A management system providing real distribution of management tasks with time and space independence

Francisco Fontes[1][2], Tomás de Miguel[2], Arturo Azcorra[3]

[1]Portugal Telecom - DID/CET
Aveiro - Portugal
Tel : +351 34 89 13 247
Fax : +351 34 20 722
Email fontes@cet.pt

[2]Universidad Politécnica de Madrid
Spain
Tel : +34 91 549 57 00
Fax : +34 91 549 57 77
Email {fontes, tmiguel}@dit.upm.es

[3]Universidad Carlos III de
Madrid - Spain
Tel : +34 91 6249993
Fax : +34 91 6249430
Email azcorra@it.uc3m.es

Abstract : Actual management systems do not fulfil a number of requirements identified as mandatory for an efficient and effective network management. The simple adoption of solutions like Java´s RMI or OMG's CORBA, do not solve all problems and may originate others. This publication describes the architectural and functional model for the proposed management system, that satisfy a number of identified requirements. A special attention is devoted to the information model, proposing a new structure for it.
Within the text some simple application examples of ATM networks are given. Finally, a brief analysis of two distinct ATM switches from different vendors is given referring to the possible added value that the proposed system can bring when managing those devices.
Keywords : Distributed management, flexibility, ATM

1. Introduction

Having an efficient and easy to use telecommunications network management system is of great advantage for the network operator that intends to be in the market lead. On the other side, integrated services and technologies are, today, replacing distinct networks that, in the past, were used to support different types of services. Broadband networks, with special relevance to ATM and IP based networks, are gathering more and more clients and are becoming the common factor for service integration and for globalisation of communications. The explosive growth of the Internet, the World-Wide-Web and the common acceptance of the Java language as the programming language to provide the development of portable applications make possible to define new approaches to the design and implementation of network management systems, and applications.
Research and Development in the network management area is very active. Distinct proposals from different research groups exist, each reflecting the different views and main concerns they have. CORBA [1] is one of these that is collecting more supporters, even if its complexity and performance potentially limit its applicability to the network management area. Solutions based on mobile code principles are also being considered. It is the basis for Active and Intelligent Networks [2] and for Management Mobile Agents [3]. In the first, the code to collect and process the data at each node travels with that data, and the second consists in autonomous entities that travel the network performing management actions for which they were mandated, acting on behalf of some managing entity. Other solutions, like WBEM [4] and JMAPI [5], address Web technologies to provide for a management platform. Management by Delegation (MdD) [6], a concept first introduced by Goldszmidt and Yemini in 1995, is taking repercussion in important organisms like ITU and IETF (discussion group *disman* [7]. Together with Mobile Agent technologies and WBEM, all have proposals for standardisation.
Regarding information models, the DMTF [8] initiative defined DMI and CIM [9], as continuing tasks of WBEM, and adopted XML [10] to represent management information. The CIM and XML are particularly interesting concepts to model and describe managed network resources and management information.

2. Requirements

The proposed management system was defined with an architecture and a number of services intended to satisfy a number of requirements. The identified requirements, presented next, address two main objectives: help users performing their management tasks with flexibility and efficiency while simultaneously keeping the system simple and reliable.
***Real distribution of management tasks execution*:** In order to achieve an efficient and useful observation of the managed environment, management tasks, defined by users, must be executed as close as possible to managed entities. This approach makes the defined system appropriate for an efficient observation of the quality of operation of the managed environment, allowing a better usage of network resources, namely network bandwidth.
Provide time and space independence to observe and control the network*:** Management operations should be defined and results should be collected from any place, regardless of security restrictions, without requiring special terminals. Additionally, tasks must have the possibility to be scheduled in advance with the management system performing them autonomously. Directly associated to these requirements is ***user mobility. During the execution of a particular management task,

and after describing it, users should be allowed to move to a different location and observe from there its current state of execution.

***Low latency to user inputs and to network events*:** The management system should react promptly to user interactions and to relevant events observed in the managed environment. The usual geographical separation that exists between users and managed entities requires the management system to be represented by appropriate elements closer to users and closer to network entities.

***Separation between user interface tools and the rest of the system*:** This to improve system flexibility, security and reliability. Using tools users are familiar with, like Web browsers, makes system usage easier. Additionally, being these components autonomous, limit their interference with other system components in case of their failure or low quality network connection.

***Easy adaptation to different managed environment dimensions*:** The management system must be applicable to a large number of situations and should be able to adapt continuously to environments frequently changing their dimension and complexity. Additionally, it should be able to change its management capabilities, all without interrupting the provided services. This is achieved by a modular structure of autonomous entities capable to work isolated, instead of the existing traditional monolithic models.

***Efficiency, Reliability, Simplicity*:** Simplicity, reliability and efficiency are characteristics that are inseparable. Efficiency and reliability are only reached if adopting simple mechanisms to support system development and operation. Complex services, with many dependencies on third entities, may lead to erroneous conclusions about problem origins and to unstable behaviour, presenting also a limited performance when compared with solutions using lower level mechanisms.

