ISO/CCITT and Internet Management Coexistence (IIMC):

ISO/CCITT to Internet Management Proxy

(IIMCPROXY)

Draft 2
May 26, 1993

April Chang (Editor)

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

Status of this Memo

This document provides information to the network and systems management community. This document is intended as a contribution to ongoing work in the area of multi-protocol management coexistence and interworking. This document is part of a package; see also [IIMCOMIBTRANS] [IIMCIMIBTRANS] [IIMCMIB-II] and [IIMCSEC]. Distribution of this document is unlimited. Comments should be sent to the Network Management Forum IIMC working group (iimc@thumper.bellcore.com).

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.

Abstract

This document is intended to facilitate the use of the ISO/CCITT 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 document describes an ISO/CCITT to 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 [IIMCIMIBTRANS].

Table of Contents

Status of this Memo i
Abstract ii
Table of Contents ii
Revision History iv
1 Introduction 1
1.1 Problem Statement 1
1.2 Overview of IIMC 1
1.3 MIB Translation Procedures 2
1.4 Native Management Model 3
1.5 Proxy Management Model 4
1.6 Scope of this Document 5
1.7 Terms and Conventions 8
2 ISO/Internet Proxy Configuration 10
2.1 ISO/Internet Proxy Containment Tree 10
2.2 System Objects 11
2.3 Translated MIB Schema Information 11
2.4 IIMC Party MIB Objects 13
2.5 IIMC Proxy MIB Objects 13
2.6 OMNIPoint 1 Capability Object 14
2.7 MIB Usage 15
2.8 Retained Information 16
3 Elements of CMIS Service Emulation 16
3.1 Association Service 16
3.2 Object Selection - Scoping and Filtering 17
3.3 Management Operation Services 18
3.4 Synchronization 21
3.5 M-GET Service 22
3.6 M-CANCEL-GET Service 22
3.7 M-SET Service 23
3.8 M-ACTION Service 24
3.9 M-CREATE Service 24
3.10 M-DELETE Service 26

3.11 Management Notification Services 27
4 Common Procedures For CMISE Service Emulation 28
4.1 Verifying Existence Of An Object Instance 28
4.2 Translating Timestamps 28
4.3 Derivation of SNMP Request Parameters 29
4.4 Derivation Of CMIS Parameters 30
5 Error Message Translation 35
5.1 Translating SNMP Error Messages 35
5.2 CMIS Processing Failure 38
6 ISO/CCITT Systems Management Functions 39
6.1 Event Report Management Function 39
6.2 Log Control Function 40
6.3 Scope of the EFD and Log 40
7 ISO/CCITT-Internet Proxy MIB 41
7.1 Proxy MIB GDMO Templates 41
7.2 Proxy MIB ASN.1 Module 46
8 Conformance Requirements 48
8.1 Management Communication Requirements 48
8.2 Management Function Requirements 48
8.3 Management Information Requirements 48
8.4 Service Emulation Requirements 49
9 Acknowledgments 50
References 52
Appendix A (Normative)
Managed Object Conformance Statements (MOCS) 55
Appendix B (Informative)
Example Operation 56

Revision History

Draft 0 - October 9, 1992
Initial draft of this document.

Draft 1 - March 28, 1993
Previous draft of this document (replaced Draft 0).

Draft 2 - May 26, 1993
Current draft of this document (replaces Draft 1).

Major Changes Since Last Revision

1. Revised Naming to have multiple system instances, but without constraining what system is bound to.
2. Modified the cmipsnmpProxyAgent object definition.
3. Deleted all system management function subsections except those for Event Report Management and Log Control. Defined the behaviour of EFD and Log.
4. Reorganized Proxy MIB GDMO and ASN.1 module sections.
5. Modified System Management Function requirements to include only optional support of AOM221 and AOM231.
6. Added requirements for mandatory support of ISO DMI System, translated MIB2 internetSystem, and optional support of the OMNIPoint 1 capabilityObject.
7. Modified minimum CMIP requirement to be AOM12, but minimum CMISE User requirement to be Kernel + Scoped Get.
8. Added security defaults as a new object contained by the cmipsnmpProxyAgent object class.

Outstanding Issues

1. Need to specify the semantics of the operational and usage state attributes in "remote system" objects.
2. No "stateful" optimizations have been proposed.
3. Refer to [IIMCIMIBTRANS] for outstanding issue regarding translation of conceptual tables; that proposal would also impact this IIMCPROXY specification.

1 Introduction

This section provides an overview of ISO/CCITT and Internet Management Coexistence (IIMC) activities, insight into the problem being addressed by IIMC, and a brief introduction to the strategy adopted by IIMC: use of translated MIBs in either a proxy or native implementation. The section concludes by describing the scope of this document, and terms and conventions used by this document.

1.1 Problem Statement

The need for enterprise network management has been addressed by development of network management standards within various communities, most notably the ISO/CCITT and Internet communities.

- The ISO/CCITT community developed the Common Management Information Protocol (CMIP) [ISO9596-1], and related SMI documents [ISO10165-1,2,4].

- The Internet community developed the Simple Network Management Protocol (SNMP) [RFC1157], and its successor, SNMPv2 [RFC1448]. The Internet SMI is defined in [RFC1155] and [RFC1442].

These standards share a nearly common management model, but diverge due to differing management philosophies. Although functionally similar, the Internet and ISO/CCITT protocols and SMIs differ in terms of their complexity and specific operations. Business requirements for end-to-end enterprise management include the need to integrate the management of components accessed by ISO/CCITT management, Internet management, and proprietary management mechanisms in a manner which presents a unified view of the network, despite protocol and SMI differences.

For example, many telecommunications and computer vendors, represented by organizations such as the Network Management Forum (NMF), and the U.S. government, as specified in the Government Network Management Profile (GNMP), have based their enterprise management model on the ISO/CCITT management model. These organizations are particularly interested in integrated management of devices that use the Internet management. This interest is primarily due to the widespread commercial implementation and use of such devices, especially devices that use the Internet TCP/IP protocol suite.

1.2 Overview of IIMC

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

[IIMCIMIBTRANS]	Translation of Internet MIBs to ISO/CCITT GDMO MIBs
[IIMCOMIBTRANS]	Translation of ISO/CCITT GDMO MIBs to Internet MIBs
[IIMCMIB-II]	Translation of Internet MIB-II (RFC1213) to ISO/CCITT GDMO MIB
[IIMCPROXY]	ISO/CCITT to Internet Management Proxy
[IIMCSEC]	ISO/CCITT to Internet Management Security

These documents together comprise a package aimed at integrating ISO/CCITT-based and Internet-based management systems. These documents represent coexistence and interworking efforts underway within the IIMC working group, chartered under the auspices of the Network Management Forum Architecture Integration ISO/Internet (AIII) technical team.

The IIMC intends to address the problem 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" [NMFTR107]. The documents included in the IIMC package are the next level of detailed specifications which implement several of the methodologies identified in the strategy.

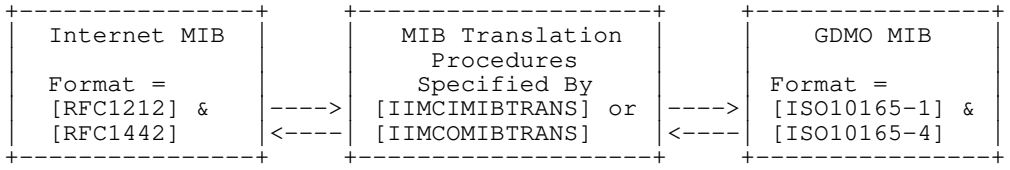
1.3 MIB Translation Procedures

The foundation of IIMC is provided by a pair of Management Information Base (MIB) translation procedures.

- [IIMCIMIBTRANS] specifies translation procedures for converting MIBs from Internet MIB macro format into ISO/CCITT GDMO template format.

- [IIMCOMIBTRANS] specifies translation procedures for converting MIBs from ISO/CCITT GDMO template format into Internet MIB macro format.

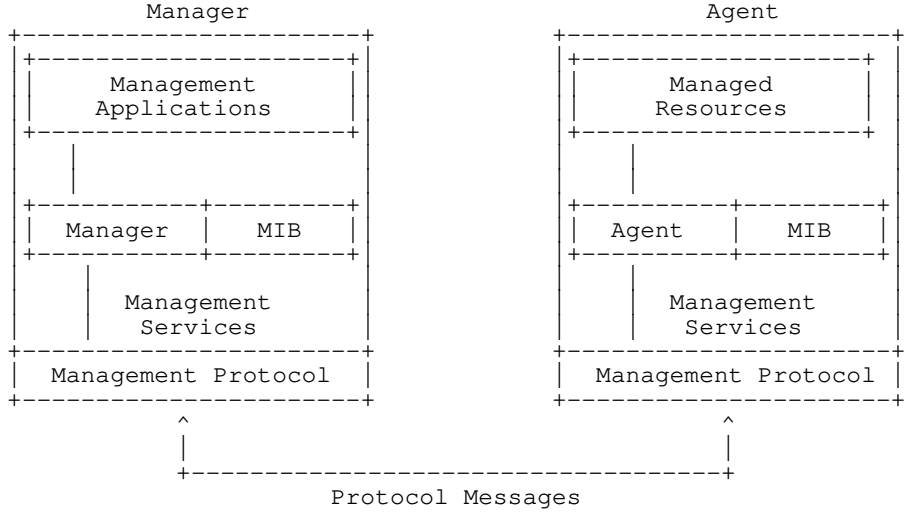
The IIMC approach is to specify direct translation procedures which yield a pair of functionally-equivalent MIBs, as shown in the following figure.



MIBs translated by these procedures may be used to take advantage of existing MIB definitions when business needs require deployment in a different management environment. Translated MIBs may also be used to provide uniformity when multiple management environments are supported by a single system (e.g., dual stack managers). Finally, IIMC MIB translation procedures may be used to support service emulation by a proxy.

