ISO/CCITT and Internet Management Coexistence (IIMC):

ISO/CCITT to Internet Management Proxy

(IIMCPROXY)

October 9, 1992

April Chang

NetLabs, Inc
4920 El Camino Real
Los Altos, CA 94022
april_chang@netlabs.com

Status of this Memo

This memo provides information to the network and systems management community. This memo is not proposed to be a standard, but is intended as a contribution to ongoing work in the area of multi-protocol management coexistence and interworking. This memo is part of a package of ISO/CCITT and Internet Management Coexistence (IIMC) drafts; see also [IIMCIMIBTRANS], [IIMCIMIB-II], [IIMCPARTY], and [IIMCOMIBTRANS].

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a "working draft" or "work in progress".

Please check the 1id-abstracts.txt listing contained in the internet-drafts Shadow Directories on nic.ddn.mil, nnsc.nsf.net, nic.nordu.net, ftp.nisc.sri.com, munnari.oz.au to learn the current status of any Internet Draft.

Distribution of this memo is unlimited. Comments on this memo should be sent to iimc@thumper.bellcore.com by November 20, 1992.

Abstract

This memo is intended to facilitate the use of the OSI Common Management Information Protocol (CMIP) for integrated management of networks via proxy management of TCP/IP networks that are managed using Simple Network Management Protocol (SNMP). This memo describes a ISO/Internet "proxy" which allows interworking between CMIP-based managers and SNMP-based agents. The proxy emulates CMIS service requests by mapping between corresponding ISO/CCITT GDMO and Internet MIB definitions, and generating SNMP message(s) needed to emulate the service. The proxy also emulates CMIS service responses and notifications by converting incoming SNMP response and trap message(s) in a similar fashion. Thus, the proxy appears as a CMIP-based agent to the manager, and as an SNMP-based manager to the agent. The proxy depends on the availability of corresponding MIB definitions translated as described in [OIMMIBTRANS].

Table of Contents

Status of this Memo i
Abstract ii
Table of Contents ii
1. Introduction 1
1.1. Background 1
1.2. Overview 2
1.3. Scope 3
1.4. Terms and Conventions 5
2. ISO/SNMP Proxy Configuration 6
2.1. Schema Information 6
2.2. Proxy Objects and Party Objects 6
2.3. Usage 8
3. Elements of CMISE Service Emulation 9
3.1. Association Service 9
3.2. Management Object Selection 10
3.3. Synchronization 12
3.4. Management Operation Services 13
3.5. Management Notification Services 19
3.6. Common Response Parameter 19
4. CMIP and SNMP Protocol Mapping 20
4.1. Management Object Selection 20
4.2. Check for Existence of the Object 21
4.3. Create With Referenced Object 23
4.4. Perform the Get Operation 23
4.5. Perform the Set Operation 23
4.6. Perform the Create Operation 23
4.7. Perform the Delete Operation 24
4.8. Form a Variable Binding 24
4.9. SNMP error to CMIP error mapping 25
5. CMIP Processing Failure 26
6. Functional Units 27
7. Abbreviations 28

8. Acknowledgements 28
Appendix A: CMIP to RFC 1157 Agent proxy 29
Appendix B: Example Operation 31

1. Introduction

1.1. Background

This memo is part of a package of ISO/CCITT and Internet Management Coexistence (IIMC) drafts. Other memos included in this package are:

- Translation of Internet MIBs to ISO/CCITT GDMO MIBs (LaBarre) [IIMCIMIBTRANS]
- Translation of ISO/CCITT GDMO MIBs to Internet MIBs (Newnan) [IIMCOMIBTRANS]
- Translation of Internet MIB-II (RFC1213) to ISO/CCITT GDMO MIB (LaBarre) [IIMCMIB-II]
- Translation of Internet Party MIB (RFC1353) to ISO/CCITT GDMO MIB (LaBarre) [IIMCPARTY]

These memos together comprise a package aimed at integrating ISO/CCITT-based and Internet-based management systems. These memos are offered as input to coexistence and interworking efforts underway throughout the industry,including organizations such as:

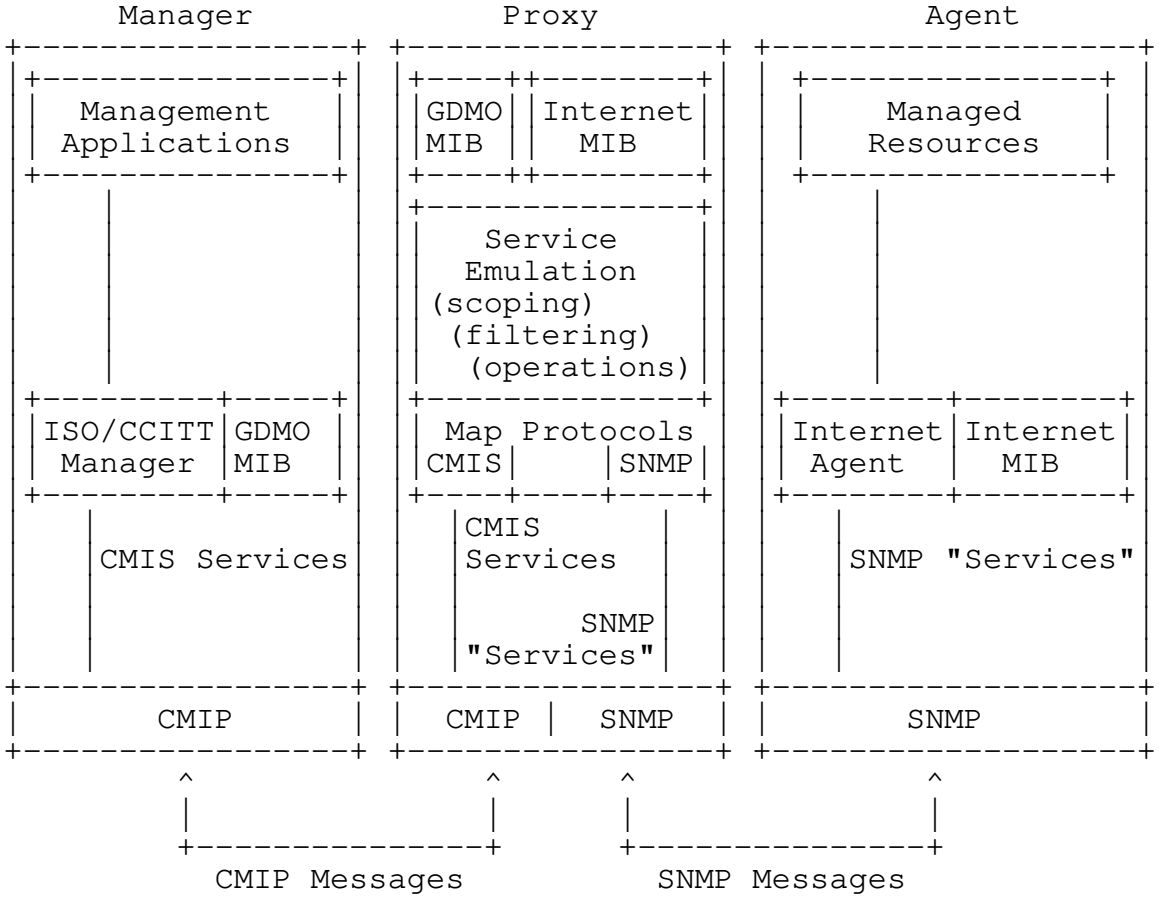
- IETF OSI Internet Management (OIM),
- Network Management Forum Technology Convergence Team,
- X/Open Systems Management (SysMan),
- OIW Network Management Special Interest Group (NMSIG), and
- OSF Management Special Interest Group (MANSIG).