Available solutions solve scalability by the addition of elements with no internal organisation or in a rigid hierarchical fashion, following two or three-tier models assuming compromises between efficiency to users or to network devices. Others simply adopt CORBA or RMI without questioning the real advantages and disadvantages or these mechanisms. Another disadvantage in existing solutions resides in the role and flexibility present by middle entities. Normally all the processing and decision reside in final entities frequently distant from managed entities.

3. Proposed system architecture and concepts

From identified system requirements, two have special importance in the definition of the system architecture: *Low latency to user inputs and to network events* and the need to have a system *providing time and space independence to observe and control the network*. In the majority of situations managed resources are distant from network managers. Additionally, managed network resources are also used to support the exchange of management information, presenting variable delays on that communication, depending on the observed load, inclusively with the possibility to fail. In order to limit the consequences of those effects, and present a higher efficiency and reliability, the management system must be present closer to system users and closer to managed resources providing different services in each case. For these reasons, recent three-tier models, an evolution of two-tier models, adding a middle server that can be replicated, are not enough due to the limited flexibility to provide a real and efficient presence in front of users and managed resources. The placement of those servers implies a compromise between serving users as managed entities. The middle layer division is the right evolutionary step originating a four-tier like model with a convenient number of distributed system entities at each layer. Users, through appropriate user tools, and resources experiment a real presence of system services. Meanwhile, users are free to move everywhere connecting to the closest supporting system entity.

The system works on a client-server model with entities at upper layers requesting services to the entities on the lower levels. With an appropriate number of Managers present in the network in conjunction with available Web browsers, the system becomes available almost everywhere. Additionally, with a convenient number of Management Servers interacting with managed resources through their available management interfaces, allows the system to perform conveniently the requested management tasks.

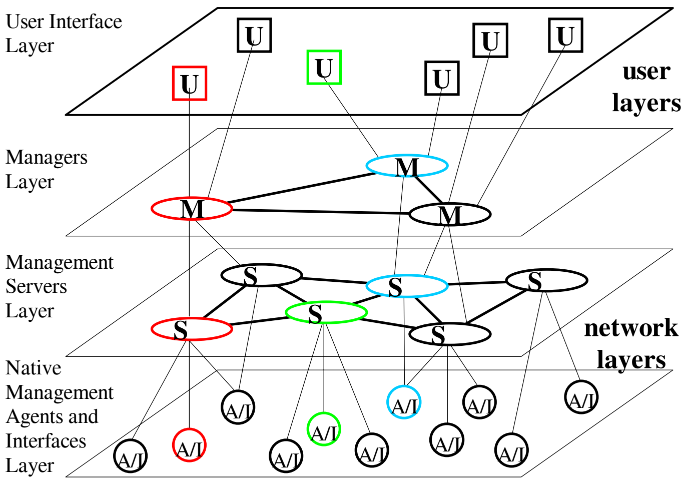


Fig. 1: System layered architecture

In the next sections a brief presentation of system constituting entities is performed. A detailed description is made in [11]

3.1 User interface tools

User interface support should be simple, based on inexpensive platforms and easily accessed from anywhere. Based on the portability of the Java code, the usage of Web browsers to interface the rest of the management system allows its usage from a wide number of platforms, with the same look and feel, and virtually anywhere on the planet, avoiding the need for specific, expensive terminals. Since Web browsers are used to navigate the Internet, all the required communications means already exist on the terminal. Actually, Web browsers and Java applets, in conjunction with some specific libraries, like the JFC [12], permit the development of rich user interfaces. OMG's CORBA and Java's RMI are supported by these applications having no limitation on the solutions to be adopted.

3.2 Managers

Managers are the system specific entities interfacing system user tools. All the base services aim at that objective, performing user authentication, supporting mobility and representing users in front of other system entities, even if they are not connected. Another important requirement put on Managers consists in the need to store operational and management information of interest to users. Users, using Java enabled Web browsers, are not able, due to the security restrictions imposed to Java applets, to permanently store information on the terminal used to interface the management system. Even if that was possible, storing information a particular terminal would limit mobility. This way, it is a Managers task to make available in the place a user connects to the system the required information for his operation. To accomplish that task, Managers must co-ordinate between them, identifying where the information is stored and how the user intends to have the information represented.

3.3 Management Servers

Management Servers are the most important entities in the system due to their role. Their principal characteristics are flexibility, achieved by a modular structure, and efficiency, to serve all the requested tasks. Management tasks to be performed are not previously defined. It is the characteristics of the environment managed by each particular Server and system users that determine which management needs will exist at each particular network entity and at each instant. Those tasks are supported by two different elements: Management Functions, organised in Functional Management Modules, and Management Operations, making use of the previous ones. While the first must be developed by experts, due to the required knowledge in programming languages, particular details of managed entities and on how to integrate them into the management system, the second are defined by system users and were defined to not require knowledge in any programming or scripting language. Both are explained in more detail later in the text.

3.4 Native Management Agents and Interfaces

Native Management Agents and Interfaces constitute the management facilities provided natively by managed entities. Following some standard or adopting a proprietary solution, they provide the access to retrieve management information or to perform management tasks. This forces the management system and, in particular, the interfacing system entities - Management Servers - to implement the same management protocol and to support the same information model. Once again, it is the Management Servers flexible modular architecture that allows the system to adapt constantly to a changing environment. In the current work special attention is being placed in SNMP, since this is a management protocol commonly accept by all manufacturers of network equipment, even if presenting some limitations.