1.4 Native Management Model

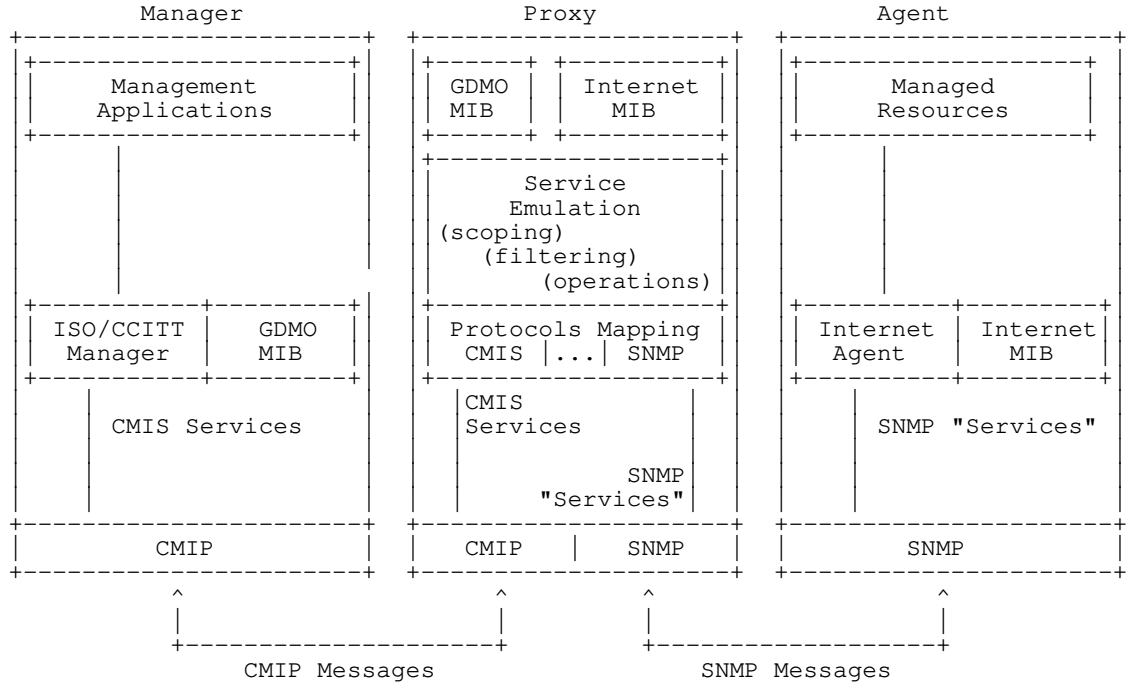
The basic model for ISO/CCITT and Internet management is illustrated in the following diagram.



Within IIMC documents, this model is referred to as the "native" management model. MIBs translated using IIMC procedures can be used by "native" agent implementations. For example, an ISO/CCITT agent can make visible TCP/IP managed resources using the translated GDMO version of the Internet MIB-II [RFC1213] specified by [IIMCMIB-II]. Dual-stack managers or agents may also be implemented which support both the original MIB and the translated MIB generated using IIMC-specified procedures.

1.5 Proxy Management Model

The basic model for ISO/CCITT to Internet proxy management is illustrated in the following diagram. This proxy is specified by [IIMCPROXY]. A similar approach could also be taken to specify an Internet to ISO/CCITT proxy, although no such IIMC document is currently specified.



This ISO/CCITT to Internet proxy 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 [IIMCIMIBTRANS]. The proxy uses these MIB definitions and rules to provide run-time translation of management information carried in service requests and responses.

The proxy 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, SNMPv1 and SNMPv2 are variants of the same protocol mapping process).

1.6 Scope of this Document

The intent of this document (IIMCPROXY) is to facilitate the use of ISO/CCITT CMIP-based managers to perform integrated management of networks via proxy management of networks that are accessed using Internet SNMP-based agents. There are two major differences between CMISE and SNMP services: the structure of management information, and the management operations supported by the underlying protocols. The ISO/Internet Proxy architecture as shown in section 1.2 provides CMISE service emulation. In another words, the ISO/Internet Proxy acts as a CMIP-based agent with respect to the manager, allowing management of Internet objects by the ISO/CCITT manager. CMIS requests are processed by the ISO/Internet proxy and CMIS responses are returned by the ISO/Internet proxy. SNMP traps and Inform requests are converted to CMIS notifications by the ISO/Internet proxy. The implementation of the proxy requires that the Internet MIBs be mapped to ISO/CCITT GDMO definitions.

1.6.1 Approaches to Service Emulation

As described by [NMFTR107], there are different approaches for mapping Internet MIBs and ISO/CCITT MIBs.

- The "direct translation" approach maps each Internet object to a newly defined ISO/CCITT GDMO 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 translation" approach maps Internet objects to different ISO/CCITT GDMO objects. For example, the MIB-II system object is similar to, and could be represented by, the ISO/CCITT system object. The abstract translation approach can also be used to map several Internet objects to a single ISO/CCITT GDMO object which provides only a summary view of the original Internet objects.

Either or both approaches could be used by an ISO/CCITT manager to manage Internet agents. This document uses the "direct translation" approach.

To perform the CMISE service emulation, the ISO/Internet proxy can use either of the approaches described by [NMFTR107] to retrieve or modify Internet MIB information.

- In the "stateless" approach, the proxy does not maintain the Internet agent's MIB data. Instead, for each

received CMIS request, the ISO/Internet proxy generates one or more SNMP requests to the Internet agent in order to achieve the same intent of the CMIS request.

- The "stateful" approach requires the proxy to replicate an Internet agent's MIB locally, and to send periodic (unsolicited) requests to Internet agents to keep the replicated MIB current. The ISO/Internet proxy then tries to fulfill each incoming CMIS request by using locally-replicated MIB data, instead of sending SNMP requests to the Internet agent.

The "stateful" approach will usually provide better response time, but has the drawback that the data retrieved might not be current. In this approach, the poll frequency used to update the locally-replicated MIB has a significant effect on the accuracy of the response.

This document uses the "stateless" approach in which the proxy responds to incoming CMIS requests by generating appropriate SNMP requests. Furthermore, SNMP traps and inform requests are converted to CMIS notifications.

If necessary, the static Internet MIB data retrieved by the ISO/Internet proxy could be cached by the proxy in order to improve the response time of an operation. This document makes no assumption that the proxy caches static information, and so takes no advantage of information which might be cached.

1.6.2 Proxy Inputs and Outputs

This document describes a proxy which emulates CMIS services through generation of appropriate SNMP protocols. The proxy is based on certain inputs and outputs, as shown below in Tables 1 and 2.

CMIS services [ISO9595] are supported by CMIP version 2 protocol [ISO9596-1]. SNMP protocols are as defined for SNMPv1 [RFC1157] and SNMPv2 [RFC1448]. The emulation is slightly different, depending upon whether SNMPv1 or SNMPv2 protocols are being used.

Service	Source
ACSE Associate Indication	CMIP Stack
ACSE Release Indication	CMIP Stack
ACSE Abort Indication	CMIP Stack
CMIS Get Indication	CMIP Stack
CMIS Cancel Get Indication	CMIP Stack
CMIS Set Indication	CMIP Stack
CMIS Create Indication	CMIP Stack
CMIS Delete Indication	CMIP Stack
CMIS Action Indication	CMIP Stack
CMIS Event Report Confirm	CMIP Stack
SNMPv1 Get Response	SNMPv1 Stack
SNMPv1 Trap	SNMPv1 Stack
SNMPv1 Error	SNMPv1 Stack
SNMPv2 Trap	SNMPv2 Stack
SNMPv2 Get Response	SNMPv2 Stack
SNMPv2 Get Bulk Response	SNMPv2 Stack
SNMPv2 Inform Request	SNMPv2 Stack
SNMPv2 Error	SNMPv2 Stack

Table 1 - Proxy Inputs

Service	Target
ACSE Associate Response	CMIP Stack
ACSE Release Response	CMIP Stack
ACSE Abort Request	CMIP Stack
CMIS Get Response	CMIP Stack
CMIS Cancel Get Response	CMIP Stack
CMIS Set Response	CMIP Stack
CMIS Create Response	CMIP Stack
CMIS Delete Response	CMIP Stack
CMIS Action Response	CMIP Stack
CMIS Event Report Indication	CMIP Stack
SNMPv1 Get Request	SNMPv1 Stack
SNMPv1 Set Request	SNMPv1 Stack
SNMPv1 Get Next Request	SNMPv1 Stack
SNMPv2 Get Request	SNMPv2 Stack
SNMPv2 Set Request	SNMPv2 Stack
SNMPv2 Get Next Request	SNMPv2 Stack
SNMPv2 Get Bulk Request	SNMPv2 Stack

Table 2 - Proxy Outputs

This document assumes that CMIP PDUs and SNMP PDUs received by the ISO/Internet proxy are always properly encoded (i.e., the underlying protocol stacks ensure the correctness of the service indications and confirmations that are passed up to the ISO/Internet proxy).

The security architecture, services, protocols, and mechanisms for the ISO/Internet proxy shall be as defined in [IIMCSEC].

1.7 Terms and Conventions

This document assumes that the reader is familiar with ISO/CCITT management and Internet management, and the terminology of each. The term "SNMP" is used throughout this document to indicate either SNMPv1 or SNMPv2, unless a distinction needs to be made.

Other terms and conventions used throughout this document include the following.

1. **ISO/CCITT manager:** An application entity that implements [ISO9596-1] and acting in the manager role.

2. **Internet agent:** An application entity that supports the agent role of one or more of the SNMP protocols, such as [RFC1157] or [RFC1448].

3. **ISO/Internet Proxy:** An application entity that is responsible for emulating CMIS requests by a) generating SNMP requests, b) using SNMP responses to generate CMIS responses, and c) mapping SNMP Traps and InformRequests to CMIS notifications, all between a given (ISO/CCITT manager, Internet agent) pair. A proxy may concurrently support more than one (ISO/CCITT manager and Internet agent) pair.