This work was initiated, in part, by NM Forum efforts to translate RFC 1214 for use with OMNIPoint 1 implementations. Through this effort, it became obvious that end-to-end management requires an integrated, unified view of the managed network, despite differences in management protocol and information structure. Integrated management can be facilitated by the development of "proxy" mechanisms which translate between functionally equivalent service, protocol, and SMI differences to create this unified view. MIB translation procedures can be used to support proxy management, as well as to take advantage of existing MIB definition and avoid duplication of effort. In this way, commercial investment in both ISO/CCITT and Internet-based management technologies can be preserved through deployment of common methods and tools which support integration.

This overall strategy was outlined in a joint publication developed by the NM Forum and X/Open entitled "ISO/CCITT and Internet Management: Coexistence and Interworking Strategy" [NMFMC92]. The memos included in the IIMC package are intended as detailed specifications which implement several of the methodologies identified in this strategy.

1.2. Overview

The basic model for ISO/CCITT-Internet proxy management is illustrated in the following diagram.



The proxy architecture provides emulation of CMIS services by mapping to the corresponding SNMP message(s) necessary to carry out the service request. The service emulation allows management of Internet objects by an ISO/CCITT manager. The left hand side of the proxy behaves like an ISO/CCITT agent, communicating with the ISO/CCITT manager using CMIP protocols. The right hand side of the proxy behaves like an Internet manager, communicating with the Internet agent using SNMP protocols.

The proxy relies on the existence of a pair of directly-related MIB definitions, where the Internet MIB has been translated into ISO/CCITT GDMO using the procedures specified in [OIMMIBTRANS]. The proxy defined in this memo uses these MIB definitions and rules to provide run-time translation of management information carried in service requests and responses.

The proxy architecture is designed with a specified interface between the proxy and the underlying protocol stacks, and so deals primarily in terms of CMIS services and SNMP "services". The proxy emulates services such as CMIS scoping and filtering, processing of CMIS operations, and forwarding/logging of CMIS notifications by performing a mapping process which must be tailored for each protocol (for example, SNMP, Secure SNMP, and SNMP-2 are all variants of the same protocol mapping process).

Finally, [IIMCOMIBTRANS] specifies translation procedures for converting ISO/CCITT GDMO MIBs into Internet MIBs. MIBs generated by this translation process cannot be utilized by the Proxy defined in this memo, although another kind of Proxy could be defined for this purpose in the future.

1.3. Scope

The intent of the memo is to facilitate the use of CMIP to perform integrated management of networks via proxy management networks that are managed using SNMP. There are two major differences between the CMISE and SNMP services: one is the object model; and the other is the operations supported by the protocols. The ISO/Internet proxy architecture as shown in the above picture provides the CMISE service emulation. In another words, the ISO/Internet proxy acts as a CMIP agent with respect to the CMIP manager, allowing management of Internet objects by the ISO/CCITT manager. The CMIP requests would be processed by the ISO/Internet proxy and CMIP responses would be returned by the ISO/Internet proxy. SNMP traps would be converted to CMIP notifications by the ISO/Internet proxy. The implementation of the CMISE service emulation requires that a mapping exist between ISO/CCITT objects and Internet objects. There are different approaches that can be used to map between Internet objects and ISO/CCITT objects.

1) Direct Translation
2) Abstract Translation

The direct translation approach maps each Internet object to a newly defined ISO/CCITT object that contains: 1) the same information as contained in the Internet object; and 2) the attributes that are inherited from the ISO/CCITT Top object class.

The abstract approach maps the Internet objects to previously defined ISO/CCITT objects. For example, the MIB-II system object could be represented by the ISO/CCITT system object. No new object classes or attributes are needed with this approach.

Either or both approaches could be used by a ISO/CCITT manager to manage Internet agents. However, a large number

of Internet MIBs have been defined by the Internet community using the Internet SMI [RFC1155] and it is quite possible that there is no equivalent ISO/CCITT objects defined for some of the Internet MIBs at this point of time. For example, an Internet object and an ISO/CCITT object might be defined by different groups, and the attributes in the two objects could be significantly different. The mapping between the two objects could be very difficult, if it not impossible to perform.

This memo uses the "direct translation" approach".

To perform the CMISE service emulation, the ISO/Internet proxy can use either of the following two approaches to retrieve or modify Internet MIB information:

1. State-less approach
2. Stateful approach

The state-less approach would not maintain the Internet agent's MIB data. For each CMIP request, the ISO/Internet proxy would perform one or more SNMP requests.

The stateful approach would be to replicate the proxied agent's MIB data. The ISO/Internet proxy would replicate the Internet agent's MIB data locally. For each CMIP request, the ISO/Internet proxy would fulfill the request using data in its own local copy of the Internet MIBs.

The stateful approach will usually provide better response time, but has the drawback that the data retrived might not be up to date. This approach also requires a data synchronization mechanism between the ISO/Internet proxy and the proxied Internet agents. The frequency at which the MIB data in the ISO/Internet proxy is updated would have a significant effect on the accuracy of the response.

This memo uses the first approach. With that approach the ISO/Internet proxy performs the following operations: translation of CMIP requests to SNMP requests; translation of SNMP responses to CMIP responses; and translation of SNMP traps to CMIP notifications.

If necessary, the Internet MIB data retrived by the ISO/Internet proxy could be cached by the proxy in order to increase the response time of an operation. This memo makes no assumption that the proxy is maintaining any state information, and so takes no advantage of information which might be cashed.

This memo describes a protocol mapping process based on the IS CMIP (ISO/IEC 9596-2;1991) and on secure SNMP (RFC 1351 - RFC 1353). A protocol mapping process based on IS CMIP and RFC 1157 SNMP would be slightly different from the IS CMIP

to secure SNMP mapping process. These differences are described in Appendix A of this memo.

This memo assumes that the the CMIP PDU and SNMP PDU received by the ISO/Internet proxy are always properly encoded. The uderlying protocol providers should ensure the correctness of the service indications and confirmations that are passed up to the ISO/Internet proxy.

1.4. Terms and Conventions

1.4.1 ISO/CCITT manager: An application entity that implements ISO/IEC 9596-2;1991.

1.4.2 Internet agent: An application entity that is conformant to RFC 1352.

1.4.3 ISO/Internet Proxy: An application entity that is responsible of mapping CMIP requests to SNMP requests, SNMP responses to CMIP responses, and SNMP traps to CMIP notifications among any number of ISO/CCITT managers and Internet agents.

1.4.4 Known Internet agents: A set of one or more Internet agents that a ISO/Internet proxy has knowledge of. The known Internet agents could be represented by proxy object instances. The SNMP proxy object class is defined in [OIMTRANSMIB].

1.4.5 Known SNMP Parties: A set of one or more SNMP parties that a ISO/Internet proxy has knowledge of. The known SNMP parties could be represented by the SNMP party object instances. The SNMP party Object is defined in [OIMPARTY].

1.4.6 Pseudo Object: A CMIP object class that represents an SNMP group. The pseudo object does not contain any SNMP attributes. For example: a CMIP object that represents "at" in MIB-II, or any CMIP objects representing SNMP tables.

1.4.7 Access Policy: The SNMP version and the security policy.

1.4.8 Multiple Instance Object: An object class that may have more than one object instance. For example, SNMP table entries.

