A Scheme for FTP Management

José A. Carrilho <carrilho@dcc.unicamp.br>
Edmundo R. M. Madeira <edmundo@dcc.unicamp.br>

Abstract

This paper presents a scheme for the application protocol management, which can be used for FTP - File Transfer Protocol [5] - management. This scheme includes the network division on hierarchical and federative domains, where each domain is composed of some agents under the same manager's administration.

In this paper, in order to manage the FTP, it is defined a new object group to be inserted in the MIB - Management Information Base, a new agent to maintain these objects and to answer the manager's requests, and some new management functions. An implementation of a prototype developed in Unicamp - University of Campinas - is also commented.

I. Introduction

The network management involves, among other activities, the maintenance of the network execution, including the network configuration and resource access control, and the maintenance of traffic rates in acceptable levels.

The rapid growth of the network number and size has made the network management so problematic. This fact has stimulated the development of network management tools.

The Internet model for network management is composed of four elements: managers, agents, a communication protocol and a MIB - Management Information Base. The managers read the agent management information and solve the problems encountered. The agent functions are to maintain the management information, to answer the manager requests and to report special events. The SNMP - Simple Network Management Protocol [1]- is the Internet protocol that defines the communication rules between the manager and the agent. The management information is hierarchically organized in the MIB. Some of the defined MIB groups [3] are:

- *system* group: contains generic configuration information about the managed node.
- *ip* group: defined for the IP - Internet Protocol - management, contains information about the sent and received IP datagrams, an address table and a routing table, among other objects.
- *tcp* group: defined for the TCP - Transmission Control Protocol - management, contains information about the retransmission algorithm, TCP connections, sent and received TCP segments and a connection table.
- *udp* group: defined for the UDP - User Datagram Protocol - management, contains statistics about the sent and received UDP datagrams.
- *snmp* group: defined for the SNMP management, contains statistics about the sent and received SNMP messages.

The Internet management is mainly based on the lower layer protocol management (network and transport layer). The SNMP is the only application protocol whose management was included in the MIB II [1]. The application protocol management is also very important, because it allows the network traffic identification and the access control to some resources. The identification of the application protocol that generates more network traffic permits a better resource allocation, for example, showing where more lines and servers are necessary.

This paper defines a scheme for the application protocol management, and in particular, for the FTP management, using the Internet network management model. The next section defines the domain concepts for application protocol management. The proposed object group and services for the FTP management are defined in section 3. Section 4 describes a scheme for FTP management implementation on ISODE [2] - ISO Development Environment. In section 5 it is shown how the scheme for the FTP management can be used by other network management systems. The conclusion is presented in section 6.

[1]MIB II: the second standard MIB defined by the Internet in 1989.
[2]ISODE: an environment used for implementation and use of the OSI/ISO softwares.

II. Network Management in Hierarchical and Federative Domains

The WAN - Wide Area Network - management can be split in domains, in order to reduce the complexity of the problem. A domain [10] is a component set with the same goals and subordinated to a common management. The components can be softwares, systems and other domains. A management domain is composed of an agent set subordinated to one or more domain managers, where each manager is responsible for a different aspect of the network management. For instance, a management domain can be an institution that has a LAN - Local Area Network - and that supports some agents in the managed systems and which has at least a manager to control them.

The domains can be classified in two types: hierarchical and federative domains.

II.A. Hierarchical Domains

Hierarchical domains reflect the network administrative hierarchy, i.e., the network administrative division is used for the hierarchical management domain definitions.

Some hierarchical management domains are shown in Figure 1, where a, b and c are managers subordinated to a superior manager, A.