4. **Known Internet agents:** A set of one or more Internet agents that an ISO/Internet proxy has knowledge of. Each known Internet agent is represented by an instance of the proxy object. This document defines the cmipsnmpProxyAgent object class.

5. **Known SNMP Parties:** A set of one or more SNMP parties that an ISO/Internet proxy has knowledge of. Each known SNMP party is represented by an instance of the partyTable object. The partyTable object class is defined in [IIMCSEC].

6. **Pseudo Object:** A GDMO object class that does not contain any attributes which may be retrieved from an Internet agent (for example, a GDMO object class that represents a group in the Internet MIB-II, or any GDMO object classes representing Internet MIB tables).

7. **Local object (instance):** An object instance that is implemented by the proxy itself (for example, the cmipsnmpProxy and cmipsnmpProxyAgent classes defined in section 7).

8. **Remote object (instance):** An object instance that physically resides within an Internet agent is considered a "remote object" (for example, Internet MIB-II objects like system, tcp, and udp).

9. **Multiple Instance Object:** An object class that may have more than one object instance. For example, Internet MIB table entries.

10. **Delete Information:** The object identifier of the attribute and its attribute value used to indicate that a particular row of a table is deleted.

11. **Proxy System Object:** The [ISO10165-2] DMI system managed object instance that represents the proxy itself.

12. **Remote System Object:** The [ISO10165-2] DMI system managed object instance that represents the remote proxied device on which the Internet agent resides.

Abbreviations

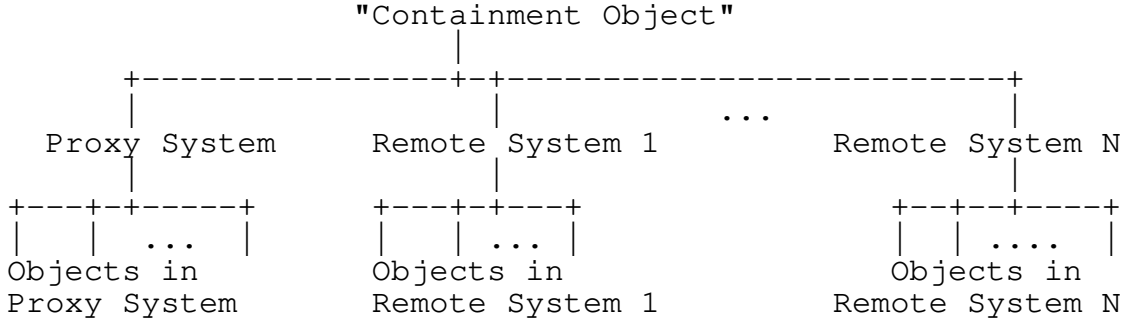
CMIP:	Common Management Information Protocol
CMIS:	Common Management Information Service
CMISE:	Common Management Information Service Element
DN:	Distinguished Name
MIB:	Management Information Base
MOC:	Managed Object Class (CMIP)
MOI:	Managed Object Instance (CMIP)
MIT:	Management Information Tree (naming tree)
OID:	ASN.1 Object Identifier
PDU:	Protocol Data Unit
RDN:	Relative Distinguished Name
SNMP:	Simple Network Management Information Protocol
SNMPv1:	Simple Network Management Information Protocol version 1 [RFC1157]
SNMPv2:	Simple Network Management Information Protocol version 2 [RFC1448]

2 ISO/Internet Proxy Configuration

In order for the ISO/Internet proxy to interwork with the known Internet agents, the proxy needs to know initialization information such as the transport address, network address, protocol version, and security policy for each of the known Internet agents. Such configuration may be done through an off-line process, or through an on-line management exchange not specified by this document.

2.1 ISO/Internet Proxy Containment Tree

The proxy shall support a forest of object instance trees, each rooted at the ISO/CCITT system managed object defined by [ISO10165-2], with one system object instance for each supported Internet agent, and one system object instance for the proxy itself. Each ISO/CCITT system object is distinguished by the value of its systemId or systemTitle attribute, containing the name associated with the Internet agent or proxy application. This ISO/CCITT to Internet Proxy containment tree is shown below.



The "Proxy System" is the [10165-2] "system" managed object instance that represents the proxy itself. The Proxy System contains all objects representing resources within the proxy itself, including the cmipsnmpProxy and cmipsnmpProxyAgent managed objects.

The "Remote System" is the [10165-2] "system" managed object instance that represents the remote proxied device on which the Internet agent resides. The Remote System contains all objects that are implemented by the remote Internet agent. These would include [IIMCMIB-II] "internetSystem", and any other object supported by the Internet agent.

All objects in the above containment tree can be accessed by scoping from the "Containment Object". This may be an instance of any managed object class. One possible "Containment Object" is the so-called global root (i.e., "CCITT Rec. X.660 | ISO/IEC 9834-1 : 1992":root). In this case, scoped requests

including all Remote Systems can be achieved by using a management operation with the base object class equal to "actualClass" and the base object instance name equal to "{}". The proxy may support other alternative "Containment Objects"; this choice is left to the implementor.

2.2 System Objects

The "Proxy System" object particular to the proxy shall be created automatically by the proxy during its local initialization.

The "Remote System" object particular to an Internet agent system shall be automatically created/deleted by the proxy when an associated cmipsnmpProxyAgent object instance in the Proxy MIB is created/deleted. The Distinguished Name of this system object shall have the same value as the cmipsnmpProxyId attribute in the associated cmipsnmpProxyAgent object instance. Creation/deletion of the system object via management operation is not allowed.

The operationalState attribute of the "Remote System" object indicates the perceived state of the Internet agent. It is the same as the operationalState attribute defined in [ISO10165-2]. That is:

- The "enabled" state means that the Internet agent is operational, as perceived by the proxy, i.e., it can be reached.

- The "disabled" state means that the Internet agent is not operational, as perceived by the proxy, i.e., it cannot be reached.

The validity of the operationalState attribute is dependent on the mechanisms used by the proxy to determine reachability, and the frequency with which it is invoked. For connectionless environments (e.g., UDP), polling will have to be performed by the proxy. For connection oriented environments (e.g., TCP), loss of connectivity as indicated by lack of "keep alive" messages can be used to provide this information.

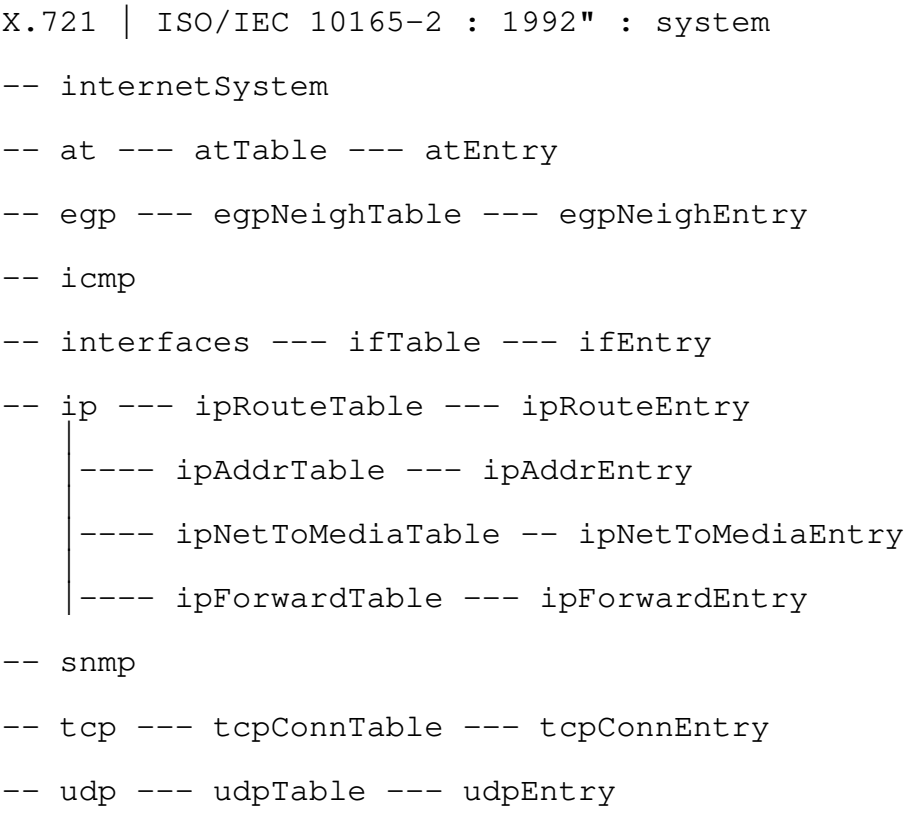
Editor's Note: [The intended semantic of these state attributes requires further consideration.]

2.3 Translated MIB Schema Information

To perform CMISE service emulation, the ISO/Internet proxy requires the Internet MIB's schema information, described in ISO/CCITT GDMO templates. These templates shall be derived from the original Internet MIB according to the procedures defined by [IIMCIMIBTRANS].

The proxy run-time translation of parameters and protocol translation procedures defined in this document depend on the MIB translation, naming and registration procedures defined in [IIMCIMIBTRANS]. The translation and registration procedures defined in that document are structured such that the maximum amount of information is preserved to facilitate the translation process.

An example containment tree for an agent supporting the ISO/CCITT GDMO Internet MIB-II [IIMCMIB-II] (derived from the Internet MIB-II [RFC1213]) is illustrated below. A proxy would have multiple instances of such a tree for each Internet agent supported. (The actual structure of the each containment tree depends upon the MIB(s) supported by the proxy.)