1.4.9 Delete Information: Delete information is the object identifier of the attribute and its attribute value used to indicate a particular row of a table is deleted.

2. ISO/Internet Proxy Configuration

In order for the ISO/Internet proxy to access each proxied Internet agent, and to access the proxied agents using the appropriate transport and network address, and using the appropriate access policy, the ISO/Internet proxy is required to have some knowledge about the proxied Internet agents.

2.1. Schema Information

To perform CMISE service emulation, the ISO/Internet proxy requires schema information for all of the Internet MIB definitions supported by a proxied agent. The schema information could be described to the ISO/Internet proxy using ISO/CCITT GDMO. Name binding definitions and matching rules are required if scoping and filtering are supported. The name binding definitions are needed to perform scoping and the matching rules are needed to perform filtering.

A translation procedure for converting Internet MIB and Internet trap definitions into corresponding ISO/CCITT GDMO definitions is described in [OIMMIBTRANS]. The proxy describes in this memo requires as input the schema information for any Internet MIB translated to ISO/CCITT GDMO format using the [OIMMIBTRANS] procedures.

The ISO/CCITT GDMO definitions based on RFC [OIMTRANSMIB] provide the object class, attribute, attribute syntax, name binding, and notification definitions. If the ISO/CCITT object class maps to an SNMP table entry, the behaviour clause (as defined in 3.3.1 of [OIMTRANSMIB] would describe the following information that is required by the ISO/Internet proxy:

a) the attributes that form the object instance;
b) whether or not the object is a multiple instance object (as defined in section 2.8 of this memo); and
c) if the object is a multiple instance object, the delete information for the object (as defined in section 2.9 of this memo).

2.2. Proxy Objects and Party Objects

RFC [OIMPARTY] defines some local object classes that are required by the ISO/Internet proxy to perform the CMISE service emulation.

The ISO/Internet proxy requires the following instances of the local object classes for each proxied Internet agent:

- one "cmipsxmpProxyEntry" object instance
- one "partyTable" object instance

- one or more "partyEntry" object instances

This memo refers to the above object instances as "local objects" or as "local object instances".

The "cmipsxmpProxyEntry" object instance contains the proxied SNMP agent's name and serves as a containment object. All the information related to a particular Internet agent would exist below the "cmipsxmpProxyEntry" in the managed information tree (MIT). This information includes all the Internet MIB objects that belong to a particular Internet agent and any configuration information needed by the ISO/Internet proxy (such as a partyTable object).

The "partyTable" object is the container object for all the "partyEntry's". Only one "partyTable" object instance should be created below the "cmipsxmpProxyEntry" object instance.

The "partyEntry" object contains the proxied Internet agent's address and the security mechanism used between the ISO/Internet proxy (acting as Internet manager) and the Internet agent. The security mechanism could either be an SNMP 1157 community-based or a secure SNMP party-based mechanism. A different "key" is stored in the "partyEntry" object, based on the chosen security mechanism.

The information described in the preceding paragraphs must be available to the ISO/Internet proxy before any CMIP requests or SNMP traps can be processed by the ISO/Internet proxy. The ISO/CCITT manager could provide this configuration information to the ISO/Internet proxy by sending M-Create requests to the ISO/Internet proxy in order to create the proxy and party objects.

If the CMIP request to create an SNMP proxy object succeeds, the ISO/Internet proxy adds the SNMP proxy object to its list of known Internet agents. If the CMIP request to create an SNMP party object succeeds, the ISO/Internet proxy adds the SNMP party object to its list of known SNMP parties.

It is the ISO/CCITT manager's responsibility to manage the known Internet agents and known SNMP parties. CMIP requests could be initiated by the ISO/CCITT manager to add, delete, or modify elements in the known Internet agents and the known SNMP parties.

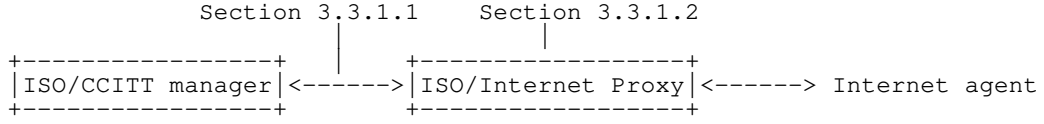
2.3. Usage

The known Internet agents, and the known SNMP parties provide the basis for the ISO/Internet proxy to perform the following translation:

- Translating CMIP requests to SNMP requests
- Translating SNMP responses to CMIP responses
- Translating SNMP traps to CMIP notifications

In a CMIP request, the "cmipsxmpProxyEntry" object is represented by one of the relative distinguished names (RDN) that make up a distinguished name (DN). In the MIT, there are one or more party objects contained under the "cmipsxmpProxyEntry" object.

2.3.1. Security



2.3.1.1. The Security Policy between the ISO/CCITT manager and the ISO/Internet Proxy

The information transferred between the ISO/CCITT manager and the ISO/Internet proxy is protected by the security policy between the ISO/CCITT manager and the ISO/Internet proxy (acting as a CMIP agent). The security policy between an ISO/CCITT manager and a CMIP agent is not part of the scope of this memo.

2.3.1.2. The Security Policy between the ISO/Internet Proxy and the Internet agent

The information transferred between the ISO/Internet proxy and the Internet agent is protected by a security policy between the ISO/Internet proxy and the Internet agent. In order to minimize the network traffic between the ISO/Internet proxy and the Internet agent, and also to reduce the number of party objects, this security policy should be enforced using the following two distinct steps:

2.3.1.2.1. Local Security Policy

First, a local security policy that maps a CMIP request to a source party and a destination party based on the distinguished name, the access control field; and possibly the operation type and the attributes in the attribute list.

The security mapping process should be based on a local security policy that is mutually agreed to and implemented by the ISO/CCITT manager and ISO/Internet proxy. The local security policy is not within the scope of this memo. A separate memo to define this security policy may be produced after the secure SNMP and SMP are finalized.

The ISO/Internet determines a pair of source party and destination party contained in the "cmipsxmpProxyEntry" object based on the DN, access control parameter, and possibly the operation type, and attribute list. The method of selecting the party pair is a local issue.

If this mapping step fails, the ISO/Internet proxy must send an "access denied" error response back to the ISO/CCITT manager. An SNMP request will not be generated by the proxy.

2.3.2. Packaging SNMP Request

Second, if the previous step is successful, a source party and a destination party are chosen as the basis for packaging the SNMP request. The selected SNMP source party and destination party specifies the SNMP protocol version and the SNMP security protocol to use with the SNMP request generated by the ISO/Internet Proxy.

SNMP request message should be encoded based on the secure SNMP protocol using the selected source party and destination party.

3. Elements of CMISE Service Emulation

The following sections describe the conceptual process for performing CMIP service emulation. In an actual implementation, it should be possible to combine some of the processes. It is highly recommended that the implementor of the ISO/Internet proxy combine the processes where possible to to optimize the implementation. If the ISO/Internet proxy needs to perform an SNMP request to fulfill a CMIP request, the ISO/Internet protocol translation that would be used to translate the request is described in section 4 of this memo.

3.1. Association Service

The ISO/Internet proxy should provide the association service defined in section 8.1 of ISO/IEC 9596-2;1991. This service includes association establishment and association release. Each CMIP request issued within the context of a specific association, should only be performed if the request specifies functional units negotiated for the association.

