralized one (Breugst & Magedanz, 1998) (PMPP) (White et al., 1998).

Virtually, any task that can be performed by mobile agents can also be performed by stationary code so there are some questions concerning the use of software agents for network management that must be considered

prior to the development of such a complex system.

4.1 Update

Concerning software updating two methods are currently proposed: dynamic code distribution and individual installation.

The client/server paradigm present in the classic management model is rigid as it predefines the agent behaviour in compile time.

Emerging networked systems often require services that may be dynamically extended by remote applications. The delegation of code allows a more expedite software update method and can be performed without user intervention (Goldszmidt & Yemini, 1995). The agent may hold only basic functionality and extend their capabilities on-the-fly, on-site, by downloading required code of the network.

Mobile code facilitates the above capability and, as the pretended behaviour travels along with the agent, it is sufficient to update only the mobile agent.

4.2 Mobility vs. Communication

A static agent is installed in network nodes, continuously retrieving and processing information to the remote manager. On the other hand, a mobile agent advances from node to node, which gives him the opportunity to learn with the environment.

There are two aspects that must be considered between static and mobile agents. Static agents rely on data communication to exchange parameters with other agents and with the management station. According to the amount of exchanged data congestion may occur. Moving the code instead of information may be a new way of deal with congestion (Figure 5).

A mobile agent, due to its nomad attribute, has the ability to correlate the information from several nodes, which may help on the decision making.

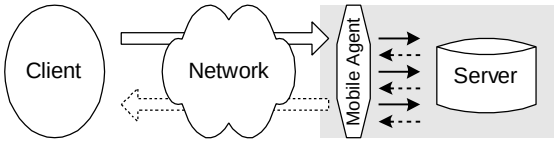

Figure 5 – Code movement instead of data movement.

4.3 Intelligence

The degree of intelligence that a software agent should have is dictated by several methods: dynamic learning, automatic decision, statistically data processing.

According to what the agent does its size varies. White (White et al., 1998-2) used a property of systems of unintelligent agents with limited individual capabilities and exhibiting collectively intelligent behaviour, such as an ant colony. This kind of system has simple units but demands a more advanced co-ordination system (Guérin et al., 1998).

On the other hand, configuration or diagnostic agents might get quite big. The use of neural networks and/or fuzzy algorithms increases the mobile agent complexity and extends its size. However, as saw above, there is always the possibility of extending the agent capabilities by downloading required code from the network.

4.4 Development paradigm

A mobile agent has a direct interaction with the environment where it is executed. The computational model defines the primitives that can be used by the agent so it is closely related to the selected mobile agent technology.

As simple examples, there are some fundamental functions in network management. One of them is the resource discovery used in network modelling. The classic `ping` is a very simple discovery tool, based on connectivity tests and, as so, it has some drawbacks. If there is the need for service discovery, more elaborated tools are necessary. As the

complexity of discovery grows, it is harder to implement in a client/server approach.

Another fundamental function is fault management, particularly network diagnosis. Specific mobile code may be used according to fault suspects as a way of refining the diagnostic. Moreover, mobile agents may encode complex fault detection and correlation algorithms not constrained to a single node.

4.5 Resource consumption

A mobile agent is executed only on one node at a time. Other nodes do not run an agent until needed. This causes a reduction of CPU and resource consumption relatively to static agents. In the later there is a duplication of functionality at every location. Mobile agents carry the functionality, which means that there is resource consumption only where it is executed.

Associated to this fact is also the reduction of network traffic. Code is typically smaller than the data it processes.

4.6 Security

Any code that is imported from the network to run locally has security risks associated. In fact, since external programs run within the same name space as the runtime system, the classic protection approaches, such as address-space containment no longer apply. Having the same permissions than the runtime system it may access unauthorized resources (privacy), abusive use or use more than its fair share of resources (integrity/privacy) and even deny resources to other programs (availability).

Data communication also suffers from security risks. Unauthorized parties may disclose the information (privacy), modify it (integrity) or even assure that the parties are correctly who they claim (authentication).

The prevention methods are similar for both situations: a) data communication and b) mobile code. In a), it is necessary to protect the information with some encryption algorithm (to prevent the message to be interpreted and to prevent the message from being modified) and it is necessary to authenticate the communicating parties. In b), the security mechanisms are similar, as it is necessary to protect the code from being modified (with the use of some encryption mechanism), correctly authenticate it and assure a trust mechanism.

5. CONCLUSIONS

Network management architectures have been standardized and put on the field over the last decade. Although there still remain unanswered questions: which will be the management architecture for choice? With the globalisation of Intranets over the Internet how do LAN management coexists with WAN management? How to deal with the explosion of management information? There are today lots of interest in new services (Aguiar & Oliveira, 1999). Through a management viewpoint the image is “more administration problems to handle”. Has the current manager/agent model, typically centralised and with a large dependence on polling, enough power to cope with this complexity? In fact market actors have not been satisfied by precedent management protocols (SNMP, CMIP, XMP, etc.) and motivate the exploit of other approaches pushed by several different fora (HTTP, WBEM, CORBA, JMAPI).