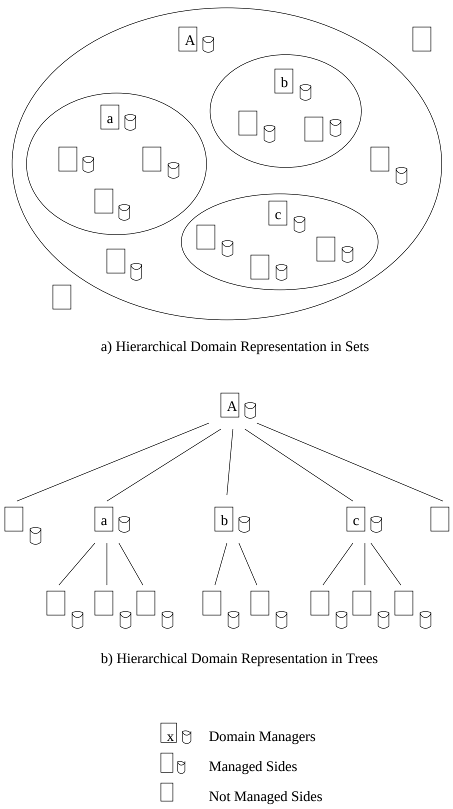

Figure 1: Hierarchical Domains

The management in hierarchical domains can be described as follows: every managed system maintains its MIB with the necessary information and an agent software that is subordinated to one or more managers. In large institutions, i.e., institutions where there are some LANs and many systems, there can be local managers. The local managers maintain a MIB with information about FTP transactions of the subordinates that involve systems of external domains. This information will be controlled by managers that stay in a superior hierarchical domain, for example, regional managers. These regional managers, in the same way, maintain a MIB with information about FTP messages of the subordinates that leave their management domain. These managers will be controlled by superior managers, like national managers, and so on.

The network management, in hierarchical domains, is able to obtain statistics about the information flow in each domain and among domains; for example, it is possible to control the information flow in the following levels:

- regional: it is possible to know how many messages are transmitted by a system and stay in the same region.
- national: it is possible to calculate how many messages are transmitted by a system and which leave the native domain, but stay in the national domain, i.e., the messages which are exchanged between regions.
- international: it is possible to know how many messages leave the national domain.

This domain management scheme can be used, for instance, by academic networks, as the BRN - Brazilian Research Network, which is illustrated in Figures 2a and 2b.

The national manager could be installed in a network backbone side, in the NOC - Network Operation Center. This manager would be responsible for the backbone control. Each regional network could contain a regional manager that would control itself and would maintain its MIB to be accessed by the national manager. Large institutions could contain a manager to control their LANs and would maintain the MIBs to be accessed by the regional managers. The managed systems would maintain their MIBs and their agents.