3.2. Management Object Selection - Scoping and Filtering

Managed object selection is used to determine a set of object instances in the MIT on which a CMIP request is to operate. Managed object selection is performed in two phases: scoping; and then, filtering. Scoping is used to

specify a sub-tree of object instances in the MIT. A filter is then applied to the set of previously scoped object instances in order to identify a subset of object instances on which the CMIP operation is to be performed.

The scope specified in the CMIP request is applied to determine a set of object instances against which a filter (if specified) is to be applied. If no filter is specified the CMIP request will be performed for all object instances identified by the scope.

There are different ways of performing scoping operation. This memo specifies one possible way of providing managed object selection service.

3.2.1. Object Class Group

The following algorithm specifies a method of determining an "object class group" that includes a set of object class object identifiers. Each of the object instance within the scope, is an instance of an object class that is in the object class group.

The variables used in these steps are fictitious variables used to control the process, not attributes or objects.

Step 1:

"current level" = 0
"current level object classes" = the base object specified
in the CMIP request
"next level object classes" is set to empty
"object class group" is set to empty

Step 2:

If the "current level" is greater than the maximum level relative to the base object in the scope or the "current level object classes" is empty, go to step 5.

Step 3.

If the "current level object classes" is empty, go to step 4.

Take the first object class in the "current level object classes". This object class is called "current object class"

For all the known name bindings in the proxy's schema, if the "NAMED BY SUPERIOR OBJECT CLASS" is equal to the "current object class", the "SUBORDINATE OBJECT CLASS" is added into "next level object classes".

If the current level is greater than the minimum level and smaller than the maximum level in the scope, the "SUBORDINATE OBJECT CLASS" is added into "object class group".

Interate through step 3 again.

Step 4:

"current level object classes" = "next level object classes"
"next level object classes" is set to empty
bump up "current level" by one

Iterate through step 2 again.

Step 5:

If the "object class group" is empty, there is no object instance selected by the scope. In this case, an empty final CMIS response should be returned.

If the "object class group" is not empty, for each object class in the "object class group", the following three steps should be taken to retrieve the object instances in the scope:

3.2.2. If the object class is a multiple instance object, the first instance (.i.e., the first row in the table) should be retrieved.

If the object class is not a multiple instance object, retrieve the object instance of the object class.

3.2.3. If a filter is specified, apply the filter to the retrived value. If the filter evaluates to FALSE, the CMIP operation will not be performed for the object instance. If the filter evaluates to TRUE, the operation will be performed for the instance.

3.2.4. If the object instance is a multiple instance object, retrieve succeeding instances (i.e., the succeeding rows in the table). If a filter is specified, apply the filter as specified in 5.2.2. Repeat this process until all instances of the multiple instance object (i.e. all rows) have been retrieved.

The scoping process could be highly optimized by local means to retrieve information in multiple ISO/CCITT object instances at once.

The filtering process should also be optimized where it is applicable. For example, if the attribute in the filter item does not exist according to the object class definition, the ISO/Internet proxy should evaluate the

filter as FALSE without trying to retrieve the attribute from the object instance. If the CMIP request is a get request, the filter process described above may also be combined with the retrieval of the attributes specified in the attribute id list.

3.3. Synchronization

If the ISO/Internet proxy receives a CMIP "atomic" request, but can not perform the operation atomically, the "synchronization not supported" CMIP error response should be returned to the ISO/CCITT manager.

The types of atomic requests that the ISO/Internet proxy can perform are
as follows:

1) if all the instances selected by a scoped CMIP request are "local object instances", then the ISO/Internet proxy can perform the CMIP request locally (and atomically); and

2) if all the CMIP request can be performed by the ISO/Internet proxy using a single SNMP request, then the operation can also be performed atomically.

For a "best effort" request, the ISO/Internet proxy should try to perform the request on all the instances specified by the request. In some cases, the CMIP request has to be translated into multiple SNMP requests in order to fulfill the original CMIP request. In the window in which these SNMP requests are being processed, another SNMP set request could be issued which could modify data in instances specified by the scope. Because of this, the complete integrity of the scoped request can not be guaranteed. A proxy which complies with this memo is not required to detect or avoid this situation, and will not usually report any error if this situation occurs.

3.4. Management Operation Services

If the specified instances (i.e., those identified by scoping and filtering specified in a CMIP request) are "local object", the ISO/Internet proxy would perform the services using local means. Instances of "partyTable", and "partyEntry" are considered to be "local objects".

If the specified instances are "remote objects", then the following sections apply. Any objects that physically reside in the SNMP agents are considered as "remote objects". For example, any MIB II objects like system, tcp, and upd ate considered as "remote objects".

3.4.1. M-GET Service

The following sections describe how to provide M-GET service.

3.4.1.1. Check the Existence of the Base Object

The existence of the ISO/CCITT base object specified in a CMIP request should always be verified before scoping is started. If the base object specified in the request does not exist in the Internet agent, then the proxy must send a "NoSuchObjectInstance" CMIP error response back to the ISO/CCITT manager.

If the base object is a table entry or an SNMP group with at least one Internet object type (i.e., not a pseudo object), the ISO/Internet proxy should try to retrieve at least one attribute of the base object from the SNMP agent to verify the existence of the base object.

If the base object is a pseudo object, the ISO/Internet proxy should try to retrieve at least one attribute from the first subordinate object in the MIT that is not a pseudo object, in order to verify the logical existence of the base object (pseudo object).

3.4.1.2. Perform the Get Operation

If scoping or filtering is specified, the process described in section 4.2 of this memo is used to select the ISO/CCITT object instances. For each selected ISO/CCITT object instance, all of the attributes specified in the "Attribute identifier list" parameter of the CMIP request are retrieved. If the "Attribute identifier list" parameter is omitted, all attributes of the instance are of retrieved. The list of attributes are in an object class are included in the definition of the object class. It is recommeded that the retrieval of the attributes be combined with the retrieval of the attributes specified by the filter.

Section 5.4 describes the CMIP M-GET to SNMP get protocol mapping.

3.4.1.3. Form the Response(s)

The CMIP M-GET response should contain the attribute value list or the appropriate get list error. Once the M-GET response has been formatted, it should be passed to the CMIP service provider.

If the CMIP get request is a scoped request, each ISO/CCITT object should be reported as link reply. a final empty CMIS M-GET response should be returned to the ISO/CCITT manager when the scoped request completes.

The managed object instance in the CMIP get response should use the the distinguished name (DN) specified in the original CMIP request as a prefix. The get responses for subordinate objects below the base object in the naming tree, should contain DN's that are formed from the concatenation of the DN from the original CMIP request and the RDN's specifying the table entry deleted.

3.4.2. M-CANCEL-GET Service

The CMIP cancel get operation should be performed as described in ISO/IEC 9596. The ISO/Internet proxy does not need to translate this request to any SNMP requests.

After receiving a cancel-get request message the ISO/Internet proxy should terminate any outstanding communications with the Internet agent that were initiated as a result of the get request that is being canceled. Once the cancel-get has been processed and responded to, the ISO/Internet proxy should not send any additional response messages to the ISO/CCITT manager for the get operation that was canceled.

3.4.3. M-SET Service

The following sections describe how to provide M-SET service.

3.4.3.1. Check the Existence of the Base Object

This operation is the same as described in section 4.4.1.1.

3.4.3.2. Scope and Filter

This operation is the same as described in section 4.4.1.2.

3.4.3.3. Perform the Set Operation

For each selected ISO/CCITT object instance, the attributes specified in the "Modification list" parameter of the CMIP request are modified based on each attribute's modify operator. Only the "replace" modify operator is supported by the ISO/Internet proxy. The modify operator is optional and if it is not specified in a CMIP request, the "replace" operator should be assumed.