3.5 Communication between system entities

Besides the correct distribution of functions by system entities in order to achieve an efficient and robust system functioning, communication mechanisms must be adopted carefully to avoid limiting system operation.
Currently some solutions with a large application in distributed environments due to the provided services, like Java’s RMI and CORBA, may be helpful but must have their application limited. One of the important aspects resides in the fact that both solutions base their operation on point-to-point TCP connections, thus, on the establishment of connection-oriented communication channels between system entities. These connections, used to remotely invoke the execution of some object method, last until the method completes execution, with or without returning some result, depending on the method particular operation. If it can be considered an advantage, in the network management area where management systems must present higher levels of reliability, the right solution can not be based directly in the remote invocation of management operations, supported by permanent connections of possible long duration. But the guaranteed service and the flexibility provided by these solutions, providing object serialisation, security, naming services, just to mention some, are advantages that must be considered. Therefore, a balance must be found between the existing advantages and disadvantages. The proposed solution uses remote method invocation mechanisms only to transfer management operations definitions in one direction and corresponding results in the opposite direction, in two different separated transactions. This process is particularly important in the interface between Managers and Management Servers, where the highest separation will exist being more sensible to network quality of operation. With this solution, connections will only last for the minimum required time duration, guarantying during that time the useful services they provide.
The same approach is applicable to communication between Managers and between Management Servers, even if in the first case due to the foreseen short duration interactions, a solution based in pure CORBA or RMI can

be adopted.
The usage of low level communication mechanisms allows the adoption of more efficient solutions like point-to-multipoint connections and the definition of multicast groups, not supported by RMI or CORBA.
Communication between user tools and managed resources to the rest of the system is governed by native mechanisms.

3.6 Management application scenarios

The proposed model presents great advantages for entities applying it to manage their networks. Besides providing an easy access to the management tasks using a normal

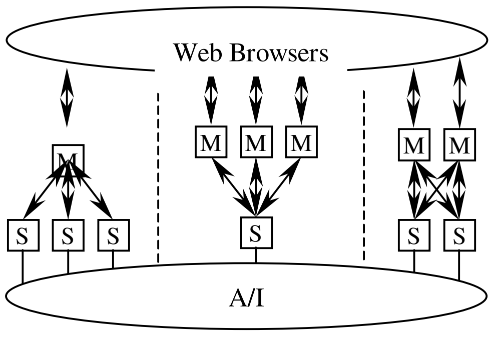


Fig. 2 – Management system application scenarios

Java enabled Web browser with a higher flexibility with no limitation on when and how to perform them, the proposed management system is applicable to a large number of scenarios. Having system functionalities distributed by the proposed four layers, permits the set-up of an appropriate number of entities in each layer adapting the management system to different environments.
In figure 2, a number of application scenarios are presented. In the first situation, the managed environment is large, in geographical extension, in the number of existing resources or in the needed volume of management tasks, requiring the presence of several Management Servers. At the same time system users will access the management system from a single point, connecting via their Web browsers. In the second situation the opposite is presented, where the managed environment is small, requiring low resources to manage them, while users will be present in several different places requiring the presence of a number of Managers to interface with. The last situation is the most common having an appropriate number of Managers and Management Servers, depending on the number and complexity of the managed environment and number of users.

4. Information model

This section proposes the information model required to support the management system operation. It refers mainly to system entities and mechanisms defined to generate, transport and analyse the relevant management information produced in a native format by the managed environment. Additionally, particular characteristics management information must present in order to fulfil the requested processing facilities inside the management system are suggested.

4.1 Functional Management Modules: a contribution to system flexibility and scalability

System specific intelligence and processing capacity is supported by two distinct components: **Management Operations (MO)**, defined by system users, and **Functional Management Modules** (FMM), developed by advanced users or other external entities, both supported by Management Servers. While the former express control and monitoring associations of interest to perform the management tasks, the later provide the communication and processing mechanisms to extract and process relevant management information from the managed environment. Each FMM makes available a number of **Management Functions** (FM) to be executed on managed entities, grouped by areas of interest and requiring common support. Some MF are applicable to the management information provided by other functions, without any direct contact with the managed environment. In any situation, MF, provided by appropriate FMM set-up at Management Servers, receive a set of input parameters, $\boldsymbol{P_i}$, and return a set of results, $\boldsymbol{R_i}$, after performing the defined management action. This way, FMM can be seen as the remote equivalent to traditional management applications running in the user terminal, providing the same functionality but closer to managed entities. An immediate application of FMM consists in the translation of single function invocations to long sequences of SNMP messages exchanged between a Management Server and a managed device in order to perform some task, such as establishing an ATM PVC. This approach provides additional efficiency, reliability and, at the same time, reduces latency to react to or report changes in the managed environment.
FMM, having the possibility to be added, removed or substituted during Management Servers runtime, provide the management system with the needed flexibility to grow unlimitedly and to adapt continuously to changing management requirements. With the possibility users, connected to Managers, have to define particular MOs, results returned by MF may be associated by arithmetic and logical operators and MF may be cascaded using results from ones as input parameters for the execution of others.
In order to instruct Servers on the MF execution, a set of associated parameters to each referred MF must be defined. These parameters may be introduced by the system using results from other management functions, use parameters defined for each particular Management Server, or be introduced by users. These parameters are:

- **Input parameters** for internal MF execution, instructing the function on how to perform particular actions it is able to do.
- **Managed resources** on which the MF will reflect its

actions.