As specified in [IIMCIMIBTRANS], name bindings for ISO/CCITT GDMO object classes derived from Internet MIB table and entry types can be automatically inferred from the Internet registration hierarchy. Thus, object classes derived from Internet conceptual table objects are bound to the object class derived from the group with which the table is associated. Object classes derived from Internet conceptual table entries are bound to the table object classes with which the tables entries are associated. Also, object classes derived from Internet groups are bound to the ISO/CCITT system object class.

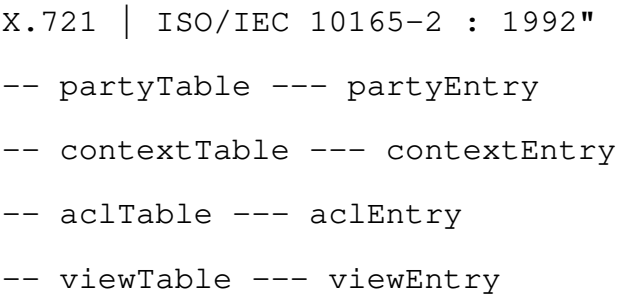
2.4 IIMC Party MIB Objects

Information regarding security policy when accessing agents is contained in Party MIB objects. Binding the Party MIB objects as subordinates of the system object which represents an individual Internet agent allows security policy to be applied on a per Internet agent basis. The Party MIB information can be used by the proxy in a manager role when security services enforcing security policy are implemented in the Internet agent. The services enforced may be authentication, access control, confidentiality and integrity as defined in [RFC1446].

In those situations where the agents may not implement the access control security service on requests from the ISO/CCITT manager (e.g., SNMPv1 agents), the proxy may enforce those services on behalf of the Internet agent. The policy regarding where access control is to be applied is controlled by variables in the cmipsnmpProxy and cmipsnmpProxyAgent managed objects defined in section 7.

The policy regarding security services other than access control, must always be enforced by the Internet agent.

A containment tree diagram for IIMC Party MIB managed object classes is illustrated below. The IIMC Party MIB is subordinate to the ISO/CCITT system managed object that represents the Internet agent.



2.5 IIMC Proxy MIB Objects

The IIMC Proxy MIB defines a set of objects for specifying the information that is needed for both community-based and party-based SNMP management on a per Internet agent basis.

The Proxy MIB consists of a cmipsnmpProxy managed object which contains cmipsnmpProxyAgent object instances, one for each agent being managed by the proxy. The cmipsnmpProxy object class is an immediate subordinate of the "Proxy System" object class. The cmipsnmpProxy object also contains an snmpSecurityParameter object instance which contains default security information.

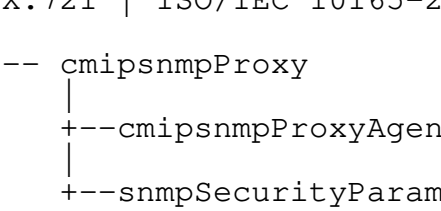
Each cmipsnmpProxyAgent object has information which identifies an Internet agent and how the agent may be reached. Its naming attribute, which contains the administratively-assigned name of the managed device where the Internet agent is located, is used in the naming tree to identify the SNMP managed device.

Creation of a cmipsnmpProxyAgent object instance to represent an Internet agent shall result in the instantiation of a corresponding "Remote System" object representing the Internet agent. The naming attribute value of the "Remote System" shall be the same as the name of the corresponding cmipsnmpProxyAgent object instance. It is recommended that a "OP1 Library Vol. 4":capabilityObject be created for the proxy also.

The cmipsnmpProxyAgent object may be created by management operation, or automatically. For example, the proxy may support discovery of Internet agents, whereby the discovered Internet agents, associated system object, and capability object shall be created automatically by the proxy itself.

This document refers to instances of IIMC Proxy MIB object classes as "local objects" or as "local object instances".

A containment tree diagram for ISO/CCITT proxy MIB managed object classes is illustrated below.



IIMC Proxy MIB GDMO definitions are described in section 7.

2.6 OMNIPoint 1 Capability Object

If used, the "OP1 Library Vol.4":capability object particular to an Internet agent system shall be automatically created and deleted by the proxy when the associated cmipsnmpProxyAgent object in the Proxy MIB is created and deleted.

The following text describes one possible implementation of gathering information defined in the Capability object's supportedObjectClassList. When the cmipsnmpProxyAgent is created, or when the supportedObjectClassList attribute changes, the proxy shall find out all the object classes defined in all the GDMO documents described in the supportedMIBs attribute. The proxy then forms an SNMP Get Next Request with all the object classes (translated to the OID used by the Internet agent) in a variable binding list to

find out whether a particular object class is supported by the Internet agent. The proxy then fills the supportedNameBindingList by finding out all the name bindings used by the object classes in the supportedObjectClassList.

2.7 MIB Usage

The information described in sections 2.1 through 2.6 is maintained by the proxy and used to perform run-time translation between corresponding CMIS and SNMP parameters.

The following definitions are extracted from [IIMCIMIBTRANS] clause 2.3.1, where (c) and (a) refer to class and attribute, respectively.

From [IIMCIMIBTRANS] clause 2.1:

```
{classOID}      ::= {iimcAutoTrans
                     <internetEntityId>(c)}

and

{attributeOID}  ::= {iimcAutoTrans
                     <internetEntityId>(a)}
```

From [IIMCIMIBTRANS] clause 2.2, the ISO/CCITT naming attribute value contains the OID formed as:

```
(naming attribute)  ::=  {iimcAutoTrans
                          <internetEntityId>(c)
                          <internet instanceId>}
```

where the "()" indicates "contents of".

The <internet instanceId> (the OID created uniquely for each Internet object instance) is "0" for object classes that may only have a single instance. The <internet instanceId> for object classes that may have multiple instances is an OID fragment derived from the values of the internet objects identified in the INDEX (or AUGMENTS) clause of the Internet Macro from which the object class is derived, as defined in [RFC1155] or [RFC1442].

The Internet uses the following convention to uniquely identify an Internet object instance:

```
{internet object name}::= {<internetEntityId>(a)
                           <internet instanceId>}
```

Refer to [IIMCIMIBTRANS] for additional detail.

2.8 Retained Information

The proxy must retain information obtained from the ISO/CCITT manager during association establishment, and for individual CMIS requests on the association.

For each outstanding CMIS request, the proxy needs to maintain the ISO/CCITT invoke id, object class and object instance. When SNMP responses are received, the proxy shall use the retained information to form the associated parameters in CMIS responses.

For scoped CMIS requests, the proxy shall maintain some state information to keep track of the portion of the Internet MIB that is being traversed.

3 Elements of CMIS Service Emulation

The following sections describe the conceptual process for performing CMIS service emulation. In an actual implementation, it should be possible to combine some of the processing. It is highly recommended that the implementors of the ISO/Internet proxy combine the processes where possible to optimize the implementation.

3.1 Association Service

The proxy should provide the association service as defined in section 8.1 of [ISO9596-1]. This service includes association establishment and association release.

In ISO/CCITT systems management, management entities may exchange initialization information during the association establishment phase. Such information is used only by the proxy for its own configuration and is not conveyed to the communicating Internet agents.

This document does not define any application context; however, a proxy may be required to support the following application contexts as defined in the ISO standards and CCITT recommendations:

ISO Systems Management application context; or
CCITT TMN application context one.

CMIP and SMASE functional units may be negotiated between the ISO/CCITT manager and the ISO/Internet proxy. Once a set of functional units is agreed, the proxy will ensure only the agreed services are accepted over the association.

The CMIP protocol used between the ISO/CCITT manager and the ISO/Internet proxy is a connection-oriented protocol which requires an association be maintained throughout the management exchange(s). The protocol between the proxy and the

Internet agent, however, may be a connection-less protocol which does not require the existence of an association. Upon receiving an association request from the ISO/CCITT manager, the proxy needs to determine whether connectivity to the agent is possible so that it may accept or reject the association request accordingly. The mechanism used by a proxy to detect the "reachability" of an Internet agent is implementation-dependent, and is not within the scope of the document.

For example, if the reliability of the association is not essential to its management applications, a proxy may assume its Internet agents are always reachable, and may accept association requests on that basis. In this approach, the proxy would terminate the association only when it detects the Internet agent is not reachable.

3.2 Object Selection - Scoping and Filtering

Managed object selection is used to identify a set of managed object instances in the management information tree (MIT) to which a CMIS request applies. Managed object selection is performed in two phases: scoping; and then, filtering. Scoping is used to select candidate object instances in the MIT to which operations may apply. A filter is then applied to attributes of the previously scoped object instances in order to identify the subset of object instances on which the CMIS operation is to be performed.

If no filter is specified, the CMIS request will be performed for all object instances identified by the scope parameter. If no scope parameter is specified, the default is the base managed object instance only.

There are different ways of performing the scoping operation, depending on the implementation. This document specifies one possible way of providing the managed object selection service.

The proxy has no direct knowledge of current object instances that exist in the Internet agent. Therefore, it must first determine the existence of an object before it knows whether it is within the scope. Obviously, all objects in the scope must be instances of object classes that are within the scope. Thus, the proxy should first determine the set of object classes within the scope, and then discover what instances of those object classes actually exist in the Internet agent. This set of object classes that are within the scope are called the "object class group" (OCG).

The following pseudo code algorithm specifies a generic method of determining the members of an "object class group".

Define the set of object classes at the current level in the naming tree that are being processed as the "class level group" (CLG).

Define the set of object classes at the next level in the naming tree that are to being processed after the CLG as the "next level group" (NLG).

Define the managed object class named by the incoming request as the "base managed object" (BMO).

Minimum Level and Maximum Level are derived from the CMIS scope parameter.

```
CLG = {BMO};
OCG = {}; /* empty set */
currentScope  = 0;

WHILE ( currentScope <= Maximum Level )
     NLG = {children of objects in CLG};
     IF (currentScope > Minimum Level)
               THEN OCG = {union of OCG and CLG};
     CLG = NLG;
     currentScope = currentScope + 1;
     ENDWHILE;
```

The determination of the set NLG as {children of objects in CLG} may be done using implementation-dependent internal data structures of the proxy.