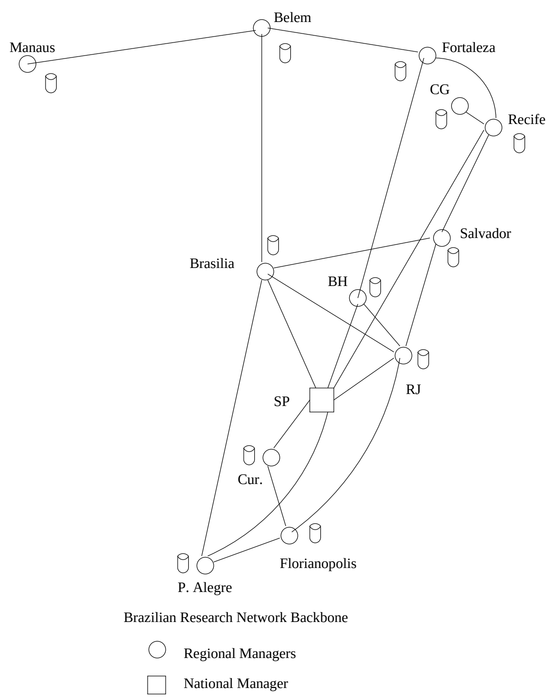

Figure 2a: BRN Management in the National Level

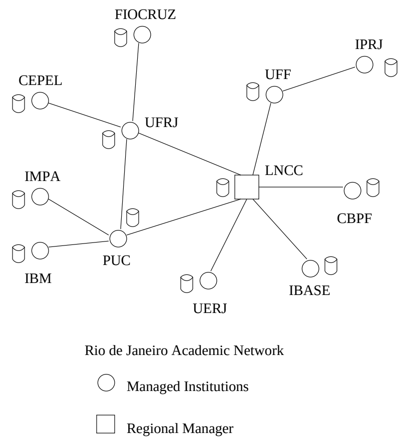

Figure 2b: RNP Management in a Regional Level

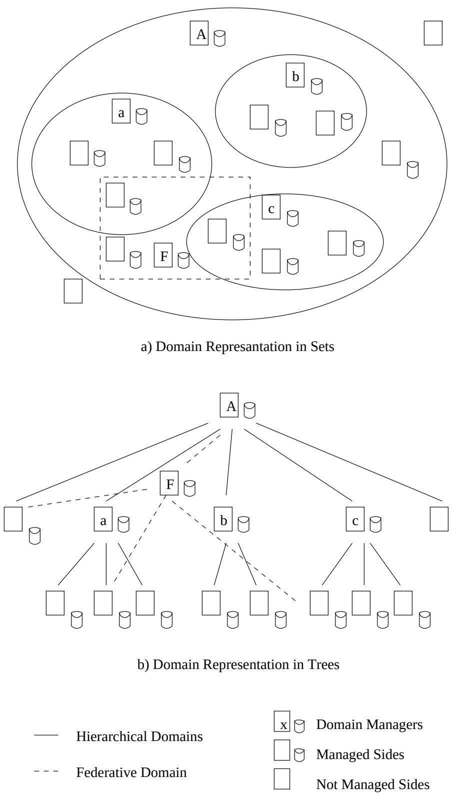

Figure 3: Federative Domain F

II.B. Federative Domains

Federative domains were defined because of the need for managing systems which stay in different hierarchical domains, but have the same goals or administration.

Figure 3 shows an example of a federative domain, where F is a federative domain manager and a, b, c and A are hierarchical domain managers. F is responsible for the administration of some systems that belong to the A, b and c hierarchical domains.

Federative domains are composed of some agents that can be in different hierarchical domains, and one or more domain managers that control the information flow among their agents. An usual application of the federative domain management is the control of institutions that have sites in differents regions, like BRN, ONU, and so on. These institutions could be managed in the following ways:

- Distributed Management: where each site is subordinated to the regional manager only.
- Centralized Management: where each site is subordinated to the regional manager and to an institution manager that would centralize all the site information. This kind of management would use federative domains.

Another application of a federative domain is to control the information flow between two specific systems that stay in different hierarchical domains. This kind of control would be done through the definition of a federative domain administrated by a new manager.

III. FTP Management

The application protocol management, like the FTP management, can be done in both hierarchical and federative domains. In this situation, each managed system that supports the FTP contains its MIB, an agent software and it is subordinated to one or more managers. These managers maintain the information about the FTP operations of their subordinate systems which involve external systems. The manager's MIB is accessed successively by higher level managers, up to the highest hierarchy level.

In order to manage the FTP, this paper defines:

- an FTP object group to be inserted in the MIB;
- a set of services that can be obtained by the FTP management.

III.A. FTP Group

In order to manage the FTP and file servers that are accessible through this protocol, the *ftp* object group was divided in 4 subgroups or bases. These bases were implemented according to the following characteristics:

1. General Base: is installed in every managed system that supports FTP in a prototype. The objects that may compose this base are:
   - an object that describes the software used for file transfer;
   - objects to account the number of sent and received messages;
   - objects to count the number of opened connections (for FTP commands and for data transfers);
   - objects to account the time that the connections were opened;
   - objects to control errors, like addressing or connection errors and bad access authorizations.
2. Manager Base: is installed in every system which contains an FTP manager software. This base may be composed of:
   - an object that describes the manager;
   - objects that contain information about the connections and the FTP messages that leave the manager's domain. This objects are similar to the General Base objects.
   - a table with the status and address of the FTP servers that stay in the manager domain.
3. Server Base: is installed in every managed file server of the prototype. The objects that may compose this base are:
   - an object that describes the file server;
   - objects that count the number of FTP connections besides the sent or received messages, including error messages. These objects are similar to the General Base objects;
   - a table that describes the server's files. This table contains: type and size of the file, access permissions, who saved the file, the date that it was saved and its version. It is also interesting to store a list of the last users that read the file.
4. Extra Base: will be installed when it is necessary to control the information flow between two specific systems. This base may store the following objects:
   - an object that describes the systems;
   - objects that maintain information about the FTP connections and messages between the systems. This objects are similar to the General Base objects.

III.B. Service Set

With The FTP management is possible to obtain statistics about the FTP data flow, i.e., we can obtain a real estimative of the network resource that is consumed by the file transfer.

Using the *ftp* group information that was defined in the last section, it is possible to get the following statistics:

- from the General Bases:
  - number of FTP opened connections;
  - average time of opened connections;
  - number of sent and received FTP messages, including errors messages;
  - number of sent and received bytes.
- from the Manager Bases:
  - more accessed file servers in the domain or in general;
  - status of the known file servers, in order to detect and solve problems;

  - information about file transfers from systems that are in the manager domain to servers that are in other domains.

- from the Server Bases:
  - more accessed files of each server;
  - who saved or read a file recently;
  - information about the FTP connections and messages that are sent and received by the file server;
  - besides, it is possible to send automatic messages to users that read a file recently, when a new file version is stored in the file server.

- from the Extra Bases:
  - the control of the information flow between specific systems. It is not recommended to have this kind of control over many systems and for a long time. This service is used in some special cases and for limited period.

The MIB configuration in the managed systems depends on the type of the service which is required. For example, this work describes the following configurations:

1. When it is necessary to offer only a simple estimative of the FTP information flow from a system, without worrying about the source and the destination of the FTP messages, it is only required the insertion of the General Base in the entity's MIB.
2. When it is necessary to control the information flow in a server, the Servers' Base must be inserted in the file server MIB.
3. When it is interesting to control the information flow in hierarchical and federative domains, it is necessary to maintain:
   - the General Base in all the managed systems;
   - a list of the servers, that are within the domain, in the managers' systems, in order to identify the FTP transactions that involve managers of the external domains.
   - the Manager Base that is accessed by the superior managers.
4. When it is necessary to control the information flow between specific systems, the Extra Base will be installed in the two systems. A federative domain can be defined and a manager can be installed in some of the systems.
5. When it is necessary to control the information flow from a system to a specific domain, an Extra Base will be installed in the system.

IV. FTP Management Implementation using 4BSD / ISODE SNMP

The ISODE [4] was used to implement the FTP management scheme in the prototype. The ISODE contains a management packet, the 4BSD/ISODE SNMP [8] that implements the SNMP. This packet is composed of an agent implementation and some tools for rapid-prototyping of network management applications. A simple management interface is presented in this packet, in order to send the SNMP requests and to receive the answers.

The MIB information is collected from the system Kernel or from the configuration files. A proxy agent can be used to export objects to the MIB of the SNMP agent, so it is possible to manage Internet external systems. The communication between the SNMP agent and the proxy agent is made with the SMUX protocol - SNMP Multiplexing Protocol [7].

In order to manage new information, such as information about the FTP, through the 4BSD/ISODE SNMP, the proposed scheme needs:

1. to define the new managed objects and insert them in the MIB using ASN.1 - Abstract Syntax Notation.One [6]. The new objects can be stored in the MIB under three subtrees [8]:
   - the *mib* group, defining a new version of the Internet standard MIB;
   - the *experimental* group, for experiments;
   - the *enterprises* group, for private objects.