- Particular system entity (Management Server) that will **support the MF execution**. Even if the default will be the one receiving the MO for execution, a different Server may be specified for a particular MF on which that particular MF will be remotely executed.
- A **selection criteria**, when applicable, for produced results (the nth result, the ith component of the nth result or one that satisfies a defined condition). This is applicable whenever the MF is able to return a set of results (e.g., all the established ATM PVC on a managed switch) and each particular result may be composed by several components (ATM PVCs can be defined by input port, VP and VC and output port, VP and VC)
- For periodic MO, the **frequency of execution** compared with the one of the MO (same, half, ...). This considers the existence of slower communications paths to some managed entities or slower managed entities that can not be overloaded with the execution of management operations.
- For periodic MO, instructions on the **usage of previous MF results**, in the same MO. This allows calculations based on changes from results of the previous MF execution or on the analysis of results from all the previous obtain results, for example, to make calculations based on the maximum value obtained during an observed period of time.

4.2 Management Operations

Management Operations (MOs), specified by system users, define relations between management functions (MF), provided by FMM and have associated a group of parameters defining execution conditions to be observed by supporting Management Servers. Depending on the defined relations, we have currently identified three useful classes of MOs, but more could be added in the future:

1. **simple operations**, making reference to a single MF, or a cascading of MF, which, after execution, will return the corresponding requested management information.
2. **arithmetic operations**, defining a mathematical expression to be evaluated, involving one or more MF results.
3. **logical operations**, complementing the definition of arithmetic associations between management functions results, defining autonomous monitoring rules and making the system react to defined events.

Arithmetic operators may be used to change the units of a measurement (e.g. change from bytes to bits) or to calculate differences between returned MF results (e.g. traffic losses, total traffic entering a switch).

In addition to the arithmetic and logical operators being used between results of management functions, it is also important to have management functions being performed having as execution parameters results from other management functions, $\mathbf{MF_n(MF_m(\ldots))}$. This is an added value to system flexibility and extensibility and may be used for:

- change management information nature or to extract relevant aspects from the information provided by $\mathbf{MF_m()}$. In both situations $\mathbf{MF_n()}$ is only executed internally to the supporting Management Server without any action in the managed environment.
- use results provided by $\mathbf{MF_m()}$ as execution parameters for $\mathbf{MF_n()}$. In this situation there is no functional difference between $\mathbf{MF_m()}$ and $\mathbf{MF_n()}$, both executing management actions on the managed resources.

The verification of the defined condition assumes the execution of an alerting or correcting action. Foreseen possibilities are:

- Alert the user - a message is sent to a specified place alerting about the defined condition verification.
- Execute a specified MF or arithmetic expression, associated to the MO, for the detected situation,
- Activate a separated correcting action,
- Stop the operation,
- Stop a different running operation

The third possibility implies the definition of two additional concepts: **detecting** and **correcting** MO. While the former checks the network for some relevant situation, the later reacts to that situation executing correcting actions. In order to make this mechanism useful, parameters must be passed from the detecting to the correcting MO. An identification of interesting parameters in the detecting operation and an indication on where to use those parameters in the correcting operation must be performed, using selection indicators for MF or MO results and input parameters. In the following example a detecting MO (MO_1) and a correcting MO (MO_2) are presented.

MO_1: $MF_1(\ldots)+MF_2(\ldots) > MF_3(\ldots) \rightarrow$ start MO_2

MO_2: $MF_4(\ldots) + MF_5(\ldots)$

MO_2 may use results from MF_1, MF_2, MF_3 or, inclusively, MF_1+MF_2 as input parameters to MF_4 and/or MF_5. This must be performed by system users signalling the system on the usage of internal system information holders that transfer and filter the produced information between MOs. Whenever the detecting MO defined condition is satisfied, the system looks for the marked information and introduces it in the correcting MO.

It may occur that simultaneous observation of different events will lead to the execution of the same correcting MO. In this situation the system must perform one of the following possibilities: ignore occurrences while serving the previous one, add to a queue for further attendance or start immediately a new instance to serve the request. It is again a user choice to indicate the best practice.

4.2.1 Management Operations components

MOs are comprise of several components. The first one, already referred in previous sections, concerns the syntactical definition of the operation. This definition, making usage of a number of suitable arithmetic and logical operators, instructs the management system on the calculations to be performed on management information provided by Management Functions (MF). The execution of these calculations must be controlled with the usage of time and numeric parameters. MOs are executed in one of three particular circumstances:

- To **correct** a detected network failure,
- To **change** at a pre-defined instant an operational or configuration aspect,
- To **gather** monitoring network operation information.

While in the first situation there are no associated time or numeric parameters, being the requirement the immediate execution of the MO, in the two last situations a scheduling must be performed. For that purpose the following parameters are associated to MOs:

- **Start time** for the first MO execution,
- **Stop time** for the last MO execution,
- **Waiting time** between the start of consecutive MO executions,
- **Number** of MO executions.

