In *Integrated Network Management, III* (H.G. Hegering and Y. Yemini (eds.)), pp. 389-401, Amsterdam: North-Holland, 1993.

An Application Gateway Design for OSI-Internet Management

Pramod Kalyanasundaram and Adarshpal S. Sethi
(email: kalyanas@cis.udel.edu, sethi@cis.udel.edu)

Department of Computer and Information Sciences
University of Delaware, Newark, DE 19716

Abstract

Network Management is an area of active current research. The problems in managing networks have been compounded by the existence of different standards. These problems can be solved by either forcing just one standard or by achieving unified management of dissimilar networks connected together. The latter solution is the one that is feasible. In this paper, we discuss and contrast two network management models (Internet and OSI) that are prevalent, define problems and issues involved in managing a network belonging to the Internet domain from an OSI domain, develop a paradigm that achieves uniform management of networks (using an application gateway approach) in either domain and present the justification for such a paradigm. Based on the paradigm developed, guidelines for the design of an application gateway are laid out. Future research directions based on this effort are included.

Keyword Codes: C.2.2; C.2.m; K.6.4
Keywords: Computer Communication Networks, Network Protocols; Miscellaneous; Management of Computing and Information Systems, System Management

1 INTRODUCTION

Internetworking has assumed importance due to the following: the existence of multiple standards and realization of the ultimate goal of networks, i.e., ubiquitous connectivity. With the evolution of large networks, network management has become a key issue from the viewpoints of administration, operation, reliability and fault tolerance. The problem of network management is compounded by the fact that more and more large networks with different network management models and schemes are to be interconnected in the process of achieving universal connectivity. This paper addresses the problem of interconnection in the context of network management only.

The scope of this paper in analysing the interconnection problem is restricted to the Internet and the Open Systems Interconnection (OSI) domain. The motivation for such an approach results from the following: these are the models that are prevalent, have current practical significance and allow one to concentrate on solving a subset of the general problem in the framework of the more general problem. An additional motivating factor is that the experience gained from solving this restricted problem will help in addressing the general problem of interoperation of two or more networks supporting dissimilar management models.

Research has been done [WS90] and is ongoing [Kni92] to achieve a unified network management environment. This paper develops a paradigm based on an application gateway approach [Ros90] that helps achieve internetworking of networks supporting the Internet model [RM90] and the OSI model [DIS90a] of network management.

We first describe and contrast these two models in Section 2 of this paper. In Section 3 we discuss different paradigms for internetworking and justify the selection of one paradigm to achieve internetworking namely the use of an application gateway. Section 4 addresses some of the issues involved in internetworking that must be considered in the design of such a gateway. It also presents possible solutions to these issues and outlines the design of an application gateway that achieves internetworking between OSI and Internet domains. The last section presents future research directions based on this effort and summarizes the conclusions drawn from this research.

2 NETWORK MANAGEMENT MODELS

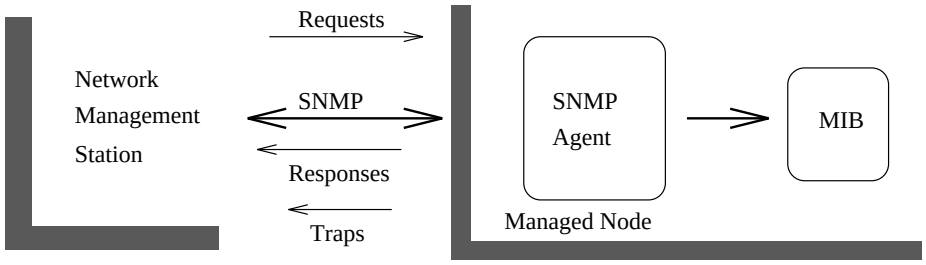
In this section, two common, prevalent models (Internet and OSI) of network management are introduced with a description of their salient features. The main idea is to provide a flavor for these two frameworks and to try and contrast the two models. The reader is referred to [DIS90a, Ros91b, Bla92, RM90] for details.

2.1 The Internet Model

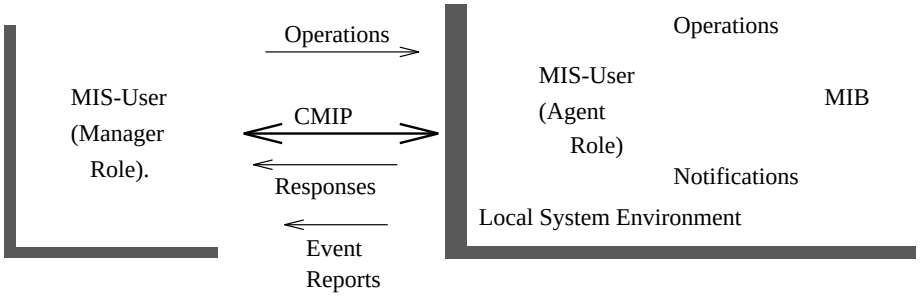
The basic model (shown in Figure 1(a)) comprises the following: at least one Network Management Station (NMS), several managed nodes and a management protocol for communicating management information between an NMS and a managed node. An NMS is a host that runs network management applications and also supports the management protocol. A managed node could be a host, a gateway or a media device. This model aims at having minimum functionality at a majority of nodes and more functionality at a small fraction of the nodes in the network.