An alternative method to determine NLG is to use the name binding templates directly. The following algorithm could then be used to determine the NLG.

```
WHILE (CLG not equal to {})
     Remove an object from CLG;
     FOR (all name bindings in this proxy's associated
          Capability object)
          IF object == SUPERIOR
               THEN NLG = {union of the
                         SUBORDINATE object and NLG};
               ENDFOR;
     ENDWHILE;
```

3.3 Management Operation Services

If the specified instances (i.e., those selected by scoping process) are "local objects", the proxy performs the services using local means.

If the specified instances are "remote objects", then the following steps apply. Any objects that physically reside in the Internet agents are considered "remote objects". For

example, Internet MIB II objects like system, tcp, and udp are considered "remote objects".

1) Determine if the attributes specified by the filter expression (if any) belong to the object class. If not, then remove the object class from the object class group.

2) If the object is a pseudo object (table object), then there is only one possible instance and the values of the attributes are known locally to the proxy. The pseudo object exists if and only if there exists a non-pseudo subordinate object (table entries). The proxy shall attempt to determine if there exists a non-pseudo subordinate object by issuing an SNMP GetNext Request using as an argument the Internet object name for the pseudo object, <internetEntityId>(c). If the SNMP response contains an Internet object that translates to an attribute of a child of the pseudo object, then the pseudo object exists. If the pseudo object does not exist, then remove it from the object class group.

3) If the object is not a pseudo object, then determine if an instance of the object exists by attempting to retrieve, for the first instance, all of the attributes specified in the CMIS filter and the attributeId list or attribute list.

   i) For single instance objects, use an SNMP Get Request or GetNext Request, with the parameters translated according to section 4.

   ii) For multiple instance objects, use an SNMP GetNext Request, with the parameters translated as shown in section 4. This will result in the retrieval of the first instance of the translated Internet objects (i.e., a table entry). If the resulting table entry has been deleted, or is otherwise unavailable for retrieval, then go to step 5.

4) Apply the filter to the attributes of the object instance identified in steps 2 and 3.

   i) If the filter evaluates to FALSE, then perform no further processing on the object. However, for multiple instance objects, save the results of step 3 part (ii).

   ii) If the filter evaluates to TRUE, then attempt to perform the operation on the object, as follows.

      - If the CMIS operation was M-GET, then the M-GET has been completed from the Internet agent. Formulate the appropriate CMIS M-GET response and send it to the manager.

- If the CMIS operation was M-SET, then perform the corresponding SNMP Set Request on the Internet objects. When the SNMP Response returns, formulate the appropriate CMIS M-SET response and send it to the manager.

- If the CMIS operation was M-CREATE, then perform the create on the Internet objects (conceptual row elements) using the algorithm appropriate to the object. When the create process is finished, formulate the appropriate CMIS M-CREATE response and send it to the manager.

- If the CMIS operation was M-DELETE, then perform the delete on the Internet objects. When the SNMP Response returns, formulate the appropriate CMIS M-DELETE response and send it to the manager.

5) For multiple instance objects, use an SNMP GetNext Request, with the parameters translated according to section 4, and with the <internet instanceId> set to the values retrieved for the previous instances for all Internet object names. This will result in the retrieval of the next instance of the translated Internet objects. If the Internet object instances are not of the same type as those requested, then all instances of the multiple instance object class have been processed; go to step 6. If the Internet object instances are of the same type as those requested then retain the results of the GetNext Request for the next iteration and repeat steps 4 and 5.

   In order to reduce the amount of traffic generated by multiple Get-Next requests, the proxy could use the SNMPv2 Get-Bulk request with the non-repeaters parameter set to zero and the max-repetitions set to some arbitrary value (either by guessing or based on previous knowledge). Setting non-repeaters to zero makes the Get-Bulk operation behave like Get-Next.

6) Attempt to select another object class from the object class group. If one exists then go to step 1; otherwise, return an appropriate final CMIP PDU (e.g., empty M-GET or M-SET response) and quit processing the request.

3.4 Synchronization

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

If SNMPv1 is used, the types of atomic requests that the ISO/Internet proxy can perform are as follows.

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

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

In SNMPv1, either a Get or Set operation would fail if the operation on any selected variable(s) failed. This remains true for SNMPv2 Set operations, but not for SNMPv2 Get operations. Using SNMPv2, if you request 4 variables in a Get request, and there is an error in one of the variables, you get a response with 3 valid values and one error. The proxy shall fail a CMIS "atomic" request if the corresponding SNMPv2 Get operation is only partially successful.

For a "best effort" request, the ISO/Internet proxy should try to perform the request on all the instances specified by the request. Since the SNMP protocol supports only "atomic" operations, an operation (especially an SNMP Set Request operation) on multiple variables may be rejected if the operation on any one of the selected variables failed. Upon receiving such an error, the proxy should retry the request by sending multiple requests with each request containing only a single variable. In the time window in which these SNMP requests are being processed, another SNMP Set Request could be issued which could modify the value of a selected variable. For this reason, the complete integrity of a CMIS scoped request cannot be guaranteed. A proxy which complies with this document is not required to detect or avoid this situation, and will not usually report any error if this situation occurs.

3.5 M-GET Service

The following sub-sections describe how the M-GET service may be emulated. Upon receiving a CMIS M-GET request, the proxy first verifies the existence of the based managed object. The procedures for verifying the existence of a managed object is described in section 4.1.

3.5.1 Form The Request

If the CMIS request's attributeIdList parameter is empty (selects all attributes), the proxy shall query the schema

information to find out what attributes are specified for the requested object class.

If the CMIS request's attribute specifies a non-null attributeIdList, the proxy shall verify that the identified attribute(s) are specified for the object class. If the identified attribute is not specified in the object class, the proxy shall return a noSuchAttribute CMIS error without sending SNMP requests to the Internet agent.

For all attributes that are specified in the object, an SNMP Get or SNMP GetNext Request shall be formed, based on the mapping specified in section 4. Use of SNMP Get or GetNext is an implementation issue; however, SNMP GetNext is recommended for performance reasons. Since some non-conforming agents may not implement all the object types in an object group, SNMPv1 Get would return a noSuchName error in this case, and the proxy will need to remove the non-implemented variable binding and resend the SNMP Request. If SNMP GetNext is used instead, the proxy would either discard the non-implemented attribute or translate the SNMP Response to appropriate CMIS getListError.

3.5.2 Form The Response(s)

The proxy shall form the CMIS response according to the mappings specified in section 4.

If the CMIS request's attributeIdList is null (selects all attributes), the proxy shall never return the CMIS getListError. If the Internet agent does not implement all the variables in an object (which violates conformance to the SNMP specification), the proxy shall form the CMIS M-GET response with all the attributes implemented by that Internet agent.

If the CMIS request's attributeIdList selects all attributes, the proxy shall supply in all the attributes that are inherited from the ISO/CCITT Top object in the CMIS response.

3.6 M-CANCEL-GET Service

The M-CANCEL-GET operation shall be performed as described in [ISO9596-1]. The ISO/Internet proxy does not need to generate any SNMP Requests in order to emulate the CMIS M-CANCEL-GET request. However, upon receiving an M-CANCEL-GET request, the ISO/Internet proxy shall stop sending further CMIS M-GET responses to the ISO/CCITT manager for the canceled M-GET request. Furthermore, the proxy shall not initiate further SNMP Requests to the Internet agent for the canceled M-GET request. If the Internet agent continues to return SNMP Get responses corresponding to the canceled M-GET request, they shall be discarded by the proxy.

Depending on when an M-CANCEL-GET request is received, the proxy may send out different responses for the canceled M-GET request and for the M-CANCEL-GET request.

If the Invoke Id of the M-GET request to be canceled is not recognized by the proxy, the proxy shall return a "no such invoke identifier" CMIS error to the ISO/CCITT manager. This can happen when the proxy has not received such an M-GET request, or when the proxy has completed the identified M-GET request.

An M-GET operation is considered completed if the corresponding M-GET response has been sent. For the single object M-GET case, this means the sending of a single M-GET response. For the scoped multiple object case, this means the sending of the final empty M-GET response for the linked replies.

If the identified M-GET request was received, but has not been completed, the proxy generates an "operation canceled error" to the ISO/CCITT manager as a response to the canceled M-GET request. In this case, the proxy will also acknowledge the successful completion of the M-CANCEL-GET request to the ISO/CCITT manager.

3.7 M-SET Service

The following sections describe how M-SET service may be emulated. Upon receiving a CMIS M-SET request, the proxy verifies the existence of the based managed object, according to the procedures defined in section 4.1.

3.7.1 Perform The Set Operation

For each selected ISO/CCITT object instance, the proxy would generate one or more SNMP Set Requests to modify the attributes identified by the CMIS modificationList parameter, according to the specified 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 CMIS request, the "replace" operator should be assumed.

The CMIS "add value" and "remove value" modify operators are not supported by SNMP protocol, and are not supported by the ISO/Internet proxy. Since SNMP uses default values only for initialization (i.e. at creation time), the "set to default" modify operator is not supported by the ISO/Internet proxy either. If the modify operator value included in an M-SET request is not supported, "invalid operator" should be reported in the CMIS setListError response.

3.7.2 Form The Response(s)

If the M-SET request is a "confirmed" request, the proxy shall construct an M-SET response. The CMIS M-SET response should contain the attribute values list or the appropriate setListError. Once the CMIS M-SET response has been constructed, it is passed to the CMIP service provider, which send the corresponding CMIP PDU to the ISO/CCITT manager.

If the CMIS M-SET request is a scoped request, attribute values of each ISO/CCITT object are reported as a linked reply.

3.8 M-ACTION Service

Since Internet MIBs do not have any actions defined, the translation of CMIS M-ACTION to corresponding SNMP operations is not needed. Any CMIS M-ACTION request which is received pertaining to a translated Internet MIB object will be rejected by the proxy with an "noSuchAction" error response. However, CMIS M-ACTION may be used by the proxy for other purposes.