   So the FTP Management requires the definition of a *ftp* object group in ASN.1 and its insertion in the MIB.
2. to implement a new agent software or to alter the existent agent in order to execute the new object management. The agent that is responsible for the *ftp* group was implemented

separately. This agent or subagent communicates with the SNMP agent through the SMUX protocol.

3. to define a manager to control the new managed objects and to carry out the new management services. The FTP managers control the *ftp* group communicating with the FTP subagent through the SNMP agent.

It was possible to access the 4 BSD/ISODE SNMP source codes, that gave us more flexibility for the object management insertion.

IV.A. FTP Information Base

The *ftp* object group, defined in section 3, was stored under the *experimental* group, because, at the moment, the FTP management is experimental. Some of the new *ftp* objects, that were defined in ASN.1, are shown in Appendix A.

The management data collection can be done in two distinct ways:

1. updating the source codes of the FTP clients and servers. In this scheme, the necessary codes for data collection are inserted into the source codes of the FTP clients and servers. Because the end-to-end management, that is, the management of the information flow between client and server, is important, all the management information can be collected in the sides of the client and server. The disadvantage of this scheme is the need to access the source codes of the FTP clients and servers.

2. opening the network packets. All the network packets are analyzed and the FTP packets are accounted for the *ftp* group. The problem with this scheme is the communication overhead caused by the packet opening.

In order to do some experimental tests, in the prototype the data collection was implemented altering the client and server source codes. In a real situation it is better to insert some collection systems in some strategic nodes. The function of this collection systems is to open the packets and to account for the necessary information, because it is impossible to update the code of all the available clients and serves.

Initially, the *Archie*'s client and server codes were altered, in order to collect the necessary information for the FTP management. As a consequence all the sent and received FTP messages in the client and server systems are computed, from the time the FTP connection is requested to its conclusion. The messages are classified as normal or error messages; also the number of sent or received bytes are computed, and information about the file transfer is stored, i.e., all the *ftp* group information is collected and stored in the MIB and in a log file. Actually the MIB data is stored in a data file that will be linked to the MIB by the agent.

IV.B. FTP Subagent

In order to make the agent's implementation more modular and opened, a subagent was created. The subagent is responsible for the FTP management only. The subagent communicates with the SNMP agent through the SMUX protocol and so it exports its management information. For this reason, it is not necessary to update the SNMP agent code. The agents are configured to perform the communication between the subagent and the managers.

Figure 4 shows the communication scheme between the FTP manager and the FTP subagent.

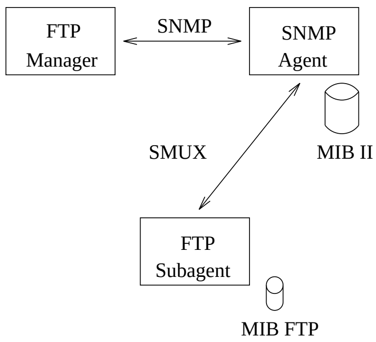

Figure 4: FTP Manager-Subagent Communication

The FTP subagent has the following functions:

- to communicate with the SNMP agent, through the SMUX protocol. A command interpreter is used to code and decode the SMUX messages.

- to execute the manager requests. After the command interpretation, it is necessary to carry out the manager requests. There are some procedures whose function is to identify the necessary objects, to find them in the data files and to read or to write the object values, only if the necessary authorizations are satisfied. After the end of the operation, an answer is sent to the manager, i.e., the data is encoded in a SMUX response message and sent to the SNMP agent.

- to send traps [3] when there are some special events to report. Trigger procedures are associated with some MIB objects, and they are executed when the objects take specific values. The subagent must identify when the procedures have to be executed. Usually, these trigger procedures send messages to the manager (traps). In this case the message is first sent to the SNMP agent and then to the manager. The traps that will be sent by an agent or subagent are previously programmed by the manager.

The FTP subagent process is always running and waiting for the connection requests that can be sent by the SNMP agent. The subagent can send traps to the agent at any time.