Each managed node supports the operational protocols, the management protocol and an interface entity. The interface entity projects the 'real' variables defined and used by operational protocols as managed objects. This entity also defines a set of operations on these variables. The set of managed objects and operations are called the Management Information Base (MIB) [RM90, RM91]. Each Network Management Station performs the role of a manager and supports the management protocol and applications.

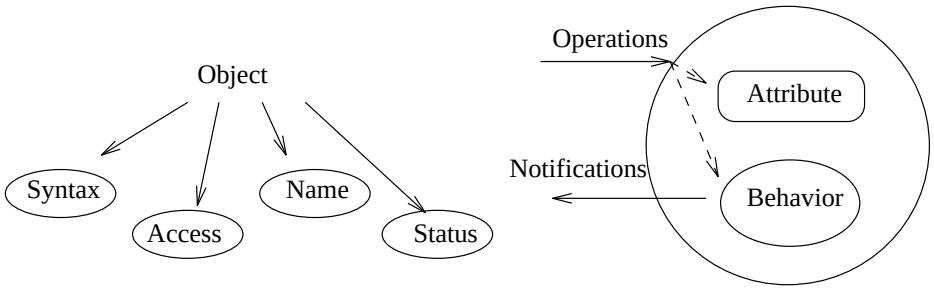
The Internet model is based on a request/response protocol. The roles of the manager and agent (managed node) are fixed. The manager by setting the value of a managed object can instruct the agent to perform a particular management operation. The manager can query the values of certain managed objects (which are changed by the managed node), to determine the current status of the node. Thus, the whole communication aspect of this model reduces to just setting and getting the value of managed objects. The Internet model defines objects (Figure 1(c)) [RM90, RM91] which have the following features: access, status, name and syntax. Access controls read/write privileges, status classifies the managed object as optional, mandatory or obsolete vis-a-vis implementation, name denotes the name of the managed object and syntax defines the object specification syntax. The name used to identify an object is called an Object Identifier (OID) which is a unique name in a global name tree.



(a) Internet Model for Network Management.



(b) OSI Model for Network Management.



(c) Internet Object Definition.

(d) OSI object Definition.

Figure 1: Internet and OSI Management Models and Objects.

The management protocol supported by the Internet Management domain is the Simple Network Management Protocol (SNMP)[1] [JDCD90]. SNMP is built over the well known User Datagram Protocol (UDP) [Pos80]. SNMP supports four basic operations: **Get** allows a manager to query an agent about the value of a particular variable. **GetNext** is a powerful operation that allows MIB traversal and facilitates table entry accesses. A **Set** operation is used by a Manager to set a variable to a specified value. The final operation, **Trap**, allows an agent to asynchronously report the occurrence of an event or fault to the manager.

Another important aspect of this model is the concept of a *proxy* agent. A proxy agent helps in achieving management of nodes that do not implement the complete set of functions defined by the management protocol or nodes that have a different protocol

[1] A revision of SNMP called SNMP Version 2 is currently under discussion in the Internet community.

suite and/or management protocol. It responds to management requests on behalf of the managed agents and can thus act as an application gateway [Ros91b] when there is a need to connect to a network supporting a different management protocol.

2.2 The OSI Model

The OSI model (Figure 1(b)) is primarily a more powerful and generalized model; like all powerful models, the trade-off is in terms of simplicity. Network management is achieved over associations established between application layer processes (called *management processes*). A management process can assume one of two possible roles: manager or agent. The same process may assume different roles over different associations but the roles are fixed for a given association as determined during association establishment. The manager process requests the execution of management operations from the agent which performs them and returns the responses to the manager. The agent may also generate event reports asynchronously to be sent to the manager. The manager-agent interactions are supported by an OSI standard management service known as the Common Management Information Service (CMIS) [DIS90d], implemented by the OSI standard management protocol called the Common Management Information Protocol (CMIP) [IS987].

As in the Internet model, each OSI agent controls a MIB. However, the structures of the MIBs in the two models are very different. The OSI model follows an object-oriented approach. Figure 1(d) highlights the managed object definition principles [DIS90b] used in this model. An OSI managed object is defined by the following: attributes, behavior, notifications and operations. Attributes are the characteristics specific to an object, operations are those that can be performed on the object (eg. setting a value, creating an instance), notifications are emitted by the object to indicate some event, and behavior which primarily dictates the changes in the object due to operations performed on it. This is distinctly different from the way objects are defined in the SNMP framework.