Under conflicting situations, the first parameter determining the MO stop should be used.

In addition to time and numeric parameters, other parameters need to be defined in order to fulfil the requirements identified for the management system, for instance, user mobility and flexibility, specifying:

- **MO Identification**, helping internal system operations,
- **Where to send** the MO for execution (executing Management Server). This can be done explicitly by system users or automatically,
- **What to do** with and where to send results from MO execution (definition on the usage of the Management Tree, a concept to be introduced later in the text),
- **Security information**, defining who has access to the operation executing (changing its parameters, for instance) and corresponding results,
- Information on **dependency relations** with other MOs, as seen for detecting and correcting MOs.

Management Operations require an appropriate representation for correct transport and distribution but also for editing and parsing at destinations. So far two different approaches were used, one purely textual and another one based in a complex object based structure. While the first proved to be more efficient for transport and distribution, the second one proved to be more efficient at destinations with less processing. The rapid growth of XML is being studied, with the advantage to be a standard, have flexibility enough to be integrated in the system and present a middle step between studied solutions.

4.3 Management information structure

The required capacity to have users specifying a number of operations between MF results and define appropriate MOs, requires the introduction of particular concepts around management information structure used as MF input parameters or produced as result of MF execution. Although the simplest situation would consist in MF needing a single parameter to execute and returning another single result, this will not be the common situation and would be a limiting factor in system flexibility and efficiency.

The following illustrative MO presents all the possible situations that can be observed on a MO definition, consisting in MF results being related by arithmetic and logical operators and acting as input parameters to other MF:

$$\mathbf{MO_i: MF_1 + MF_2 > MF_3 * MF_4 \rightarrow MF_5(MF_6)}$$

Assuming MF correspond to object methods, these methods need a number of separate input parameters to execute and return a value, possibly another object. In order to increase efficiency, MF may need to return a higher number of results. This is not supported in Java, as it is not supported in common programming languages, where object methods only have a single return type. In order to perform the expressed operations, and allow the return of more results it is proposed to structure management information in *components*. Each information component ***c*** consists in an indivisible information unit supported by some programming object or variable of appropriate type. When used as input parameters or being returned as results, these components will exist isolated or grouped in two possible structures: **basic elements**, for single results, and **matrix elements**, of n dimensions (n>0), for MF returning a number of elements. A MF returning a list of ATM PVCs, described by their characteristics would return a matrix with a number of rows corresponding to the existing PVC and a number of columns corresponding to the used components needed to characterise each PVC.

The suggested structure limits the returned management information components for the same MF to have all the same structure. Having ATM PVCs described by a different number of parameters must be supported by different MF. However, since information components may be objects, objects of different types may be used as the components of the management information provided by a MF solving that limitation.

Scalar, one dimensional and two-dimensional elements have a particular relevance in the system. With the proposed approach, standard rules applicable to operations between those types of mathematical elements may be applicable by Management Servers supporting MO execution.

Simple: $M = c$

One-dimensional matrix format:

$$M \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \ldots \\ c_n \end{bmatrix}$$

Two-dimensional matrix format:

$$M \begin{bmatrix} c_{11}, c_{12}, \ldots, c_{1m} \\ c_{21}, c_{22}, \ldots, c_{2m} \\ c_{31}, c_{32}, \ldots, c_{3m} \\ \ldots \\ c_{n1}, c_{n2}, \ldots, c_{nm} \end{bmatrix}$$

The available types and structures of management information components c depend directly on the adopted programming language. Using the Java language, where no matrices exist being the best solution the usage of Vector type variables, there is no direct method to distinguish between an information element constituted by a number of components and a number of scalar information elements organised in a matrix structure. The resort to management information associated attributes, presented in next section, makes this distinction possible.

In order to increase system flexibility, a set of default system functions must exist to allow the extraction of partial aspects of management information. These functions must allow the selection of a particular component $c_{i\ldots j}$ or sub-sets of components like a column, corresponding to a common aspect of several management information elements, or a row, corresponding to all the components of a particular management information element, in two-dimensional organisations. The selection criteria should be based on definitions made by the user. An immediate application of these operators to SNMP tables, after a complete retrieval using, for example, sequences of *GetNext* SNMP messages, and corresponding construction of two-dimensional matrixes, increases flexibility and processing capacity.

The structure description of input parameters and output results for each MF, following the presented concepts, should be supplied by FMM developers, in order to allow others to develop additional FMM that may act as either clients of results or providers of input parameters to already existing FMM in the system.

Management system should be able to adapt to some structural differences like when a MF expects a single scalar input parameter but it is presented with a matrix of input parameters. In this situation it should be the Management Server to extract one by one the input information components, run the MF for each component and return the aggregation of all results.

4.3.1 Associated attributes to produced Management Information

In order to support some of the system functionalities and mechanisms, management information produced by lower level entities, must be improved with additional attributes describing required details to be used by the management system or for user presentation. These details must address three aspects: **description of originating process** (generating managed entity and MO, instant of generation and order number), **destination identification** and **structural information**. With these three components it will be possible to process the management information during its transport inside the management network without loosing its identity, regardless of the presented contents. In simpler management models, like SNMP and CMIP, management information consists only in the content of some management object being the identification of sender and receiver automatically solve by the client-server adopted model, where management information is produced in response to requests made by managers.