3.9 M-CREATE Service

3.9.1 Request Validation

The ISO/Internet proxy is responsible for validating that incoming CMIS M-CREATE requests do not violate name binding and object class definitions.

3.9.2 Name Binding

The ISO/Internet proxy must determine if an instance may be created according to the CREATE clause of the NAME BINDING template specified for the object class. If the instance cannot be created, the CMIS error response "classInstanceConflict" is returned.

The ISO/Internet proxy must also determine from the NAME BINDING template if the instance specified in the request maybe created under the superior object instance identified in the M-CREATE request. If the NAME BINDING does not specify the identified containment relationship, an "invalidOperation" CMIS error response should be returned.

3.9.3 Check For Duplication

The proxy must determine if the instance already exists. If it does, a "duplicate managed object instance" CMIS error response should be returned.

3.9.4 With Referenced Object

If a CMIS M-CREATE request includes a reference object, the ISO/Internet proxy should retrieve the referenced object instance from the Internet agent.

The proxy uses an SNMP GetNext Request for retrieval, with the parameters translated according to section 4, and with the <internet instanceId> set to the translated <internet instanceId> of the reference object for all Internet object names.

The proxy checks if the attribute used for SNMP row creation indicates that the row is not available for use (e.g., has been deleted or is in some other not ready condition). This attribute is the CREATEDELETEATT attribute indicated in the scannable portion of table entries translated according to [IIMCIMIBTRANS]. (Note that this attribute is always defined as GET only in the translated GDMO MIB.)

If the reference object instance does not exist, the proxy must send a "No such reference object" CMIS error response to the ISO/CCITT manager.

3.9.5 With Automatic Instance Naming

A CMIS M-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 CMIS M-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 select an instance name based on the behavior of the object class and name binding. In some cases, the relative distinguished name (RDN) is formed using attributes provided in the CMIS M-CREATE request. For example, the RDN for the Internet MIB-II "atEntry" 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 cannot create the object instance and a "processing failure" CMIS error should be returned.

3.9.6 Perform The Create Operation

The CMIS M-CREATE is realized by setting the status column of the corresponding Internet MIB table entry to a valid value with all other columns of the table entry properly initialized. If the combination of the attributes specified in the CMIS M-CREATE request and the attributes obtained from the reference object do not provide a complete set of 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 with the incomplete attribute list succeeds, the ISO/Internet proxy should retrieve all the attributes of the newly-created entry, including the attributes which have values supplied by the Internet agent with using default values. This complete list of attribute values is returned in the CMIS M-CREATE response.

If the actual creation process with this partial attribute list fails, the ISO/Internet proxy sends a "missing attribute value" CMIS error back to the ISO/CCITT manager.

3.9.7 Form The Response

The results from the Internet agent are used by the proxy to construct a CMIS M-CREATE response, which is then returned to the ISO/CCITT manager, using the mappings defined by section 4.

3.10 M-DELETE Service

3.10.1 Perform the Delete Operation

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

3.10.2 Name Binding

Determine from the NAME BINDING template if the instance specified in the request may be deleted. If the name binding does not allow the deletion of the identified object, a CMIS error response is returned.

3.10.3 Perform The Delete Operation

If the object instance identified in the CMIS M-DELETE request exists, the delete operation is performed. In SNMPv1, object deletion is achievable only if there is a columnar object representing the status of each conceptual row. Deleting an object instance is realized by setting the status columnar object to an invalid value. The value representing "invalid" is implementation-specific. The proxy therefore needs to be aware of the "invalid" value and the status columnar object in order to perform the deletion. For SNMPv2, the object deletion can be achieved by sending an SNMP Set Request to the Internet agent to change the Row Status value to "destroy."

3.10.4 Form The Response(s)

This process includes formatting the CMIP M-DELETE response with the appropriate attribute list or deleteListError

parameters. Once the CMIS M-DELETE response has been constructed, it is returned to the ISO/CCITT manager.

3.11 Management Notification Services

Although SNMPv1 and SNMPv2 use different PDU structures to convey Traps, all SNMP Traps are mapped to the IIMC-defined internetAlarm notification and sent as CMIS M-EVENT-REPORTs. Since SNMP Traps are not confirmed, the translated CMIS M-EVENT-REPORTs are sent as "unconfirmed" event reports.

If the SNMPv2 manager-to-manager communication is supported between an Internet manager and an ISO/CCITT manager, it is possible for the proxy to receive an InformRequest from the Internet system. Like Traps, InformRequests are also mapped to CMIS M-EVENT-REPORTs. Unlike Traps, the internetAlarm notifications resulting from InformRequests are sent as "confirmed" event reports.

If the translation of Traps to notifications fails, no CMIS M-EVENT-REPORT will be generated and the SNMP Traps are simply discarded.

The proxy shall expect a CMIS M-EVENT-REPORT response for all internetAlarm notifications sent in confirmed mode. The CMIS M-EVENT-REPORT response shall contain an empty event report argument. Upon receipt of the CMIS M-EVENT-REPORT response, the proxy shall return an SNMP Response PDU to the Internet agent that is in accordance with SNMPv2 protocol rules and contains an error code of "noError".

If the translation of an SNMPv2 InformRequest to a CMIS M-EVENT-REPORT fails, the proxy shall send an SNMP Response to the originator of the SNMP InformRequest with the error code of "genErr".

If the proxy cannot determine the Internet agent that initiated the SNMP Trap, then the CMIS M-EVENT-REPORT shall be sent as if it originated from the cmipsnmpProxy managed object class. This can occur when there are multiple agents associated with the same network address or when the proxy is unable to recognize the network address. Otherwise, the proxy should always be able to determine the originator from the network address in the Trap message and the event will be sent as if it originated from the MIB-II internetSystem under the remote system.

Refer to section 6 for additional information regarding discrimination of notifications.

4 Common Procedures For CMISE Service Emulation

The procedures described in this section are used during CMISE service emulation defined in section 3. These procedures are collected together here for ease of specification, and to facilitate common implementation.

4.1 Verifying Existence Of An Object Instance

Since the proxy does not maintain a replicate copy of the MIB data maintained by the Internet agents, the existence of the a managed object, such as a based managed object specified in an incoming CMIS request, may need to be verified before further processing, such as scoping and filtering.

If the base object specified in the request does not exist in the Internet agent, then the proxy must send a "NoSuchObjectInstance" CMIS error response back to the ISO/CCITT manager.

If the base managed object is a pseudo object, the ISO/Internet proxy tries to determine if there exists a non-pseudo subordinate object. The base object exists if and only if there exists a non-pseudo subordinate object.

4.2 Translating Timestamps

This document does not specify a standard translation for the timestamp value in CMIS responses. However, the following paragraphs describe two potential implementations for this translation: ISO/Internet proxy's local time, and Internet agent's local time with fixed unknown delta.

4.2.1 ISO/Internet Proxy's Local Time

The timestamp value in the CMIS response can be set to the time provided by the ISO/Internet proxy's internal clock when the final SNMP Response is received to complete processing of a given CMIS request.

4.2.2 Internet Agent's Local Time

The ISO/Internet proxy can query the Internet agent for "sysUpTime", in addition to the original SNMP variable binding list in the first SNMP Request. Using this method, this value is recorded as the "agent's initial sysUpTime" and the ISO/Internet proxy's local time is recorded as "initial contact time".

Each CMIS request is then translated to one or more SNMP Requests by the ISO/Internet proxy to fulfill the CMIS

request. If the last SNMP Request for the same CMIS request is an SNMP Get Request, the "sysUpTime" is 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 CMIS response is then calculated as follows: 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 indication of when the operation was actually performed by the Internet agent (rather than the time that the response processed by the ISO/Internet proxy).

However, in order to convert the sysUpTime, which is the time ticks since the system was last re-initialized, to the CMIS event time, the proxy must be made aware of every reinitialization of the Internet agents. Although each Internet agent generates a Trap when it first comes up, there is no guarantee that the proxy will receive this Trap. Therefore, this method may also be inaccurate.

4.3 Derivation of SNMP Request Parameters

4.3.1 SNMPv2 Party and Context Parameters

The SNMPv2 source/destination party and context parameters shall be derived from the values in the privileged attribute certificate (PAC) passed in the access control parameter of the incoming ACSE or CMIS request.

If no incoming access control parameter is received, the proxy shall use the default context and parties in the snmpSecurityParameter object instance contained by the cmipsnmpProxyAgent. If no default applies, the operation shall be rejected by the proxy with an access denied error.

4.3.2 SNMPv1 Community String Parameter

The SNMPv1 community string parameter shall be derived from the value in the privileged attribute certificate(PAC) passed in the access control parameter of the ACSE or CMIS request.

If no incoming access control parameter is received, the proxy shall use the default community string in the snmpSecurityParameter object instance contained by the cmipsnmpProxyAgent. If no default applies, the operation shall be rejected by the proxy with an access denied error.

4.3.3 Internet Agent Transport Address

For SNMPv2, the proxy uses the value of the destination party identifier, derived according to the procedures in 4.3.1, to look up the transport address in an entry of the partyTable.

For SNMPv1, the Internet agent transport address shall be derived from the associated transport address in the table of cmipsnmpProxyAgent entries. The cmipsnmpProxyAgent is the one which has the same systemId as the attribute value within the RDN of the system object provided in the AETitle (if local name is used), or the CMIS managed object instance parameter.

4.3.4 SNMP Variable-Bindings Parameter

The SNMP variable-bindings list parameter contains a sequence of varBinds, each of which is an (Internet object name, value) pair.

For CMIS M-CREATE, M-SET, M-DELETE requests, the Internet object name shall be derived from the DN contained in the CMIP managed object instance parameter, and the attribute identifier provided in the CMIS request attributeIdList or attributeList parameter, using the algorithm defined in [IIMCIMIBTRANS] clause 2.3.1.