We now list the services provided by CMIS. The **M-GET** service is used by a manager to retrieve attribute values from managed objects. **M-CANCEL-GET** operation allows a manager to cancel a previously requested get operation before such an operation completes. The **M-SET** allows a manager to set the attributes of managed objects. The **M-ACTION** service facilitates operations on managed objects other than those defined as part of the OSI standard. Such a service allows designers to implement object specific actions based on bilateral agreement. **M-CREATE** allows the manager to request the agent to create an instance of a managed object. When the agent receives such a request, it attempts to create the specified object and responds with the result of the operation. To delete an instance of a managed object, a manager uses the services of the **M-DELETE** operation. Finally, **M-EVENT-REPORT** is used by the agent to report to the manager the occurrence of an event associated with a managed object.

The primary source of power in the OSI framework results from two major functions allowed by CMIS, namely *scoping* and *filtering*. Scoping allows a manager to select either a single object or multiple objects within a subtree of the MIB for a specific operation. Filtering allows the manager to specify a set of conditions which have to be met if the specified operation is to be performed on an object. In a typical manager's request both scoping and filtering may be specified. The agent first applies scoping and comes up with a collection of objects suitable for the operation. It then applies the specified

filter on the resulting objects. The requested operation is performed on all the objects selected by the filter.

3 PARADIGM DEFINITION

To achieve internetworking it is important to come up with a good paradigm that forms a basis for interoperability. In this section three paradigms that facilitate internetworking between CMIP and SNMP are described and contrasted. Of the three paradigms presented, one is chosen for this research. The reasons for such a choice are also explained. The three paradigms are shown in Figures 2(a), 2(b) and 2(c).

The first paradigm (Figure 2(a)), has most of the complexity built into the Network Management Station [WS90]. This paradigm implements a three tier system with the topmost tier being the management applications with suitable user interfaces, an intermediate layer (Network Management Language System (NMLS)) that provides a standard management command interface to protocol stack machines[2] and finally the stack machines themselves. Users specify commands in NML. These commands pass through the NMLS and are received by the stack machines through a standard command format. The stack machines are then responsible for converting this "standard command" into stack specific command format. These functions add complexity to the manager software and if implemented on a single node would lead to large space and time requirements for the NMS; if the implementation were to be distributed, then a separate interface (or protocol) needs to be designed that implements the distributed interfaces in a consistent fashion. With this paradigm existing managers cannot achieve internetworking without implementing the functions mentioned.

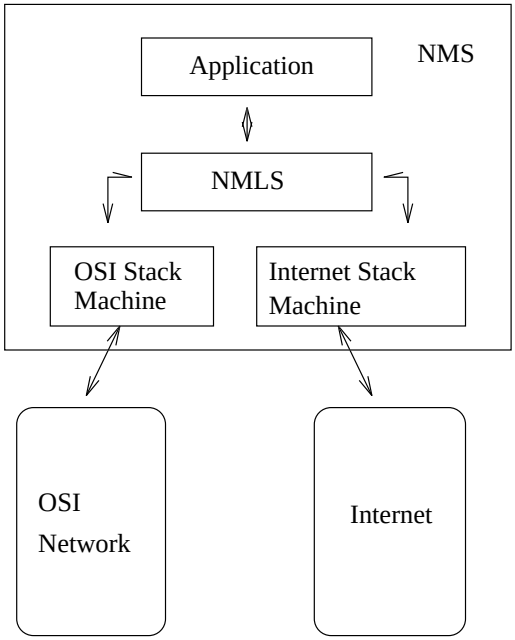
The second paradigm (Figure 2(b)), attempts internetworking using the concept of duplicate MIBs. This approach maps objects in the Internet MIB onto corresponding objects in an OSI MIB (for instance, MIB-II is mapped into OIM-II) [RFC91]. In this paradigm each node has an implementation of an SNMP agent as well as a CMIP agent. The Internet MIB and its mapped OSI counterpart are present at each Internet node. A request from an OSI manager is passed on to the OSI agent whereas a request from an SNMP manager is handled by the SNMP agent. The drawbacks with this paradigm are (a) The MIB translation is manually done. Hence changes in MIB objects involve re-work. Translating all existing MIBs in this manner would be a gigantic task. (b) There needs to be a version of a MIB and an agent that manages the MIB for each type of network being interworked and (c) The presence of more than one agent can lead to multiple reader-writers problems.