5. Achieving scalability: the Management Tree

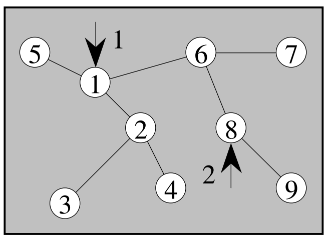
When the managed environment becomes too complex or too large, management tasks, if performed in the traditional mode from a single central point where all the processing and intelligence to take decisions is located, gets too difficult and the management system soon will reach its limit.

In order to obtain a higher level of scalability and an almost unlimited capacity to grow, in addition to the vertical organisation of system entities by layers where each level shelters a different system entity, an horizontal organisation of system entities is proposed. This horizontal organisation is reflected by Management Servers, since being these entities closer to managed resources, they are the ones that better reflect the managed environment dimension and complexity.

For this purpose, Management Servers, during set-up phase, establish a unique, fixed, logical and bidireccional communication channel with a Management Server already placed in the managed environment. Performing this operation for each new Management Server placed in the network, originates a tree like the one shown in figure 3.a, called the **Management Tree**. More that one Tree can be defined in the same environment, with the need to select for each new Server the Trees it should connect to.

The most important characteristics of the obtained organisation consists in the capacity it is has to connect, in the same structure, all the Management Servers of

a)



b)

Node	1	2	3	4	5	6	7	8	9
Neighbour	2	1	2	2	1	1	6	6	8
	5	3				7		9	
	6	4				8			

c.1)

c.2)

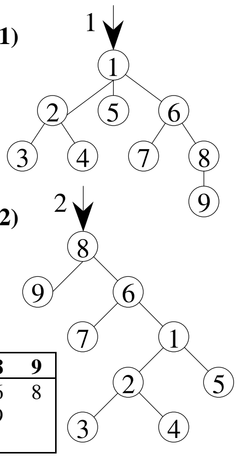


Figure 3 – Management Tree example:

interest, all the included nodes have the same importance and no loops will exist. In the Management Tree operation the inexistence of loops is of extreme importance. Otherwise the main tasks attributed to it, mainly the distribution of management tasks and management information gathering, would fail.

The Tree root location depends in the node taken as reference to start each operation that will use the Management Tree services. In figure 3.c two views of the same Tree are presented, corresponding to two different operations that selected different Tree nodes to be performed. From the picture it is clear a difference in the number of steps needed to reach all the Tree nodes. This is a decision factor in the Tree root selection process where other factors may be taken into consideration being the most important one the proximity to the system user (or, more precisely, to the Manager used to connect to the system).

Figure 3.b shows the connectivity table, built from information maintained by the Management Servers present in the Management Tree, existing in an upper layer entity that will serve for re-configuration operations. From this table it is possible to build the entire Management Tree.

In the situations where the technology used to support the network communications allow it, Management Servers organised by Management Trees, may constitute multicast groups and therefore increment operation efficiency. Otherwise they must enter more complex mechanisms by the exchange of appropriate messages to perform their tasks, as presented below.

In order to set-up, maintain, use or change the Management Tree, a number of messages must be supported by the entities involved in the process. Those messages, if making reference to the Java programming language, are supported by RMI operations between participating entities. Here some references are given without presenting an exhaustive list of operations:

Set-up messages: Management Servers registering/unregistering, report to upper level entity.

Maintenance messages: List established connections, detect loops on the Management Tree, substitution of a Tree node.

Usage messages: statistics on performed operations with resort to Management Tree services, operations to distribute MO through the Management Servers involved in the process, operations to report produced management information.

Operational messages: distribution of MOs, transport of management information.

5.1 Management Tree usage to distribute MOs

A managed environment is constituted by entities distributed in the network being natural that some of those entities, presenting similar characteristics, have similar management needs. Therefore, performing management operations in a distributed way, using Management Servers services, being each responsible for sub-sets of those entities, is the most appropriate way to proceed in order to guarantee an efficient realisation of tasks.

In large networks where the number of managed entities and the corresponding number of Management Servers is significant, the Management Tree can contribute in the setting-up of the defined MOs to all the existing Management Servers. For that purpose, a system client connecting to the closest Manager, defines a MO, which, using a particular Server (the Management Tree root), is distributed to the Servers in the network. Each Management Server, after receiving the operation from one of its neighbours in the Tree, stores that operation and makes a copy for each of its neighbours with exception to the one that delivered it. Making an analysis of the target entities to be managed by that operation, the Management Server starts as many operation instances as needed, each targeting at a particular managed resource.

In order to have the possibility to distribute the MO using this mechanism and apply them to a large number of managed resources, indication of those resources can not be explicit but must be implicit, using a selecting or filtering process. Otherwise, all Management Servers will discard the MO and only the one having the indicated managed resource on its management domain will execute the distributed MO. This could be a process to send a MO with an explicitly indication of the target managed resource but for which the corresponding Management Server is not known in advance.