The FTP subagent is installed in all managed systems that support the FTP together with the *ftp* object group. The subagent is associated with a SNMP agent, that can be installed in a local or remote system. The local SNMP agent installation depends on the interest to have the management of other MIB objects.

IV.C. FTP Manager

The FTP manager process runs according to the polling method. Frequently, the manager verifies the agent objects and performs its management functions. When some trap is received by the manager, it can choose between contacting the agent immediately or waiting the agent time in the polling process. Some of the manager functions are:

- to control the FTP data flow;
- to get information in order to update its data base;
- to send messages to FTP users about, for example, new file versions, new file servers available, and so on.

In order to offer management services in hierarchical and federative domains, as defined in section 2, each manager stores a table that contains information about the file servers that belong to its domain. Therefore, the institution managers maintain their institution server addresses, the regional managers maintain a server list that is the union of the institution lists from their domains, and so on.

An FTP association is classified as internal when the manager knows the connected file server. When the FTP association connects a local system with an external domain server, the manager stores the association information in its MIB. The superior managers query the base and classify the associations as internal or external . So it is possible to maintain the information about the data flow among the domains.

When an FTP client requests an FTP association, the information in the local base is updated, independently of the domain in which the accessed server is. After the association is concluded, the information is stored in a log file, that will be frequently read by the manager. The manager collects the data and classifies the association as internal or external for its domain. The external association information is stored in the Manager's MIB and in a log file, in order to be queried by the superior manager, and so on.

V. FTP Management using other Network Management Systems

In this section it is described how two network management systems can be used by the FTP management. These systems are:

- *AIX NetView/6000* developed by *IBM Corporation*;
- *SunNet Manager* developed by *Sun Microsystems, Inc.*

V.A. AIX NetView/6000

The AIX NetView/6000 [2] is a network management application that implements the Internet management framework. There are SNMP agents and managers, and the managed information is stored in the MIB. Subagents can be used by other systems' management. The subagents communicate with a SNMP local or remote agent through two different protocols: the SNMPDPI, for the VM, MVS and OS/2 operation systems, and the SMUX, for the Unix systems. The managers send their requests to the agents using the SNMP, so the agent will translate the SNMP message to one of these protocols when necessary.

In order to manage new information, it is necessary to insert the new objects in the MIB using ASN.1, and to implement a subagent that will maintain these objects.

[3] Trap is a SNMP message sent to the manager to report some event.

Once the *ftp* group is defined and the FTP subagent is implemented in the ISODE, the FTP management using the AIX NetView/6000 is not difficult. It is only necessary:

1. to configure the NetView agent to communicate with the FTP subagent using the SMUX. It is possible to use the ISODE agent, since the NetView manager uses the SNMP;
2. to load the new MIB in the manager;
3. to configure the manager to execute the new management functions.

V.B. SunNet Manager

The SunNet Manager [9] is a network management framework composed of: agents, managers, MDB - Management Data Base - and proxy agents. The agents and managers use a service library for their communications. The managers and agents access these services through APIs - Application Program Interfaces, that use RPC/XDR - Remote Procedure Call / External Data Representation. The managers and agents use the same data definitions. The MDB is composed of an ASCII file set, where each file is maintained by an agent. The proxy agent allows the management of other different agent systems using the SunNet manager. For example, there is a SNMP proxy agent that supports the SNMP allowing the management of SNMP agents. The SNMP proxy agent translates the SunNet Manager RPCs to SNMP messages and so it can send them to the SNMP agent, and vice-versa. The SNMP proxy agent translates the SNMP agent information to MDB attributes. For instance, there is an ASCII file that contains the MIB II objects.

It is possible to develop new agents and managers, and to expand the MDB using the SunNet Manager framework, in order to manage new objects. The FTP management can be done either in this way or using the FTP subagent defined in the last sections. If the second option is chosen, then it will be necessary:

1. to translate the *ftp* group to an ASCII file that will compose the MDB;
2. to configure the SNMP proxy agent to perform the communication between the SunNet Manager and a SNMP agent, as for example the ISODE agent;
3. to configure the SunNet Manager to control the new MDB or to implement new management functions using the service library.