The final paradigm being presented (Figure 2(c)) uses the concept of an *application gateway*. In this approach there are special nodes designated as "gateways" that implement the translation between CMIP and SNMP. The rest of the nodes in either network retain their functionality. A scheme that allows easy name and object mapping is used that is independent of MIB changes. The mapping uses the characteristics of the Internet MIB instead of a direct one-to-one mapping of objects. This paradigm does not require more than one agent implementation, since the CMIS agent can achieve translation between protocols and only the translation algorithms need to be added. With such a scheme, a manager in the OSI domain can manage Internet nodes with absolutely no

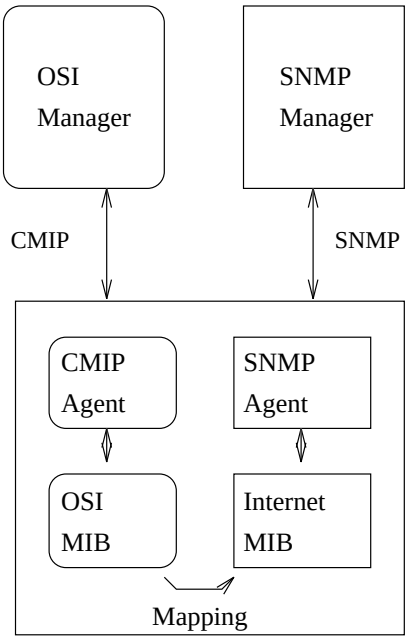
[2] A stack machine is a term used to denote the software implementing a particular protocol stack/suite of protocols.

change. This definitely is a major advantage on the OSI side since adding more functionality to a manager means that this has to be added to all nodes since any node could function as a manager.

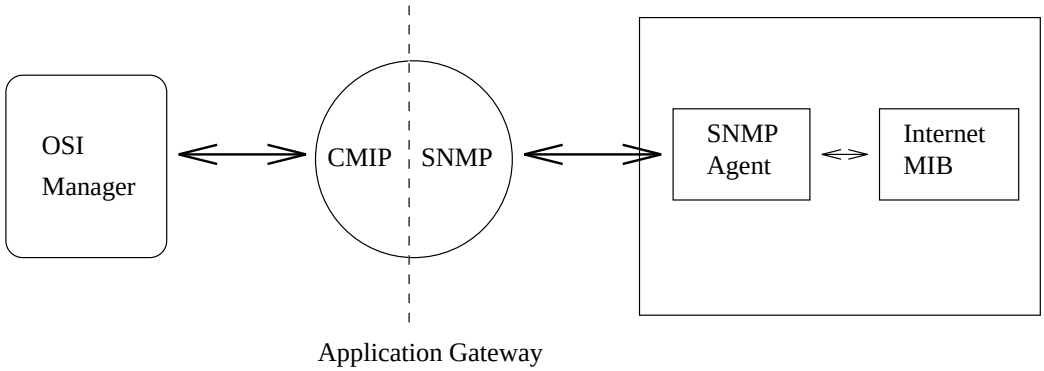
Application gateways interface with the OSI domain on one side and interface



(a) Unified Management of Heterogeneous Networks.



(b) Paradigm using Mapped MIBs and Separate Agents.



(c) Application Gateway Approach.

Figure 2: Paradigms that achieve CMIP-SNMP translation

with the Internet domain on the other (Figure 2(c)). A gateway inputs CMIP requests and outputs SNMP requests; it receives SNMP responses/traps and responds on the OSI side with CMIP responses to achieve internetworking. This paradigm allows a lot of flexibility and power. If a convenient interface can be generated, then several different translations could be achieved by a single agent by just allowing the agent to choose a suitable translation. This paradigm ensures reliability since the loss of a gateway does

not result in the manager losing control. By choosing an alternate gateway, the manager can continue unhindered. If an alternate gateway is not available still only that part of the network connected to the failed gateway cannot be managed. Another advantage is that only the required translations need to be present at each of these gateways. All these features have led us to choose this paradigm for our CMIP/SNMP interoperability research.

4 DESIGN OF AN APPLICATION GATEWAY

Based on the paradigm selected, this section presents the design for an application gateway, discusses the problems involved and presents solutions to such problems.

4.1 Name and Instance Mapping

Name and instance mapping assumes importance by virtue of the fact that the two management domains (OSI and Internet) have entirely different schemes for naming and defining objects. The Internet scheme defines two types of managed objects: *scalars*, which have only one instance and are static, and *tables*, which can have one or more dynamically instantiable rows. The scalar objects are named using Object Identifiers (OIDs) that are unique. The instance is identified using a standard suffix to these OIDs [Ros91a]. The table objects are defined in terms of the rows with each row object being made of several column fields. Each column field is named using an OID. Uniqueness of a row entry is achieved by using a sequence of fields called the *index*. The index is usually composed of the values of row entries belonging to specified columns. A column variable in a row is identified uniquely by suffixing the value of the index to the OID of the column field that needs to be operated upon.