Each distributed MO must be registered in order to know where to send the corresponding results, if using the Management Tree services, as explained in next section. Even if the MO is not applicable to the Server domain, it must register the neighbour that delivered the MO in order to make the information travel the inverse path defined by the MO distribution. This way, it is the distribution of the MO that defines the path management information will follow if it is to be collect using also the Management Services, as it is explained in next sections. Additionally, all MO started instances have defined the Server used to distribute the MO as the destination for the produced results. Servers do not know about users; they just deliver the management information to a requesting Manager that makes all the needed actions to guarantee the corresponding end user will receive all the results in time and correct order. As it will be seen in next section, the Management Tree can also participate in reporting management information that is generated by an higher number of origins (Management Servers).

5.2 Management Tree usage to collect Management Information

After setting-up a MO in a number of Management Servers present in the network, it is important to consider the available possibilities to transport the corresponding generated information back to the originator Server from where it can be retrieved by a Manager responsible for that task. That operation may take advantage of the

Management Tree. Three possibilities are available:

- Information **stays at Management Servers** where it was produced.

In this situation it is the Managers responsibility to retrieve the produced management information and, in conjunction with user tools, to represent it in the most convenient way (using attributes associated to the information to relate it)

- Information is **reported directly from all Servers to a central Management Server**.

In this situation the default used Management Server will be the one that initiated the MO distribution but may also be any other present in the network. From that Server, information is periodically read by a Manager.
Even in this situation where no change is made to management information inside the Management Servers level, the Management Tree may be used to transport it to destination. This may be based in the consideration that paths used to connect Tree nodes are more reliable and perform better than alternative ones. Another justifying situation occurs when there is no connectivity between one of the Servers producing information and the Server specified to collect it. Making the information travel through the Management Tree, where participating Servers only have to know the originating MO (that serves to determine the path towards the particular Tree root) and how to reach their direct neighbours, guarantees the information will reach its final destination. On these situations, the Management Tree only serves to gather, in an elegant and secure way, the produced information avoiding the need to poll, from a central point, each of the producing Management Servers. If each of the Management Tree nodes and the connecting paths are performing correctly, this is a simple but yet powerful way to perform that task.

- Information is reported towards the originating Management Server **using the Management Tree to extract at each Tree node the information relevant to the user**.

In order to reduce traffic near the final network node, providing an higher scalability, efficiency and robustness, the volume of received management information can be reduced by extracting at each node the relevant aspects defined by the user, while the information travels the network towards its destination. In this situation, the produced information travels using the inverse path defined by the distribution of the MO, being mandatory to have the Server that initiated the MO distribution as the final destination. The reason for that is based in the fact that Servers do not have a global vision of the network and, in particular, of the Management Tree configuration. They only know about their neighbours and it is the information registered during the MO distribution, even if at final they will not be running a copy of the MO, that allows them to send the received information, or the information produced locally, to the right neighbour in direction to the specific Tree root.

The volume of information reduction inside the Management Tree can be performed by **combination** or **selection** using special MF included in simple MO. Statistical and filtering operations will be the most common ones, even if FMM developers' creativity may define other functions. Identification of the *n* ATM PVP with highest traffic rates in a network with several ATM switches, being managed by different Servers, or the users average log-in time on the existing UNIX machines in a network, are only two examples showing how the Management Tree may avoid the final node from having to analyse huge amounts of information.
These operations impose additional requirements on system operation. First, when performing selecting operations, it should be capable to identify, at final, the resulting information origin. This is solved by the associated attributes to each management information sample that is forwarded to the next node, up in the Management Tree. With this, the final user is able to identify the presented resulting information origin, instant of generation and originating event. Second, in order to perform combining or merging operations, where the final result reflects global aspects of the generated information, complementary information components must accompany relevant components in order to allow calculations execution. Some calculations, like an average, need input parameters that are lost in the final result. In the example, instead of the final average result, returning the number of considered samples and the sum of all samples permits the inclusion of additional samples by the next Management Tree node. This way a careful development of FMM should be performed in order to not limit their usage using system services.

6. Application scenario for two distinct ATM switches

The first application scenario where the system is being applicable consisted in a local ATM network of switches, IP routers and terminals of different vendors. A special attention has been devoted to two particular ATM switches: Fore's ASX200BX and Cisco's LS1010. These two switches may be managed through the console or *telnet* using specific interfaces, or through SNMP. This last situation makes usage of SNMP MIBs supported by the agents running in the switches. Three SNMP MIBs have a special importance when managing these switches: IETF's AToM MIB, Cisco's Cisco-Atm-Conn MIB and Fore's Fore-switch MIB. While Cisco uses private MIBs to complement standard ones, Fore assumes a complete management based on private MIBs with exception to the mandatory ones like MIB-II.
The provided information models are limited by SNMP types, sometimes not the most appropriate ones. Most of network operation observation is based on indication of rates (e.g. traffic rates or error rates). This type of result is not provided by SNMP, being necessary to calculate changing rates retrieving two samples of supporting objects, normally based on counters variables of type

Counter or *Gauge*. In this situation, where, as referred previously, the network may present different delays to different messages, the existence of a timestamp field identifying the instant SNMP agents generate responses, would be helpful. This limitation is reduced in our system by the placement of Management Server entities closer to managed resources, limiting the sources of delay, and by the addition of *sysUpTime* (MIB-II) in the *varbind* field of each SNMP request. But even in this situation limitations exist consisting in the way agents make the update SNMP objects. It was observed a different but in both cases limiting behaviour. While Cisco updates objects at regular intervals of 1 sec. (*sysUpTime* is updated each µsec.), Fore updates all objects at each half second. Therefore, measurements requiring an higher accuracy can not be based on SNMP mechanisms at all, at least for the two observed devices.
For the two identified ATM switches, specific FMM have been developed to manage ATM connections (PVP and PVC) and to gather network traffic. These operations require the exchange of several SNMP messages. In the next table a minimum sequence of SNMP messages to establish an ATM PVC in the Cisco switch is presented. This is supported in the AToM MIB. The same operation for the Fore switch is entirely supported in the Fore-switch MIB.