For M-CREATE and M-SET requests, the Internet object value shall be assigned the attribute value associated with the attributeId from which the Internet object name was derived.

For M-GET requests, it is recommended the Internet object value is NULL.

For M-DELETE requests, the proxy shall use the delete information as described in the NAME BINDING template behavior defined for the object class. Within the BEHAVIOUR text, the CREATEDELETEATT specifies the Internet object name and CREATEDELETEVALUE specifies the Internet object value which signifies row deletion.

4.4 Derivation Of CMIS Parameters

Given the rules specified in this section, and knowledge of the IIMC containment hierarchy (name bindings), the ISO/CCITT {classOID}, {attributeOID}, and distinguished name can be derived from Internet names and the agent identifier.

The iimcAutoTrans OID is known to the proxy. It is defined in [IIMCIMIBTRANS].

4.4.1 Attribute Id Parameter

Using knowledge of the Internet name structure, and knowledge of valid <internetEntityId>(a) values known to the proxy, the

<internetEntityId>(a) and <internet instanceId> may be extracted from the Internet object name.

The extraction process is not possible if the valid <internetEntityId>(a) value is not known to the proxy. In that case ,the translation process cannot be performed.

The ISO/CCITT attribute identifier is formed as:

{iimcAutoTrans <internetEntityId>(a)}

4.4.2 Managed Object Class Parameter

In CMIS response, the managed object class parameter can be derived from the proxy's retained information.

If actual class is used in the incoming CMIS request, the proxy must derive the object class parameter from the DN in the original CMIS request. The proxy shall compare the attribute value of the last RDN in the CMIS request with all the known ISO/CCITT object classes. The proxy shall assume that object class that has the longest match with the attribute value of the last RDN is the actual object class. If the CMIS request is a scoped request, the object class shall be derived from the retained information. If the Distinguished name is an empty set of RDN, "CCITT Rec. X.660 | ISO/IEC 9834-1 : 1992": root is assumed as the object class.

4.4.3 Managed Object Instance Parameter

The managed object instance value (the base managed object's DN) is retained by the proxy during processing of the CMIS request. However, for DNs other than the base managed object instance, the following steps shall be taken to derive the subordinate RDNs.

i) The value of the internetClassId naming attribute associated with the object class, may be formed as:

{iimcAutoTrans <internetEntityId>(c) <internet instanceId>}

ii) The internetClassId value, and the internetClass OID are used to form the final RDN for the object's DN. Assume that the object class was able to be determined using the procedures of 4.4.2. The sequence of other RDNs for the DN may be determined as follows.

Use knowledge of the containment hierarchy defined by name bindings, and the Internet agent's identifier. The object class's name binding may be identified as that name binding which contains the object class OID as its final component, in accordance with the name binding registration procedures defined in [IIMCIMIBTRANS] clause 2.1.3. Use the superior/subordinate relationships in the

name bindings to build the DN in reverse, beginning with the final RDN and ending with the RDN for the ISO/CCITT system object. For superior classes that can have only a single instance, the internetClassId value for the object is created by appending the integer zero to the class OID. The agent's identifier is used as the value for the RDN of the ISO/CCITT system object.

One case exists for MIBs derived according to [IIMCIMIBTRANS] where it is possible for the superior object class to have multiple instances. This may occur when the subordinate object class was translated as the result of an SNMPv2 AUGMENTS clause and the superior object class is a table entry. In that case, the instance of the superior object class is identified by the same instanceId used to identify the subordinate object, prepended with the superior object's class OID.

If the Internet agent's address cannot be determined, then it may not be possible to associate a notification with a specific agent. This may be a problem if multiple Internet agents are associated with the same network address. In such cases, the DN for the cmipsnmpProxy object instance shall be used as the object instance.

4.4.4 EventId Parameter

The eventId parameter shall be the OID assigned the internetAlarm as defined in [IIMCIMIBTRANS].

4.4.5 InternetAlarm ProbableCause Parameter

The internetAlarm notification probableCause parameter shall be derived as defined in [IIMCIMIBTRANS] clause 3.2.5.

Internet traps/notifications are registered using the OID corresponding to the value of the Internet snmpTrapOID object defined in [RFC1450].

For SNMPv1 Trap PDUs, the snmpTrapOID is derived as stated in the SNMPv1/SNMPv2 Coexistence document [RFC1452] clause 4.1.2 (2). That definition is repeated below:

"... if the value of the generic-trap field is 'enterpriseSpecific' then the value used is the concatenation of the enterprise field from the trap PDU with additional sub-identifiers, '0', and the value of the specific-trap field."

For notifications defined according to the SNMPv2 SMI, the probableCause is determined by either the snmpTrapOID.0 or snmpEventID.i, which is contained in the second variable binding of the Trap or InformRequest, respectively.

Only the "globalValue" (i.e., OID form) of the probableCause syntax shall be used.

4.4.6 InternetAlarm AttributeIdentifier List

The following process shall be followed for each variable in the variable-bindings, excluding the first two variable-bindings.

The name portion of the variable binding shall be converted to an ISO/CCITT attributeId using the procedure specified in section 4.4.1. The converted ISO/CCITT attributeId shall be placed in the attributeIdList.

4.4.7 InternetAlarm ObjectInstanceList

The following process shall be followed for each variable in the variable-bindings, excluding the first two variable-bindings.

The name portion of the variable binding shall be converted to an ISO/CCITT object instance name using the procedures specified in section 4.4.3. The converted ISO/CCITT object instance name shall be placed in the object instance list.

If the proxy cannot determine the object instance name (e.g., because the Internet agent's identifier cannot be determined), then the objectInstanceList parameter shall not be included in the internetAlarm.

4.4.8 InternetAlarm InternetTrapInfo Parameter

The following process shall be followed for each variable in the variable-bindings.

The name portion of the variable binding shall be converted to an ISO/CCITT object instance name using the procedures specified in section 4.4.3. The converted ISO/CCITT object instance name shall be placed in the objectInstance field of the internetTrapInfo parameter.

If the Internet agent's identifier cannot be determined, or the <internetEntityId>(a) is unknown to the proxy, then the object instance name cannot be determined and the objectInstance field shall not be included in the internetTrapInfo parameter, but shall be included in the unknownVarBindList parameter.

4.4.9 InternetAlarm UnknownVarBindList Parameter

If the proxy cannot determine the attributeId for a variable binding (i.e., because the <internetEntityId>(a)portion of the Internet object name is not known to the proxy), then that

variable binding shall be included in the unknownVarBindList parameter.

4.4.10 InternetAlarm PerceivedSeverity Parameter

The proxy cannot determine the perceivedSeverity for the translated SNMP Trap without specific knowledge of the Trap. Therefore, the proxy always assumes "indeterminate" for all the CMIS M-EVENT-REPORTs generated as a result of SNMP Traps.

However, a proxy can build in some local knowledge of the SNMP Traps and assign different perceivedSeverity values based on its local knowledge. Such local knowledge is not within the scope of this document.

4.4.11 InternetAlarm TransportDomain Parameter

For SNMPv2 Traps, the transportDomain parameter may be determined by using the one of the party identifier parameters associated with the Trap. The partyEntry object identified by the party identifier contains the partyDomain attribute.

For either SNMPv1, or SNMPv2 Traps, knowledge of the transport protocol used may be provided to the proxy. Alternatively, if the transport address can be determined, the proxy can determine the transport protocol from the format of the address. The proxy may then be able to determine the appropriate transportDomain value from local knowledge of the OIDs registered for different transport domains.

4.4.12 InternetAlarm TransportAddress Parameter

See section 4.3.3 for possible ways to determine the transport address.

4.4.13 InternetAlarm AccessControl Parameter

The access control parameter shall be assigned the community string or party identifiers associated with the SNMP Trap.

5 Error Message Translation

5.1 Translating SNMP Error Messages

SNMP error responses received by the ISO/Internet proxy are translated to CMIS error responses and sent back to the ISO/CCITT manager. The following sections provides a mapping for SNMP error messages to CMIS error responses.

5.1.1 tooBig

If the SNMP error "tooBig" is received, the ISO/Internet proxy should try to break the SNMP Request into smaller requests and resend the requests. If it is not feasible to break the

request to any smaller request, and this error occurs, the CMIS error response "Complexity limitation" should be returned to the ISO/CCITT manager.

5.1.2 noSuchName

If the SNMP error "noSuchName" occurs when an attribute is queried as part of a CMIS 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 managed object, then a "NoSuchObjectInstance" CMIS error response should be returned.

If the object exists and the SNMP "noSuchName" error occurs when attempting to read or modify an attribute, a CMIS "No Such Attribute" error response should be returned to the ISO/CCITT manager.

If the ISO/Internet proxy maintains correct schema information and the Internet agent is a conforming agent, an Internet object's attributes should either all exist or none 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-conforming Internet agent.

5.1.3 badValue

If the SNMP error "badValue" is returned for an SNMP Get Request, then a "processing failure" CMIS error response should be returned to the ISO/CCITT manager. In the ProcessingFailure parameter of the CMIS 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 fulfill a CMIS M-DELETE request, a "processing failure" CMIS 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.

5.1.4 readOnly

The proxy should never receive an SNMP readOnly error from an SNMPv1 agent. If this error is received, a "processing failure" CMIS 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.

For SNMPv2, if the SNMP error "readOnly" occurs when checking for the existence of a base object, a "processingfailure" CMIS error response should be returned to the ISO/CCITT manager. In the ProcessingFailure parameter of the CMIS error response, the errorId should be "snmpReadOnly", and the errorInfo should be the variable binding identified by the error-index. If the error occurs when deleting the object, then the deleteErrorInfo field in the response shall be set to "accessDenied".