The OSI scheme offers an approach where a set of guidelines are provided for object definition [DIS90b] and a user is allowed to specify the definition of object classes based on these guidelines. In the OSI framework, object classes are named using OIDs. Each attribute of an object is identified using an OID. However, instance identification uses the concept of a *containment tree* and is independent of the OIDs used to name the classes and attributes. The containment tree is a dynamic instance tree that captures the containment relationships between managed object instances. There is no equivalent concept on the Internet side. Each object instance in this tree is contained in another instance called its parent. All children of a given parent are identified uniquely with respect to that parent by a *Relative Distinguished Name* (RDN) which is an attribute-value pair. A *Distinguished Name* (DN) is obtained by concatenating the RDNs from the root of the instance tree to the node which represents the object instance being considered. The reader is referred to [DIS90b, Bla92, DIS90c] for further details on OSI object definitions and naming.

Since the two schemes are radically different, finding an efficient and clean method of mapping objects from one scheme to the other is of great importance in achieving internetworking. Such mapping will ensure that either management domain will be able to address objects and object instances that is in keeping with the local scheme by bridging the differences. In essence, name mapping must be simple and easy to automate. It must be generic enough so that changes in managed objects do not lead to the requirement for new mapping schemes. The scheme must capture the essential characteristics of the MIBs.

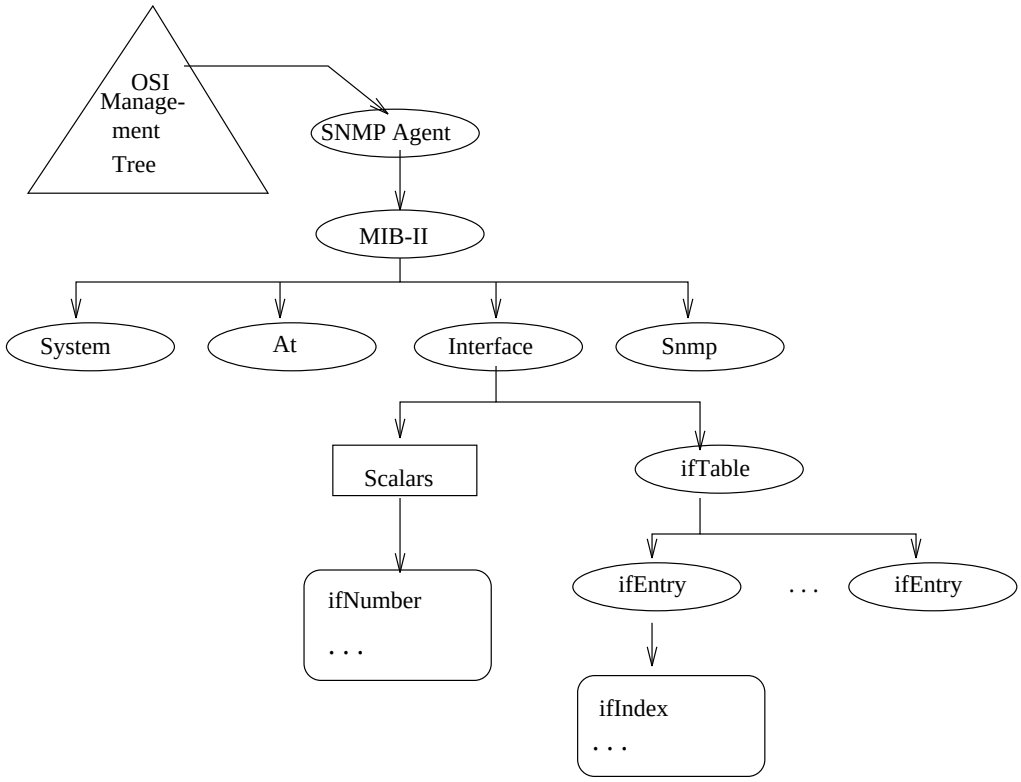


Figure 3: Mapping that preserves Internet MIB heirarchy

The name mapping scheme presented here maps the Internet MIB in such a way that an OSI manager is able to view the Internet MIB as a part of an OSI MIB structure. The name mapping structure has been chosen to preserve the Internet MIB structure (shown in Figure 3). The mapping scheme suggested does not aim at mapping each object on the Internet into an equivalent object or class on the OSI side. Instead it aims at providing the required OSI view by defining new object classes and mapping objects in the SNMP domain to objects and attributes in the OSI domain. This mapping defines two new object classes: *scalars* and *tables*. These classes are defined so that they are consistent with the nature of the objects supported by the Internet domain. These object classes could in future be enhanced to allow better internetworking.

Scalar objects on the SNMP side can have at most one instance. This characteristic maps well with the concept of an object attribute on the OSI side. Hence all scalar objects on the SNMP side can be mapped on to attributes of objects on the OSI side, with their OIDs retained.