CISCO PVC creation

operation : SNMP object	Value to Set
set: atmVclRowStatus.lowPortIndex.lowVPI.lowVCI	CreateAndWait(5)
set: atmVclRowStatus.highPortIndex.highVPI.highVCI	CreateAndWait(5)
set: atmVclRowStatus.lowPortIndex.lowVPI.lowVCI	Active(1)
set: atmVclRowStatus.highPortIndex.highVPI.highVCI	Active(1)
get: atmVcCrossConnectIndexNext	(Not applicable)
set: atmVcCrossRowStatus.atmVcCrossConnectIndex. lowPortIndex.lowVPI.lowVCI.highPortIndex.highVPI. highVCI	CreateAndWait(5)
set: atmVcCrossRowStatus.VccrossConnectTableIndex. lowPortIndex.lowVPI.lowVCI.highPortIndex.highVPI. highVCI	Active(1)

Other messages are needed in addition to the presented ones in order to gather switch particular information (number of ports, valid VPI and VCI ranges, etc.), to check the success of each SetRequest or to specify additional parameters regarding traffic descriptors.

Next, formulas for obtaining the traffic being generated and received in some ATM VP are presented. These requires the MF to retrieve two samples of the identified SNMP objects at instants t0 and t1.

Fore switch traffic per VP

Traffic per VP - Supported in the Fore-switch MIB
$T_{in} = (pathCells|_{t1} - pathCells|_{t0})/(t1 - t0)$
$T_{out} = (opathCells|_{t1} - opathCells|_{t0})/(t1 - t0)$

Cisco switch traffic per VP

Traffic per VP - Supported in the Cisco-Atm-Conn MIB
$T_{in} = (ciscoAtmVplInCells|_{t1} - ciscoAtmVplInCells|_{t0})/(t1 - t0)$
$T_{out} = (ciscoAtmVplOutCells|_{t1} - ciscoAtmVplOutCells|_{t0})/(t1 - t0)$

This formulas can repeating at regular time intervals in order to produce a graphical representation of its evolution. This is achieved by the parameters associated to the executing MO where the required MF is referenced.

7. Conclusions

Defined with the objective to provide users with an higher level of flexibility, having the possibility to schedule management tasks and to operate the system from everywhere independently of the target entities location, obtained system characteristics fulfil main identified requirements. The system presents a very high capacity to grow being accompanied by enough flexibility to adapt to all foreseen application scenarios. This is achieved placing a suitable number of organised modular system entities with required particular functionalities.
High level services, provided by Java's RMI and OMG's CORBA are integrated in a suitable way taking benefits from them but avoiding existing potential limiting characteristics.
Additionally, and in order to achieve required characteristics, managed information had to be complemented with additional parameters in order to permit its processing by the distributed system entities. This is in contrast with traditional information models where gathered information samples simply transmit the required value.
However, when used to improve SNMP operation, a number of inevitable limitations were found when used in a real scenario. Assuming the impossibility to place system entities directly inside managed resources, internal limitation on SNMP agents limit the system efficiency.
The main system drawback resides in the usage of the Java language in the development of some system entities where efficiency is a determinant factor. Here, the usage of faster machines and JIT compilers help overcome this aspect.

8. References

[1] : http://www.corba.org/
[2] : IEEE Network, Vol. 12 No. 3, "Special Issue: Active and Programmable Networks"
[3] : Stephen Corley, Marius Tesselaar, Kames Cooley, Jens Mienkohn,, Fabio Malabocchia, Francisco Garijo, "The Application of Intelligent and Mobile Agents to Network and Service Management", 5th International Conference on Intelligence in Services and Networks, IS&N'98, Antwerp, Belgium, May 1998, Proceedings
[4] : http://www.dmtf.org/wbem/index.html
[5] : http://java.sun.com/products/JavaManagement/index.html
[6] : J. Schönwälder, "Network management by delegation - From research prototypes towards standards", Computer Networks and ISDN Systems 29 (1997) 1843-1852
[7] : http://www.ietf.org/html.charters/disman-charter.html
[8] : http://www.dmtf.org/
[9] : http://www.dmtf.org/spec/cims.html

[10] : http://www.dmtf.org/spec/xmls.html
[11] : Francisco Fontes, João Bastos, Tomás de Miguel, Arturo Azcorra, "Proposal for a real distributed network management architecture", ConfTele'99, Sesimbra, Portugal, April 1999
[12] : http://java.sun.com/products/jfc/index.html