The "add value" and "remove value" modify operators are not supported by SNMP protocol, and are not supported by the ISO/Internet proxy. The "set to default" modify operator is not supported by the ISO/Internet proxy since the ISO/Internet proxy is acting as an Internet manager. The "set to default" behaviour is optionally implemented by the Internet agent, not the Internet manager. If the modify

operator is not supported, "invalid operator" should be reported in the set list error.

Section 5.5 of this memo describes the CMIP M-SET to SNMP set protocol mapping.

3.4.3.4. Form the responses

The CMIP M-SET response should contain the attribute value list or the appropriate set list error. Once the M-SET response has been formatted, it should be passed to the CMIP service provider.

If the CMIP set request is a scoped request, each ISO/CCITT object should be reported as link reply. A final empty CMISE M-SET response should be returned to the ISO/CCITT manager when the scoped request completes.

The managed object instance in the CMIP set response should use the the distinguished name (DN) specified in the original CMIP request as a prefix. The set responses for subordinate objects below the base object in the naming tree, should contain DN's that are formed from the concatenation of the DN from the original CMIP request and the RDN's specifying the table entry deleted.

3.4.4. M-ACTION Service

The proxy is not required to emulate the CMIS M-ACTION service because Internet MIBs do not include any definitions which translate into M-ACTIONs in an ISO/CCITT GDMO MIB. Any CMIS M-ACTION request which is received pertaining to an Internet MIB object will be rejected by the proxy with an invalidOperation error response. However, CMIS M-ACTION may be used by the proxy for other purposes

3.4.5. M-CREATE Service

3.4.5.1. Request Validation

The ISO/Internet proxy is responsible for validating that the CMIP create request does not violate name binding and object class definitions.

3.4.5.1.1. Name Binding

The ISO/Internet proxy must determine from the CREATE parameter in a name binding if an instance may be created. If the instance cannot be created, an "access denied" error response should be returned.

The ISO/Internet proxy must also determine from the name binding if the instance specified in the request may be created under the specified superior in the MIT. If the

name binding does not allow the specified containment relationship, an "invalidOperation" error response should be returned.

3.4.5.1.2. Check for Duplication

Determine if the instance already exists. If it does, a "duplicate managed object instance"i error response should be returned.

See 4.4.1.1. of this memo for checking the existence of an object.

3.4.5.2. With Referenced Object

If a CMIP create request specifies a reference object, the ISO/Internet proxy should retrieve the referenced object from the Internet agent. If the reference object does not exist, the proxy must send a "No such reference object" error response back to the ISO/CCITT manager.

3.4.5.3. With Automatic Instance Naming

A CMIP create request can use automatic instance naming to form a name for the object instance to be created. Automatic instance naming is used if: a) a CMIP create request does not specify a distinguished name for the object instance to be created; and b) the request specifies an object class that has a name binding allowing automatic instance naming.

It is the responsibility of the ISO/Internet proxy to form the name based on the behavior of the object class. In some cases, the relative distinguished name (RDN) is formed using attributes provided in the CMIP request. For example, the RDN for "atEntry" in MIB-II could be formed from the "atNetIfIndex" attribute and the "atNetAddress" attribute. In other cases, the RDN can be assigned by the ISO/Internet proxy.

If the superior object instance is not specified, the ISO/Internet proxy can not create the object instance and "processing failure" error should be returned.

3.4.5.4. Perform the Create Operation

If the combination of the attributes specified in the CMIP request and the attributes obtained from the reference object do not provide attribute values for all of the mandatory attributes for the entry specified by the object class being instantiated, then the ISO/Internet proxy should still try to create the object with all the available attributes.

If the actual creation process based on the available but incomplete attribute list succeeds, the ISO/Internet proxy should retrieve all the attributes on the newly created entry since the Internet agent may fill in the missing attributes with the Internet agent's own default values. If the missing attributes are provided by the agent, the CMIP create response should include the attributes that were provided by the Internet agent in its CMIP response.

If the actual creation process based on the available but incomplete attribute list fails, the ISO/Internet proxy should send a "missing attribute value" error back to the ISO/CCITT manager.

See 3.4.1.1. of this memo for checking the existence of an object.

3.4.5.5. Form the responses

The CMIP M-CREATE response should be formatted and then should passed to the CMIP service provider.

The managed object instance in the CMIP set response should use the the distinguished name (DN) specified in the original CMIP request as a prefix. The set responses for subordinate objects below the base object in the naming tree, should contain DN's that are formed from the concatenation of the DN from the original CMIP request and the RDN's specifying the table entry deleted.

3.4.6. M-DELETE Service

3.4.6.1. Check the Existence of the Base Object

This operation is the same as described in section 3.4.1.1.

3.4.6.2. Scope and Filter

This operation is the same as described in section 3.4.1.2.

3.4.6.3. Perform the Delete Operation

For all the selected ISO/CCITT object instances, the following procedures should be taken:

3.4.6.3.1. Name binding

Determine from the name binding if the instance specified in the request may be deleted. If the name binding does not allow the specified delete operation, "access denied" error response should be returned.

3.4.6.3.2. Check the Existence of the Selected Object

For each selected object, the following steps should be taken.

If the selected object is a table entry or an SNMP group with at least one Internet object type (not a pseudo object), the ISO/Internet proxy should try to retrieve at least one attribute of the object from the SNMP agent to verify the existence of the selected object.

If the base object is a pseudo object, the ISO/Internet proxy should try to retrieve at least one attribute from the first subordinate object in the MIT that is not a pseudo object, in order to verify the logical existence of the base object (pseudo object).

3.4.6.3.3. Perform the Delete Operation

If the object instance specified in the CMIP delete request exists, the delete operation is performed.

3.4.6.3.4. Form the responses

Format the CMIP M-DELETE response with the appropriate attribute list or delete list error. Once the M-DELETE response has been formatted, it should be passed to the CMIP service provider.

The managed object instance in the CMIP delete response should use the the distinguished name (DN) specified in the original CMIP request as a prefix. The delete responses for subordinate objects below the base object in the naming tree, should contain DN's that are formed from the concatenation of the DN from the original CMIP request and the RDN's specifying the table entry deleted.

3.5. Management Notification Services

All the SNMP traps should be mapped to CMIP notifications. Only unconfirmed mode is used since the SNMP traps are not confirmed.

An SNMP trap is translated to a single CMIP event report. The following procedures should be used to translate an SNMP trap to a CMIP event report.

3.5.1. Object Class

The object class parameter in the CMIP event report should be set to the appropriate object class as defined in the object class definitions.

3.5.2. Object Instance

The object instance parameter in the CMIP event report is determined from information provided in the SNMP trap. The SNMP trap sender's transport type, transport address, and network address are used to determine the DN of the proxy object instance.

3.5.3. Event Time

The event time parameter in the CMIP event report should be calculated based on the time stamp field in the SNMP trap.

3.5.4. Event Type

The event type parameter in the CMIP event report should be determined by the generic trap field and enterprise specific trap field in the SNMP trap.

3.5.5. Event Info

The event type parameter in the CMIP event report should be determined from a Notification definition.