Table objects on the SNMP side are made up of rows that can be dynamically created or deleted. This feature of table objects fits in well with the concept of dynamic instantiation or deletion of OSI objects. Hence a *row object class* has been defined. There can be multiple instances of this object since multiple row entries for a table are permitted. A row object has the column variables as its attributes. Each instance of a row object on the OSI side corresponds to one row entry on the Internet side.

Based on the above object class definitions, we present the structure of the mapped MIB (Figure 3). At the highest level, there is an object class for an SNMP agent. Depending on the number of SNMP agents being managed there will be a number of

instances of objects of this class. These SNMP agent nodes form the root of the SNMP part of the OSI MIB.

Each SNMP agent node contains a MIB object class. Each SNMP agent node is allowed to have exactly one instance of a MIB object associated with it. This MIB corresponds to either MIB-I or MIB-II. For our examples, MIB-II has been chosen. The MIB class contains the SNMP MIB-II groups (system, interfaces, at, ip etc.) as object classes. There can be exactly one instance of each of the group objects for each MIB. Each of these group classes contain two special object classes: scalar and table object class. Such definition preserves the features of the Internet MIB structure on the OSI side and conform to the OSI MIB environment. Each group object contains an instance of a scalar object and one instance of the table object for each corresponding table defined in the Internet MIB.

The scalar object in a group on the OSI side has scalar variables corresponding to that group in MIB-II as its attributes. Each Internet scalar variable can be operated upon similar to operations permitted on attributes of objects on the OSI side. Powerful operations can be achieved by using filters on these scalar variables. It is also possible to get any combination of these scalar variables. Each table object class consists of row objects that are defined using a row object class. Each table object instance contains row object instances that correspond to the row entries of a table on the Internet side. A row object allows an OSI manager to access a row of a table, and modify, create or delete it. Methods for identifying these row object instances are discussed later.

4.2 Functional Mapping

In order to achieve interoperability of the two management domains providing different services, it is mandatory that the services or functions provided by one domain be translated into the other. Such translation is achieved by functional mapping. Name mapping ensures that Management Trees of two different management domains appear similar to a manager in either domain. Functional mapping allows an operation allowed in one domain to be mapped onto one or more operations in the other domain. Functional mapping is the prime focus of this subsection.

The main problem in functional mapping arises from the difference in complexity of functions offered by the two domains. A powerful function in one domain may map into several simple functions in the other domain. It may not be possible to map all the complex functions provided by one domain due to inherent lack of power in the other domain.

CMIS defines services that allow querying and changing attributes of objects, reporting events, and creation and deletion of object instances. In addition, CMIS defines three other functions namely scoping, filtering and synchronization. These functions add complexity vis-a-vis mapping onto the Internet domain in which there is no counterpart. Hence, any functional mapping scheme between CMIS and SNMP must cover scoping, filtering and synchronization. These mappings must be applied in conjunction with the basic mapping between services in the two domains. For instance, an **M-SET** may map to a **Set** for a single object. However, if scoping and filtering result in more than one managed object selection, then the same **M-SET** service will map to one or more **Set**s if best effort synchronization is specified.

Packet size and bandwidth can present problems during function mapping. Packet size restrictions may force a request on the CMIP side to be broken up into more than

one request on the SNMP side. Hence an application gateway will have to maintain state information for such requests. This can add to the complexity of the application gateway.

Functional transformation may result in one of two possible cases: a single operation on one system may result in one or more operations on the other system; several operations on one system may map to a single operation on the other system. In either case there is an uneven distribution of packets resulting from functional transformation. This could lead to bandwidth hogging on the system on which multiple operations are required. Functional transformation schemes must ensure that the resulting transformations are both time and bandwidth efficient.

It is obvious that there are major differences in the schemes adopted by the two domains to access and operate upon managed objects, and traverse the Management Tree. Any functional transformation that is attempted must be simple, clean, efficient and easy to automate. The following paragraphs present the design of functional aspects of an application gateway. The transformations are unidirectional i.e. from OSI domain to the Internet domain.

Distinquished Name (DN) and Relative Distinguished Name (RDN): On the OSI side, generation of DN/RDN is important since a manager has to specify the DN when requesting an operation on a managed object. While accessing a scalar object, the DN is that of the scalar object instance to which the requested attribute belongs. The scalar object instances have fixed RDNs. The gateway maps requests for scalar objects by appending the OID of the attribute requested with a default suffix of zero to request a scalar variable on the Internet side. While accessing a row entry, an OSI manager specifies a DN that has the DN of the table object class suffixed by the values of the index fields. Since the index fields ensure uniqueness, the RDN is also unique. This DN contains sufficient information to enable the application gateway to construct the Internet OID along with the index to be used to access a specific row entry in the request to the SNMP agent.