Scripting languages and virtual machine code such as Java permit programs to survive over heterogeneity and provides good platforms for code mobility. The advantage of mobile agents is related to software updates, resource savings, convenient development paradigm, reduction in network traffic and correlating capability. These characteristics make them an attractive choice for networks management environments.Its association with the automation of management tasks may help the user to deal more effectively with network growing complexity.

6. REFERENCES

Agent Society, The, http://www.agent.org/.

Aguiar, R., Oliveira, J.L., 1999, "Issues on Internet Evolution and Management", *ICEIS'99, 1st International Conference on Enterprise Information Systems*.

Bierman, A., et al., 1998, "Distributed Management Framework", Internet Draft <draft-ietf-disman-framework-02.txt>, August.

Bieszczad, A., Pagurek, B., White, T., 1998, "Mobile Agents for Network Management", *IEEE Communications Surveys*, September.

Breugst, M., Magedanz, T., 1998, "Mobile Agents – Enabling Technology for Active Intelligent Network Implementation", *IEEE Network Magazine, Vol. 12 No. 3*, May/June.

DISMAN-framework http://www.ietf.org/html.charters/disman-charter.html

Finin T., Labrou, Y., Peng, Y., 1998, "Mobile Agents Can Benefit from Standards Efforts on Interagent Communication", *IEEE Communications Magazine, Vol. 36 No. 7*, July.

FIPA, *Foundation for Intelligent Physical Agents*, http://www.fipa.org/.

FIPA 1998 Specification, Part 11, "Agent Management Support for Mobility", October.

Goldszmidt, G., Yemini, Y., 1995, "Distributed Management by Delegation", *Proc. Of the 15th International Conference on Distributed Computing Systems*, June.

Goldszmidt, G., Yemini, Y., 1998, "Delegated Agents for Network Management", *IEEE Communications Magazine, Vol. 36 No. 3, 66-71*, March.

Green, S., Hurst, L., Nangle, B., Cunningham, P., Somers, F., Evans, R., 1997, "Software Agents: A review", May.

Guérin, S., Snyers, D., Fourcassier, V., Théraulaz, G., 1998, "Modeling Ant Foraging Strategies by Computer Simulation", Ants'98, Brussels, Belgium, October.

ISO/IEC 10165-1, Information Processing Systems – Open Systems Interconnection – Structure of Management Information – Part 1: Management Information Model.

ISO/IEC 7498-4, Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part4: Management Framework.

KIF, "Knowledge Interchange Format (KIF)", http://ksl-web.stanford.edu/knowledge-sharing/kif/

KQML, "UMBC KQML Web", http://www.cs.umbc.edu/kqml/

LDAP Lightweight Directory Access Protocol, http://www.umich.edu/~dirsvcs/ldap/ldap.html

Lopes, R., Oliveira, J., 1997, "An Approach to Self Management based on Automatic Diagnostics", *Proc. of the 3rd. IEEE Malaysia International Conference on Communications (MICC'97)*, Kuala Lumpur, November 11-13.

MASIF, *Mobile Agent System Interoperability Facilities*, http://www.omg.org/cgi-bin/doclist.pl

Oliveira, J., 1995, *Arquitectura para Desenvolvimento e Integração de Aplicações de Gestão*, PhD Thesis, Universidade de Aveiro, September.

PMPP Perpetuum Mobile Procura Project, http://www.sce.carleton.ca/netmanage/perpetum.shtml

Ptopo PTOPOMIB Working Group, http://www.ietf.org/html.charters/ptopomib-charter.html

Rose, M., McCloghrie, K., 1990, "Structure and Identification of Management Information for TCP/IP-Based Internet", *Internet Request for Comments 1155*, May.

Stallings, W., 1995, *Network And Internetwork Security: principles and practice.*, Prentice Hall, 1st edition.

Stallings, W., 1993, *SNMP, SNMPv2 and CMIP – The Practical Guide to Network Management Standards*, Addison-Wesley.

Sun Microsystems, Inc., 1996, "Java Management API Architecture", http://www.javasoft.com/, June.

Trader CORBAservices – Trader Service, http://www.omg.org/

Vieira, G., Sá Morais, O., 1997, *Bancada de Gestão de Redes, University of Aveiro*, September.

WBEM Consortium, 1996, http://wbem.freerange.com/, July.

White, T., Pagurek, B., Oppacher, F., 1996, "Connection Management using Adaptive Mobile Agents", *Proc. Of 1998 International Conference on Parallel and Distributed Processing Technics and Applications* (PDPTA'98).

White, T., Pagurek, B., Oppacher, F., 1998 – 2, "ASGA: Improving the Ant System by Integration with Genetic Algorithms", *Proc. Of the 3rd Conference on Genetic Programming (GP/SGA'98)*, July.