3.5.6. If any of the translations in the preceding sections fail, no CMIP event report is produced.

3.6. Common Response Parameter

This memo does not specify a standard translation for the timestamp value in the CMIP response. However, the following paragraphs describe two potential implementations for this translation:

- ISO/Internet proxy's local time
- Internet agent's local time with fixed unknown delta

3.6.1. ISO/Internet Proxy's Local Time

The timestamp value in the CMIP response is set to the time provided by the ISO/Internet proxy's internal clock when a request's final SNMP response is received.

3.6.2. Internet agent's local time with fixed unknown delta

The ISO/Internet proxy should query the Internet agent for "sysUpTime" in addition to the original SNMP variable binding list in the first SNMP request. The value should be recorded as the "agent's initial sysUpTime" and the ISO/Internet proxy's local time should be recorded as "initial contact time".

Each CMIP request is translated to one or more SNMP requests by the ISO/Internet proxy to fulfill the CMIP request. If the last SNMP request for the same CMIP request is an SNMP get request, the "sysUpTime" should be added into the SNMP

variable binding list. Otherwise, an extra SNMP get request is issued with "sysUpTime" as the only element in the variable binding list. This new "sysUpTime" is called "agent's current sysUpTime".

The timestamp in the last CMIP response should be calculated using this formula: "initial contact time" + (agent's current sysUpTime - agent's initial sysUpTime").

This approach eliminates the inaccuracy caused by network delay between the ISO/Internet proxy and Internet agent and gives the ISO/CCITT manager a more accurate time on when an operation is actually performed by the Internet agent rather than the time that is processed by the ISO/Internet proxy.

4. CMIP and SNMP Protocol Mapping

To retrieve and modify SNMP entries, the ISO/Internet proxy is required to perform CMIP and SNMP protocol translation. A CMIP request is translated to one or more SNMP requests. SNMP responses are translated to CMIP responses. SNMP traps are translated to CMIP notifications.

4.1. Management Object Selection - Scoping and filtering

A managed object selection process is described in section 4.2 of this memo. Section 3.2.2 and 3.2.4 state:

> 3.2.2. If the object class is a multiple instance object, the first instance (.i.e., the first row in the table) should be retrieved.

The ISO/Internet proxy would issue SNMP get-next requests to retrieve the instance information from the table.

Each attribute in the object class is used to form a variable binding in the variable binding list in the SNMP request. The attributes in the object class are maintained by the ISO/Internet proxy locally.

Each attribute in the object class is used as the "name" field in the variable binding. The ISO/Internet proxy composes a variable binding list and then encodes an SNMP get-next request.

> 3.2.4. If the object instance is a multiple instance object, retrieve succeeding instances (i.e., the succeeding rows in the table). If a filter is specified, apply the filter as specified in 3.2.2. Repeat this process until all instances of the multiple instance object (i.e. all rows) have been retrieved.

The ISO/Internet proxy should issue SNMP get-next requests to retrieve the succeeding object instance. The variable binding list is the same as the the variable binding list in the previous SNMP get response (that resulted from the previous get-next request).

If the SNMP error, noSuchName, occurs when an attribute is queried as part of a filter evaluation, then the filterItem will be evaluated as FALSE.

4.2. Check for Existence of the Object to be Created

Section 3.4.1.1 of this memo:

> If the base object is a table entry or an SNMP group with at least one Internet object type (i.e., not a pseudo object), the ISO/Internet proxy should try to retrieve at least one attribute of the base object from the Internet agent to verify the existence of the base object.

The ISO/Internet proxy should issue an SNMP get request to retrieve at least one attribute in the base object. The variable binding is composed using information specified in the CMIP request.

> If the base object is a pseudo object, the ISO/Internet proxy should try to retrieve at least one attribute from the first subordinate object in the MIT that is not a pseudo object, in order to verify the logical existence of the base object (pseudo object).

The ISO/Internet proxy should issue an SNMP get-next request is used to retrieve at least one attribute from the first subordinate object in the naming tree that is not a pseudo object.

In both of the above cases, a variable binding is needed in the SNMP request. The variable binding contains "name" and "value". Only "name" is significant in the SNMP request.

4.2.1 SNMP Entry

If the base object is an "SNMP entry", the first component in the "name" field is taken from any attribute in the object class. The object class is specified in the CMIP request and the attribute is obtained from the object class definition by the ISO/Internet proxy using local means. The process specified in section 5.8 of this memo could be used to form the variable binding. Once the variable binding has been formed, an SNMP get request is issued to get the attribute value, which will validate the existence of the base object.

4.2.2. Pseudo Entry

If the base object is a "pseudo entry", there is no SNMP entry that maps to this ISO/CCITT entry. Therefore the first SNMP subordinate entry of this base object is queried to verify the existence of this base object.

The "name" field is taken from the object class object identifier directly to form the SNMP get-next request.

If the "name" field in the variable binding in the SNMP get response contains the same object identifier prefix as the request, the ISO/Internet proxy can conclude that this pseudo entry should exist. If the "name" field in the SNMP request and the "name" field in the SNMP response does not have the same object identifier prefix, the ISO/Internet proxy can conclude that the pseudo entry does not exist. The same object identifier prefix means that the "name" field, which is a object identifier, in the response should contain the same object identifier as the one in the request with one or more additional digits

If SNMP "no such name" error is received, the ISO/Internet proxy can conclude that the pseudo entry does not exist. Any other error should refer to section 5.9 of this memo.

Note. The same method can also be used to verify the existence of an SNMP "group" entry.

4.3. Create With Referenced Object

An SNMP get request should be issued to obtain the attributes in the referenced object.

Each attribute in the object class forms a variable binding in the variable binding list in the SNMP request. The attributes in the object class id maintained by the ISO/Internet proxy locally.

For each attribute in the object class, the process described in section 5.8 of this memo should be taken to form the variable binding. The ISO/Internet then compose a variable binding list.

If SNMP "no such name" error is received, the ISO/Internet proxy can conclude that the pseudo entry does not exist. Any other error should refer to section 5.9 of this memo..

4.4. Perform the Get Operation

For each selected attribute, the ISO/Internet proxy should use the process defined in 5.8 of this memo to form the variable binding. The ISO/Internet proxy should then form

the variable binding list and then issue an SNMP get request to perform the get operation.

4.5. Perform the Set Operation

For each attributes, the ISO/Internet proxy should use the process defined in section 5.8 of this memo to form the variable binding. The "value" field will be taken from the modification list. The ISO/Internet proxy should then form the variable binding list and then issue an SNMP set request to perform the set operation.

4.6. Perform the Create Operation

For each available attributes, the combination of the attributes specified in the CMIP request and the attributes obtained from the reference object, the ISO/Internet proxy should use the process defined in section 5.8 of this memo to form the variable binding list. The "value" field will be taken from either the CMIP request or the referenced object.

4.7. Perform the Delete Operation

The ISO/Internet proxy needs to determine the attribute type and attribute value that indicates an entry has been deleted. This information is maintained locally by the ISO/Internet proxy.

Each object selected by the CMIP delete request is translated to a variable binding. All of the selected objects could be translated into variable bindings within one SNMP set request. If the request is too big, the ISO/Internet proxy should divide the request to smaller requests.

For each to-be-deleted object, the ISO/Internet proxy should use the process defined in section 5.8 of this memo to form the variable binding. The "value" field will be filled in using the attribute value that indicates an entry has been deleted.

4.8. Form a Variable Binding

The variable binding contains two fields: "name" and "value". The "name" field is composed by the ISO/Internet proxy based on its object class, attribute, and a RDN.

The "name" field in the variable binding is formed from the concatenation of two components.

The first component in the "name" field is the attribute's object identifier.

There are two cases for the second component in the "name" field.