The DN scheme allows a gateway to extract only the matching entry from the SNMP agent as opposed to getting several entries and then filtering them out at the gateway. This saves bandwidth and time. The gateway has internal tables that allow MIB information to be stored. These tables contain Internet managed object characteristics and are used to interpret and translate the DNs received in the CMIP requests.

Scoping: Scoping can result in multiple object selection. This implies that a specified operation (e.g., **M-GET**) has to be performed on all or a subset (due to filtering) of the objects selected. On the Internet side, scoping can be mapped by repeating the requested operation for each object selected. This repetition is implemented as part of an application gateway functional mapping. The application gateway maintains the mapped MIB information and uses this to decide which objects need to be selected for the requested operation.

The following choices are possible in a scoping request (as defined in CMIP): (1) Base object alone, (2) nth level subordinates of the base object, or (3) the base object and all its subordinates.

In case (1), the gateway maps the scoping function as a single get request to retrieve the base object alone. In case (2), the gateway has to traverse the MIB and come up with all managed objects that are at a defined level under the specified object. For instance, if all objects that are at depth 2 from the Interface group object are specified

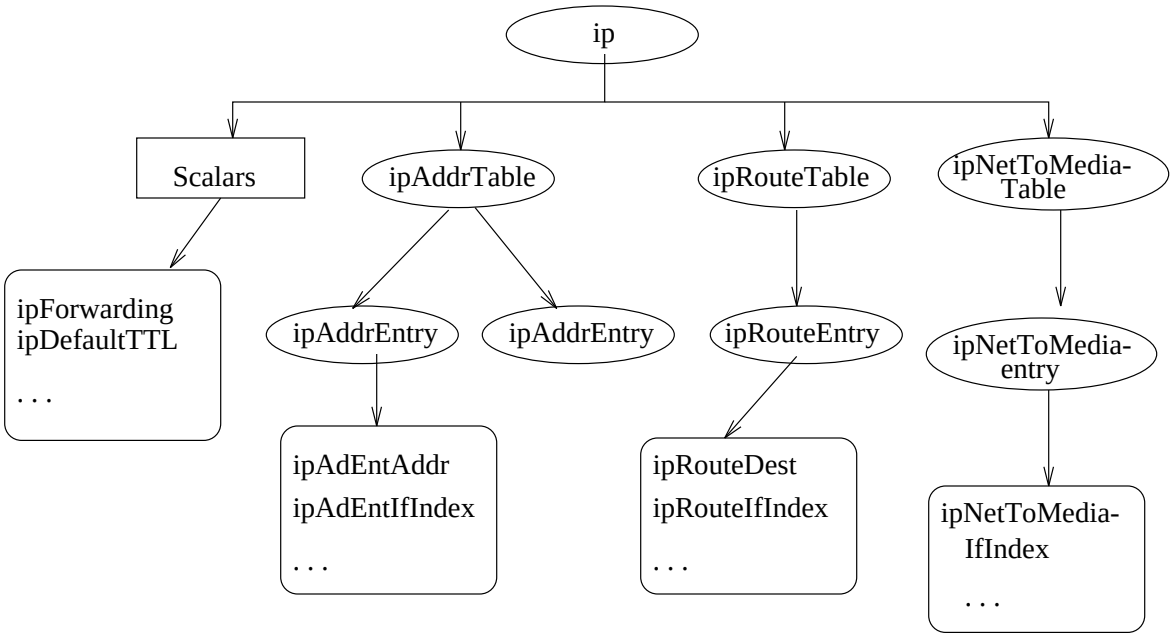


Figure 4: Instance tree of mapped MIB

by scoping , then the gateway will have to come up with all the row objects (Figure 4). Level 1 objects are the scalars and table object instances. The gateway will use the **GetNext** operation to extract information about all the objects at a specified level. In case (3), the function of the gateway is somewhat similar, with the main difference being that the gateway will extract information from all the objects defined in the MIB under the specified object. For instance, if the scoping reflects all objects under group object **Interfaces**, then the gateway will use the **GetNext** operator to extract all the objects defined under group object **Interfaces** using the OID of the Interfaces group.

In effect, the gateway will simulate the actions required by an SNMP manager to operate on several objects and at the same time preserve the interface on the OSI side. Such simulation ensures transparent operation of the OSI manager.

Filtering: A filter specifies a condition that must be true in an object for it to be selected for a specified operation. Filtering imposes further constraints on the objects selected by scoping for an operation. While scoping uses the object hierarchy in an instance tree to select an object, filtering uses the state of an object instance.

In order to implement the filtering function on the Internet side, an application gateway on receiving a request on the OSI side stores the filters associated with the request. The gateway then applies scoping to select objects on which the specified filters need to be applied. The managed objects selected are retrieved from the SNMP agent specified in the operation request. Once the objects have been retrieved, the gateway applies the stored filters and performs the operation only on those objects which test positive for the filter(s) specified.