V.C. An Example using the ISODE, NetView and SunNet Managers

This paper presented a scheme for the FTP management using the ISODE, NetView and SunNet Managers. This section describes an example of an FTP subagent being managed by these three managers.

The FTP subagent communicates with any SNMP agent through the SMUX protocol. Figure 5 shows the FTP subagent communication with a SNMP agent, that may be ISODE or NetView agent. Because the ISODE and NetView managers use the SNMP, these managers communicate directly with the SNMP agent, in order to manage the FTP subagent. However, the SunNet manager uses RPCs to communicate with its agents. Therefore, the SNMP proxy agent is used in the communication between the SunNet manager and the SNMP agent.

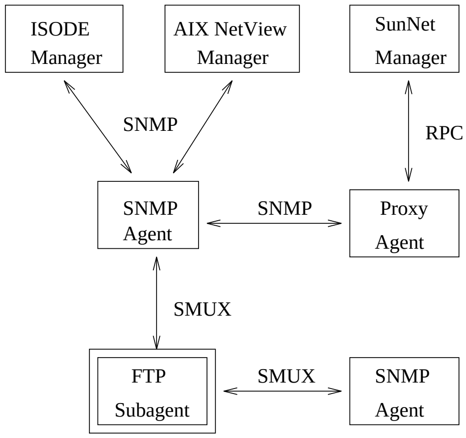

Figure 5: Managers - FTP Subagent Communication

In this example, there are three types of protocols involved in the communication between the managers and the FTP subagent:

1. SMUX: between the FTP subagent and the SNMP agent;
2. SNMP: between the SNMP agent and one of the three following entities: the ISODE manager, the NetView manager, and the SNMP proxy agent;

3. RPC: between the SNMP proxy agent and the SunNet Manager.

VI. Conclusion

The proposed scheme for the FTP management is very flexible, in the sense that it permits the FTP management according to the system's needs. The scheme may offer from a simple management where only the sent and received messages of the system are controlled, without concerning about their destination or source, until management services in complex hierarchical and federative domains. Also, the control of the information flow between two specific systems may be offered.

Although this management scheme has been implemented in the ISODE in the prototype, it can be used by other network management systems, as described in section 5.

The FTP management scheme presented in this paper can be extended to the management of other application protocols, like Telnet and SMTP - Simple Message Transfer Protocol. The application protocol management allows a better evaluation of the traffic generated by each application protocol and, therefore, a better resource allocation.

A new way of collecting the management information can be used in order to improve the efficiency of this management scheme. For instance, a software for messages' analysis, that can be installed in strategic network points, may be used. In fact, changing the data collection process does not mean that the management scheme also needs alteration. It is only necessary that the collected data be correctly stored in the data file to be found by the FTP subagent.

VII. References

[1] J. Case, M. Fedor, M. Schoffstall, and J. Davin. A Simple Network Management Protocol (SNMP). RFC 1157 (obsolete RFC 1067), May 1990.

[2] IBM Corporation, International Technical Support Center . Overview and Examples of Using AIX NetView/6000, May 1992.

[3] K. McCloghrie and M. T. Rose. Management Information Base for Network Management of TCP/IP-based Internets. RFC 1156 (obsolete RFC 1066), May 1990.

[4] Performance Systems International, Inc. The ISO Development Environment: User's Manual, Jul 1991.

[5] J. B. Postel and J. Reynolds. File Transfer Protocol - FTP. RFC 959, Oct 1985.

[6] M. T. Rose. The Open Book - A Practical Perspective on OSI. Prentice-Hall International, Inc., 1st edition, 1990.

[7] M. T. Rose. SNMP MUX Protocol and MIB. RFC 1227, May 1991.

[8] M. T. Rose. The Simple Book - An Introduction to Management of TCP/IP-based Internets. Prentice-Hall International, Inc., 1st edition, 1991.