If the object class is a "single instance" object, (i.e., an SNMP group or an SNMP table), the digit "0" is used for the second component.

If the object class is a "multiple instance" object (i.e., a table entry), and if a RDN is supplied, the Internet object instance might be derived from the RDN. An RDN consists of an attribute type and an attribute value. The attribute value of the RDN has two parts. The first part of the value is the object class object identifier. The second part of the value is the Internet object instance. This second part of the value is used as the second component in the "name" field of the variable binding.

Example 1: single instance object

Input: Object class: system
Attribute: sysName
attribute value in the RDN: system

The SNMP variable binding: sysName.0

Example 2: multiple instance object

Input: Object class: ipRouteEntry
Attribute: ipRouteDest
attribute value in the RDN: ipRouteEntry.192.22.83.97

The SNMP variable binding: ipRouteDest.192.22.83.97

4.9. SNMP error to CMIP error mapping

SNMP error responses received by the ISO/Internet proxy are translated to CMIP error responses and sent back to the ISO/CCITT manager. The sections that follow list the SNMP errors that could be received, and describes which CMIP error responses they will be translated to.

4.9.1. tooBig

If the SNMP error, tooBig, is received, the ISO/Internet proxy should try to break the SNMP request to smaller requests and resend the requests. If it is not feasible to break the request to any smaller request, and this error occurs, the CMIP error response, "Resource limitation", should be returned to the ISO/CCITT manager.

4.9.2. noSuchName

If the SNMP error, noSuchName, occurs when an attribute is queried as part of a filter evaluation, then the filterItem will be evaluated as FALSE.

In order to check if an object exists, all the object class's attributes should be queried until at least one attribute's existence is verified. if none of the attributes exist, and the object is the base object, then a "NoSuchObjectInstance" CMIP error response should be returned.

If the object exists and the SNMP error, noSuchName, occurs when attempting to modify an attribute while processing the attribute list, a "No such attribute" CMIP error response should be returned to the ISO/CCITT manager.

If the ISO/Internet proxy maintains correct schema information and the SNMP agent is a conformant agent, an Internet object's attributes should all exist or not exist. In order to make the ISO/Internet proxy a practical solution, the preceding error situation is included in order to deal with a non-conformant Internet agent.

4.9.3. badValue

If the SNMP error, badValue, is returned for an SNMP get request, then a "processing failure" CMIP error response should be returned to the CMIP manager. In the ProcessingFailure parameter of the CMIP error response, the errorId should be snmpBadValue, and the errorInfo should be the variable binding identified by the error-index.

If the badValue error occurs during an SNMP set request to fulfil a CMIP delete request, a "processing failure" CMIP error response should be returned. In the ProcessingFailure parameter, the errorId should be canNotDelete and the errorInfo should be the variable binding that is identified by the error-index.

4.9.4. readOnly

If the SNMP error, readOnly, occurs when checking for the existence of a base object, a "processing failure" CMIP error response should be returned to the ISO/CCITT manager. In the ProcessingFailure parameter of the CMIP error response, the errorId should be snmpReadOnly, and the errorInfo should be the variable binding identified by the error-index.

If the readOnly error occurs when deleting the object, then the deleteErrorInfo in the delete response should be set to "access denied".

4.9.5. genErr

If the SNMP error, genErr, occurs, the "ProcessingFailure" CMIP error response should be returned to ISO/CCITT manager. If the entry exists, scoping continues; otherwise, scoping terminates. In the ProcessingFailure parameter of the CMIP error response, the errorId should be snmpGenErr.

5. CMIP Processing Failure

There are many error scenarios in which the error can not be mapped to a specific CMIP error. In this case, the "processing failure" CMIP error response should be reported back to the ISO/CCITT manager. In order to provide the ISO/CCITT manager a better description of the error, the specificErrorInfo field in ProcessingFailure is used to record the cause of the problem.

The following object identifiers are defined:

```
errors         OBJECT IDENTIFIER :: { oim 11 }

snmpTooBig     OBJECT IDENTIFIER :: { errors 0 }
snmpBadValue   OBJECT IDENTIFIER :: { errors 1 }
snmpReadOnly   OBJECT IDENTIFIER :: { errors 2 }
snmpGenErr     OBJECT IDENTIFIER :: { errors 3 }
noResponse     OBJECT IDENTIFIER :: { errors 4 }
canNotDelete   OBJECT IDENTIFIER :: { errors 5 }
notImplemented OBJECT IDENTIFIER :: { errors 6 }
```

For errorId snmpTooBig, the errInfo is defined as VarBindList.
For errorId snmpBadValue, the errInfo is defined as VarBind.
For errorId snmpReadOnly, the errInfo is defined as OBJECT IDENTIFIER.
For errorId canNotDelete, the errInfo is defined as VarBind.
For errorId notImplemented, the errInfo is defined as

```
INTEGER {
          filter(0),
          scope(1),
          transport(2),
          AuthenticationProtocol(6),
          PrivacyProtocol(7)
        }
```

6. Functional Units

A ISO/Internet proxy implementation that claims conformance to this memo must state the functional units that it supports. Two functional units are defined:

a) a minimum proxy functional unit; and

b) an additional scoping functional unit.

The minimum proxy functional unit requires the support of create operations, scoped get operations, base-object-only set operations, base-object-only delete operations, and notifications.

The additional scoping functional unit requires the support of scoped set operations and scoped delete operations.

This memo assigns the following object identifier:

{ cmipsxmpProxy aAssociateUserInfo(7) }
Editor's note: cmipsxmpProxy is specified in [OIMTRANSMIB].

as a value of ASN.1 type FunctionalUnitPackageId defined in CCITT Rec. X.701 | ISO/IEC 10040 to use for negotiating the following functional units:

0 minimum proxy functional unit; and
1 additional scoping functional unit.

Where the above number identifies the bit position in the BIT STRING assigned to the functional units.

Editor's note: Is functional unit appropriate? Maybe some sort of compliance sentence is more appropriate? Is this section needed at all? Suggestions?

7. Abbreviations

CMIP: Common Management Information Protocol
MIB: Management Information Base
DN: Distinguished Name
RDN Relative Distinguished Name
SNMP: Simple Network Management Information Protocol

8. Acknowledgements

The following individuals provided valuable comments on the memo prior to its initial distribution.

Dean Voiss, NetLabs
Jon Biggar, NetLabs
Lee LaBarre, MITRE
Lisa Phifer, Bellcore
Steve Ng, MPR Teltech

References

[ISO8824] ISO/IEC IS 8824: Information Technology - Open System Interconnection - Specification of Abstract Syntax Notation One (ASN.1),1990.

[ISO10165-4] ISO/IEC IS 10165-4: Information Technology - Open Systems Interconnection - Structure of Management Information - Part 4: Guidelines for the Definition of Managed Objects, 1991.

[ISO9595] ISO/IEC IS 9595, Information Technology - Open Systems Interconnection - Common Management Information Service Definition, 1991.

[ISO9596-1] ISO/IEC IS 9596-1, Information Technology - Open Systems Interconnection - Common Management Information Protocol - Part 1: Specification, 1991.

[RFC1155] RFC1155, M. Rose and K. McCloghrie, Structure and Identification of Management Information for TCP/IP-based internets, May 1990.

[RFC1157] RFC 1157, J.D. Case, M.S. Fedor, M.L. Schoffstall, C. Davin, Simple Network Management Protocol (SNMP), May 1990.