Packet Size: CMIP has no restriction on packet size. SNMP uses UDP to implement its services. UDP has a packet size restriction which is carried over to SNMP. This packet size restriction adds complexity to the state machine of an application gateway. In cases

where a single request on the OSI side translates to several requests on the Internet side the gateway has to maintain the state of each request on the OSI side, collect several replies from the Internet domain and format them into a single reply. In such a case, an application gateway retains the state of the OSI request and collects the responses from the Internet side. Once the complete information is obtained from the Internet side, it is formatted and sent as a response on the OSI side. This adds complexity to the gateway state machine.

Synchronization: CMIS allows two types of synchronization across objects: **atomic** and **best effort**. Atomic synchronization has all-or-none semantics, i.e., the operation succeeds or fails on all objects selected. Best effort aims at performing the operation on as many selected objects as possible. SNMP supports only atomic synchronization[3]. Atomic synchronization on the OSI side translates to all the managed objects being a part of a single request. Best effort synchronization on the OSI side connotes that each request on the OSI side will generate requests that have a single object contained in them.

Based on the above, an application gateway has to allow only atomic synchronization when a manager requests update of multiple objects in a row of a table. Such atomic updates reflect SNMP behaviour and ensure table entry consistency. However, if a manager requests an operation on several scalar objects, then the gateway may either implement them as atomic synchronization or as best effort synchronization.

5 CONCLUSIONS

In conclusion, this paper introduces a paradigm that uses an application gateway approach to achieve internetworking between the OSI and Internet domains. Several key issues in achieving such internetworking are identified and discussed. Based on these issues, the basic design of a gateway that attempts internetworking using the suggested paradigm is suggested. Much work needs to be done in determining the guidelines for reverse translation (i.e from SNMP to CMIP), error mapping and trap to event-report mapping. Future directions include studying the effects of translation between connnection oriented service (CMIS) and connectionless service (SNMP). Extensions or modifications to existing protocols to facilitate better internetworking need to be addressed. This research has helped unearth a myriad of problems, the solutions to which will result in effective internetworking between OSI and Internet management domains. We hope that this work will help in finding a generic solution to the more important problem of achieving internetworking between any two dissimilar network management domains using an application gateway approach.

References

[Bla92] Uyless Black. *Network Management Standards.* McGraw Hill, 1992.

[DIS90a] International Organization for Standardization. *Information processing systems - Open Systems Interconnection - Systems management overview.*, 1990.

[3]Internet Management does not mention synchronization explicitly. It is mentioned as part of the **Set** operation that all or none of the objects specified will be updated.

[DIS90b] International Organization for Standardization. *Information Technology - Open Systems Interconnection - Management Information Services - Structure of Management Information Part 4: Guidelines for the Definition of Managed Objects.*, May 1990.

[DIS90c] International Organization for Standardization. *Information technology - Open Systems Interconnection - Management Information Services - Structure of Management Information - Part 1: Management Information Model*, January 1990.

[DIS90d] International Organization for Standardization. *ISO DIS9595, Information Technology - Open Systems Interconnection - Common Management Information Service Definition*, 1990.

[IS987] International Organization for Standardizaton. *Management Information Protocol Specification - Part 2: Common Management Information Protocol*, June 1987.

[JDCD90] Martin L. Schoffstall Jeffrey D. Case, Mark S. Fedor and C. Davin. *Simple Network Management Protocol. RFC 1157.* DDN Network Information Center, SRI International, May 1990.

[Kni92] Graham Knight, 1992. Private Correspondence.

[Pos80] Jon B. Postel. *User Datagram Protocol. RFC 768.* DDN Network Information Center, SRI International, August 1980.

[RFC91] DDN Network Information Center, SRI International. *OSI Internet Management: Management Information Base. RFC1214*, April 1991.

[RM90] Marshall T. Rose and Keith McCloghrie. *Structure and Identification of Management Information for TCP/IP based internets. RFC 1155.* DDN Network Information Center, SRI International, May 1990.

[RM91] Marshall T. Rose and Keith McCloghrie. *Management Information Base for Network Management of TCP/IP based internets: MIB-II. RFC 1158.* DDN Network Information Center, SRI International, March 1991.

[Ros90] Marshall T. Rose. Transition and coexistence strategies for tcp/ip to osi. *IEEE Journal on Selected Areas in Communications*, 8(1):57–66, January 1990.

[Ros91a] Marshall T. Rose. *The Open Book.* Prentice Hall, 1991.

[Ros91b] Marshall T. Rose. *The Simple Book.* Prentice Hall, 1991.

[WS90] Unnikrishnan S. Warrier and Carl A. Sunshine. A platform for heterogeneous interconnection network management. *IEEE Journal on Selected Areas in Communications*, 8(1):119–126, January 1990.