[9] Sun Microsystems, Inc. SunNet Manager 1.1 : Installation and User's Guide, 1991.

[10] A. Wolisz V. Tschammer and J. Hall. Support for Cooperation and Coherence in an Open Service Environment. Proc. 2nd IEEE Workshop on the Future Trends of Distributed Computing in the 1990s, Sep 1990.

Appendix A: FTP Group

A list of some of the *ftp* group objects, which were installed in the MIB, is presented below.

```
ftpDescr OBJECT-TYPE
   SYNTAX DysplayString
              (SIZE (0..255))
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
     "A textual description of the  FTP
      entity. This value should include
      the FTP  entity  type  (client or
      server) and the software name."
   ::= { ftp 1 }

ftpInPkts (ftpOutPkts) OBJECT-TYPE
   SYNTAX  Counter
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
     "The total number of FTP  messages
      delivered  to  the FTP  entity by
      the  transport  service  (or  the
      reverse)."
   ::= { ftp 2 (3) }
```

```
ftpInBytes (ftpOutBytes) OBJECT-TYPE
   SYNTAX Counter
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
     "The total number of bytes
      delivered to the FTP entity by
      the transport service (or the
      reverse)."
   ::= { ftp 4 (5) }

ftpConnOp (ftpDtConnOp) OBJECT-TYPE
   SYNTAX  Counter
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
     "The number of FTP opened ( data )
      connections."
   ::= { ftp 6 (7) }

ftpConnTime OBJECT-TYPE
   SYNTAX TimeTicks
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
     "The Time which the connections
      have been opened."
   ::= { ftp 8 }

ftpBadAdds (ftpBadAuth) OBJECT-TYPE
   SYNTAX Counter
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
     "The number of FTP connections
      (or access command) refused due
      to an address error (or a bad
      access authorization)."
   ::= { ftp 9 (10) }

ftpConnTable OBJECT-TYPE
   SYNTAX  SEQUENCE OF ftpConnEntry
   ACCESS  not-accessible
   STATUS  mandatory
   DESCRIPTION
     "The FTP Connection Table."
   ::= { ftp 11 }

ftpConnEntry OBJECT-TYPE
   SYNTAX  ftpConnEntry
   ACCESS  not-accessible
   STATUS  mandatory
   DESCRIPTION
     "An entry of the FTP Connection
      Table. There are information
      about an FTP Connection."
   INDEX   { ftpRemoteAddr }
   ::= { ftpConnTable 1 }

ftpConnEntry ::= SEQUENCE {
              ftpConnState  INTEGER,
              ftpRemoteAddr IpAddress }

ftpConnState OBJECT-TYPE
   SYNTAX  INTEGER {
               opened(1),
               closed(2),
               trying(3),
               fail(4) }
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
     "The state of the local system
      with respect to an FTP
      Connection."
   ::= { ftpConnEntry 1 }

ftpRemoteAddr OBJECT-TYPE
   SYNTAX  IpAddress
   ACCESS  read-only
   STATUS  mandatory
   DESCRIPTION
     "The IP address of the remote
      entity connected."
   ::= { ftpConnEntry 2 }
```

Author Information

José Aparecido Carrilho is a master student at UNICAMP - University of Campinas in Brazil. He received a bachelor's degree in Computer Science from UNICAMP in 1990. He worked at BRN - Brazilian Research Network - with distributed directory and mailers using ISODE, from 1990 until 1992, when he began his master study on network management. Now, he is finishing his master thesis about application protocol management and returning to BRN.

Edmundo Madeira is a Lecturer of Computer Science at UNICAMP - University of Campinas in Brazil. He received, from UNICAMP, the master degree in Computer Science in 1985 and a Ph.D. in Electrical Engineering in 1991. His research currently focuses on higher layer protocols, specially on network management and transaction processing. He is also interested on Open Distributed Processing, currently leading in UNICAMP a group that develops a Middleware Layer for ODP.