[RFC1351] RFC 1351, Davin, J., Galvin, J., and K. McCloghrie, "SNMP Administrative Model", MIT Laboratory for Computer Science, Trusted Information Systems, Inc., Hughes LAN Systems, Inc., July 1992.

[RFC1352] RFC 1352, Galvin, J., McCloghrie, K., and J. Davin, Trusted Information Systems, Inc., Hughes LAN Systems, Inc., MIT Laboratory for Computer Science, July 1992.

[RFC1353] RFC 1353, McCloghrie, K., Davin, J., and J. Galvin, "Definitions of Managed Objects for Administration of SNMP Parties", Hughes LAN Systems, Inc., MIT Laboratory for Computer Science, Trusted Information Systems, Inc., July 1992.

[OIMTRANSMIB] Lee LaBarre, ISO and Internet Management Coexistence (IIMC): Translation of Internet MIBs for ISO/Internet Proxy. The MITRE Corporation, September 92.

[OIMPARTY] Lee LaBarre, ISO and Internet Management Coexistence (IIMC): Proxy MIB for the SNMP Party MIB. The MITRE Corporation, September 92.

[OIMMIB-II] Lee LaBarre, ISO and Internet Management Coexistence (IIMC): Proxy Management Information Base II for TCP/IP Networks. The MITRE Corporation, September 92.

[IIMCOMIBTRANS] O. Newnan, ISO/CCITT and Internet Management Coexistence: Translation of ISO/CCITT GDMO MIBs to Internet MIBs, October 1992.

Appendix A: CMIP to RFC 1157 Agent proxy

The philosophy of implementation for either a ISO/Internet proxy that acts as a proxy for an RFC 1157 Internet agent, or for a proxy for a secure Internet agent, is quite similar. The following sections describe the differences between the two types of proxy agents.

1. SNMP Protocol Selection

The partyauthprotocol attribute in the selected SNMP party object indicates which SNMP protocol is to be used. If the community protocol is selected, the ISO/Internet proxy encodes the SNMP message based on the definition in RFC 1157. If the md5 authentication protocol is selected, the ISO/Internet proxy encodes the SNMP message based on the definition in the secure SNMP memos.

2. Security Protocol Selection

If the attribute, partyAuthProtocol, indicates that the community protocol is to be used, the attribute, partyAuthPriv, provides the value for the community string that is to be used to form the SNMP request.

If the attribute, partyAuthProtocol, indicates that the md5 authentication protocol is to be used, the SNMP message should be encoded using the appropriate authentication and encryption mechanisms based on the definition in the secure SNMP memos.

3. Internet agent Address Resolution

For each of the CMIP requests received by the ISO/Internet proxy, an SNMP source party and a destination party are selected based on the DN and access control fields in the CMIP request. The SNMP party object selected is the source party object contains the Internet agent's address information in the following two attributes: partyTDomain; and partyTaddr.

The attribute, partyTDomain, specifies the transport mechanism. The attribute, partyTAddr, specifies the

transport and network address. The address format of the partyTaddr is determined from the value of the partyTDomain attribute.

4. readyOnly error

If the Internet agent that the ISO/Internet proxy represents is an RFC 1157 community-based Internet agent, the the SNMP readOnly error should never be received by the proxy. If this error is received, a "processing failure" CMIP error response should be returned to the ISO/CCITT manager. In the ProcessingFailure parameter, the errorId should be snmpReadOnly and the errorInfo should be the variable binding that is identified by the error-index.

Appendix B: Example Operation

```
Operation: CMIP get request
Object class: ip
Object instance: {cmipsxmpProxyId = "NetLabs" }
                 {internetClassId = ip }
Scope: 2nd level down
Filter: ipRouteType == indirect
Attribute id list: ipRouteDest
```

Example SNMP ipRouteEntries:

ipRouteDest	ipRouteType	Other object types
192.95.93.1	direct	
192.95.93.2	indirect	
192.95.93.3	invalid	
192.95.93.4	other	
192.95.93.5	indirect	
(end of the table)		

The following sections show an example of how the ISO/Internet proxy fulfills the above CMIP get request based on the example SNMP ipRouteEntries.

1. Check the existence of the base object:

The ISO/Internet proxy issues a SNMP get next request.

```
GetNextRequest { ip }
GetResponse { ipForwarding = 1 }
```

If the get is successful, the ISO/Internet proxy would have verified that the base object exists.

2. Managed object selection

Based on the name binding definition, the following object classes are found in the "object class group":

a) ipAddrEntry
b) ipRouteEntry
c) ipNetToMediaEntry

For each object in the "object class group", the object instances may be retrieved via SNMP get next requests.

a) ipAddrEntry: The object class definition for this object class does not contain the attribute specified in the filter (ipRouteType), therefore the ISO/Internet proxy will conclude that there is no object instances with ipAddrEntry object class in the scope.

b) ipRouteEntry: The object class definition for this object class contains the attribute specified in the filter (ipRouteType), therefore the ISO/Internet proxy would issue SNMP get next request to retrieve the object instances. The SNMP requests issued and the responses received would be as follows:

```
GetNextRequest {ipRouteDest, ipRouteType}
GetResponse {ipRouteDest.192.94.93.1 = 192.94.93.1,
             ipRouteType.192.94.93.1 = direct}

GetNextRequest {ipRouteDest.192.94.93.1,
                ipRouteType.192.94.93.1}
GetResponse {ipRouteDest.192.94.93.2 = 192.94.93.2,
             ipRouteType.192.94.93.2 = indirect}

GetNextRequest {ipRouteDest.192.94.93.2,
                ipRouteType.192.94.93.2}
GetResponse {ipRouteDest.192.94.93.3 = 192.94.93.3,
             ipRouteType.192.94.93.3 = invalid}

GetNextRequest {ipRouteDest.192.94.93.3,
                ipRouteType.192.94.93.3}
GetResponse {ipRouteDest.192.94.93.4 = 192.94.93.4,
             ipRouteType.192.94.93.4 = other}

GetNextRequest {ipRouteDest.192.94.93.4,
                ipRouteType.192.94.93.4}
GetResponse {ipRouteDest.192.94.93.5 = 192.94.93.5,
             ipRouteType.192.94.93.5 = indirect}

GetNextRequest {ipRouteDest.192.94.93.5,
                ipRouteType.192.94.93.5}
GetResponse {ipRouteIfIndex = 5,
             ipRouteProto = 1}
```

c) ipNetToMediaEntry: The object class definition for this object class does not contain the attribute specified in the filter (ipRouteType), therefore the ISO/Internet proxy will conclude that there is no

object instances with ipNetToMediaEntry object class in the scope.

There are a set of five object instances in the scope. After the filter is applied to these five object instances, there are only two object instances left on which the get operation is performed.

3. Form the response

The following CMIP responses should be formed and sent back to the ISO/CCITT manager

```
CMIP get link reply:
     Object class: ipRouteEntry
     Object instance:{cmipsxmpProxyId = "NetLabs" }
          {internetClassId = ip }
          {internetClassId = ipRouteTable }
          {internetClassId = ipRouteEntry.192.94.93.2 }
     Attribute list: ipRouteDest = 192.4.93.2

CMIP get link reply:
     Object class: ipRouteEntry
     Object instance:{cmipsxmpProxyId = "NetLabs" }
          {internetClassId = ip }
          {internetClassId = ipRouteTable }
          {internetClassId = ipRouteEntry.192.94.93.5 }
     Attribute list: ipRouteDest = 192.4.93.5

Final Empty CMIS M-GET response
```