5.1.5 genErr

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

There are additional error messages in SNMPv2. Most of the errors are defined for the Set Request. Since a Set Request may be originated when processing a CMIP M-SET request, an M-CREATE request or an M-DELETE request, the proxy must ensure each error code is translated to the one which is most compatible with the original CMIS request. In addition, the proxy must ensure the selected error value is compatible with the use of other parameters such as scoping, filtering, synchronization and multiple linked reply.

5.1.6 noAccess

This error indicates the variable binding's name specifies a variable which is not accessible by an SNMP Set Request. This error should be mapped to the CMIS "accessDenied" error.

5.1.7 wrongType

This error indicates the variable binding's value field of an SNMP Set Request specified a type which is inconsistent with that required for the variable. This error may be mapped to the CMIS "invalidAttributeValue" error.

5.1.8 wrongLength

This error indicates the variable binding's value field of an SNMP Set Request specifies, according to the ASN.1 language, a length which is inconsistent with that required for the variable. If the original CMIS request is M-CREATE or M-SET, the CMIS error "InvalidAttributeValue" shall be returned. If the original CMIS request is M-DELETE, the CMIS "processing failure" error shall be returned.

5.1.9 wrongEncoding

This error is used to indicate the variable binding's value field of an SNMP Set Request contains an ASN.1 encoding which is inconsistent with that field's ASN.1 tag. This error should be mapped to the CMIS "processingFailure" error.

5.1.10 wrongValue

This error indicates the variable binding's value field in an SNMP Set Request specifies a value which could under no circumstances be assigned to the variable. This error should be mapped to the CMIS "invalidAttributeValue" error.

5.1.11 noCreation

This error is generated when an SNMP Set Request variable binding name specified a variable which does not exist and could not ever be created. This error should be mapped to the CMIS "invalidObjectInstance" error.

5.1.12 inconsistentValue

This error indicates that an SNMP Set Request variable binding value field specified a value that could under other circumstances be assigned to the variable, but is presently inconsistent. If the SNMP Set Request was generated as a result of a CMIS M-CREATE or M-SET operation, the error should be mapped to the CMIS "invalidAttributeValue" error.

If the SNMP Set Request was generated as a result of CMIS M-DELETE operation, the error may be mapped to the CMIS "processingfailure" error.

5.1.13 resourceUnavailable

This error indicates that the assignment of a value by an SNMP Set Request requires the allocation of a resource which is presently unavailable. This error may be mapped to the CMIS "resourceLimitation" error.

5.1.14 commitFailed

When performing an SNMP Set Request, the Internet agent must ensure all variable assignments occur atomically. If any of the assignments fail, an SNMP "commitFailed" error is returned. If the original CMIS request is a "best effort" request, the proxy should either retry the failed variable assignments by sending multiple SNMP Set Requests, or return a CMIS setListError with a "processingfailure" error.

5.1.15 undoFailed

When performing an SNMP Set Request, the Internet agent must ensure all variable assignments occur atomically. If any of the assignments fail, the agent should undo all the assignments. An SNMP "undoFailed" error is returned when the agent cannot undo all the assignments. CMIS does not have any error value equivalent to this. The CMIS "processing failure" error must be returned.

5.1.16 authorizationError

This error indicates that an SNMP Request has been discarded because the authorization context used in the request does not allow the PDU type. This error is mapped to the CMIS "accessDenied" error.

5.1.17 notWritable

The "notWritable" error is used to indicate that an SNMP Set Request is trying to modify the value of a variable which is not modifiable, no matter what new value is specified. This error shall be mapped to the CMIS "invalidOperation" error.

5.2 CMIS Processing Failure

There are many error scenarios in which the error cannot be mapped to a specific CMIS error. In this case, the "processing failure" CMIS error response should be reported back to the ISO/CCITT manager. In order to provide the ISO/CCITT manager with 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:

```
snmpTooBig     OBJECT IDENTIFIER ::= { iimcErrors 0 }
snmpBadValue   OBJECT IDENTIFIER ::= { iimcErrors 1 }
snmpReadOnly   OBJECT IDENTIFIER ::= { iimcErrors 2 }
snmpGenErr     OBJECT IDENTIFIER ::= { iimcErrors 3 }
noResponse     OBJECT IDENTIFIER ::= { iimcErrors 4 }
cannotDelete   OBJECT IDENTIFIER ::= { iimcErrors 5 }
notImplemented OBJECT IDENTIFIER ::= { iimcErrors 6 }
wrongLength    OBJECT IDENTIFIER ::= { iimcErrors 7 }
wrongEncoding  OBJECT IDENTIFIER ::= { iimcErrors 8 }
```

where the errInfo parameter depends on the value of errorId:

```
errorId        errInfo
-------        -------
snmpTooBig     VarBindList
snmpBadValue   VarBind
snmpReadOnly   OBJECT IDENTIFIER
cannotDelete   VarBind
notImplemented INTEGER {
               transport(0),
               authenticationProtocol(1),
               privacyProtocol(2)
               }
```

6 ISO/CCITT Systems Management Functions

ISO/CCITT systems management standards include a set of Systems Management Function specifications. An ISO/Internet proxy may choose to support some or all of these systems management functions. This section provides some of the modeling approaches which may be used in supporting ISO/CCITT systems management functions.

6.1 Event Report Management Function

The ISO/CCITT Event Report Management Function [ISO10164-5] defines an Event Forwarding Discriminator (EFD) managed object which allows an ISO/CCITT manager to control the forwarding and processing of potential event reports by an ISO/CCITT agent. The Event Report Management Function maybe supported by an ISO/Internet proxy to allow the ISO/CCITT manager to control where and how Internet Traps and Inform Requests may be forwarded.

Since all Internet Traps and Inform Requests are translated by the proxy and are forwarded to their destinations by the proxy, EFD managed objects are best supported by the proxy as local objects. Upon receiving a CMIS M-CREATE request for an EFD, the proxy creates the EFD object instance according to the specified name binding. Once created, the EFD is used by the proxy to determine which CMIS M-EVENT-REPORTs are to be forwarded to a particular destination during a specified time period.

6.2 Log Control Function

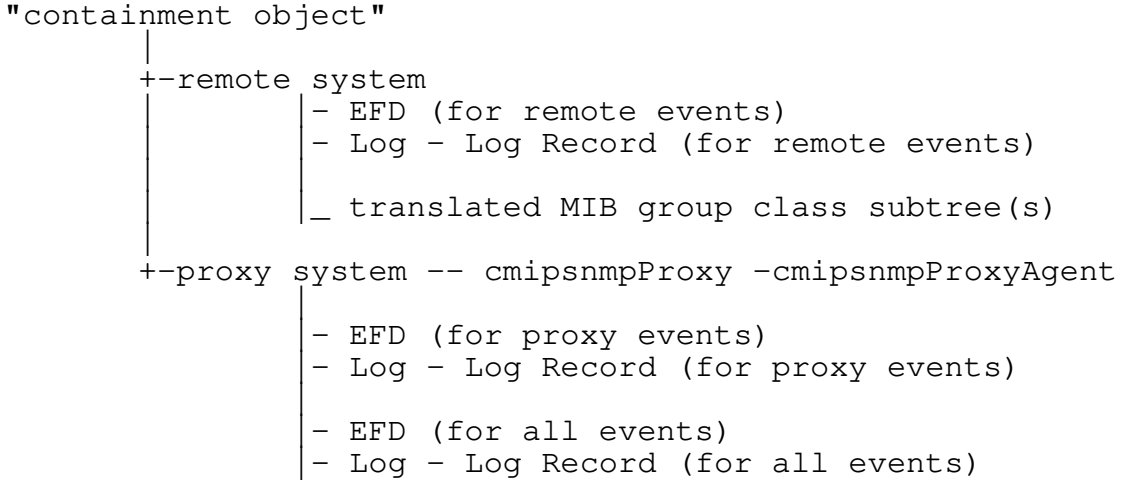
The ISO/CCITT Log Control Function [ISO10164-6] defines a Log managed object which allows control and monitoring of a log and the retrieval of its log records. If the Log managed object is supported, Internet Traps and Inform Requests may be logged according to a predefined criteria.

Since only notifications are logged and these are constructed by the proxy, the Log managed object can be defined as a local object of a proxy. Upon receiving a CMIS M-CREATE request for a Log object, the proxy creates the Log instance according to the specified name binding. Once created, the Log is used by the proxy to process the received Traps and Inform Requests (and local notifications) for logging.

An InternetAlarmRecord managed object class is defined in [IIMCIMIBTRANS].

6.3 Scope of the EFD and Log

EFD and Log objects can be created under either the remote system or the proxy system objects. EFD and Log object instance behaviour is different depending on its position in the containment tree.



EFD and Log objects contained by the remote system object shall process only those events generated by the objects known to each Internet agent (i.e., objects contained by the same remote system object).

EFD and Log objects contained by a proxy system object with the instance name "ProxyOnly" shall process only those events emitted from the object instances contained by the proxy system. Any other EFD or Log object contained by the proxy system shall apply to any event (including all events generated by any object). If an EFD or Log is created under the proxy system using automatic instance naming, the proxy shall choose a name other than the name "ProxyOnly".

7 ISO/CCITT-Internet Proxy MIB

The ISO/CCITT-Internet Proxy MIB defines a set of objects for specifying the information that is needed for both community-

based and party-based SNMP management on a per Internet agent basis. The containment hierarchy and other introductory information regarding this Proxy MIB can be found in section 2.3.

The GDMO templates and ASN.1 modules are included here in one section to facilitate automated processing. Comments and subsection headers are included in the form of ASN.1 comments, i.e., preceded by "--".

This document (IIMCPROXY) is allocated the following registration identifier for purposes of referencing material contained herein.

```
iimcIIMCProxy OBJECT IDENTIFIER ::= {iimcManagementDocMan 3}
```

7.1 Proxy MIB GDMO Templates

-- 7.1.1 Proxy MIB Managed Object Classes

```
cmipsnmpProxyAgent MANAGED OBJECT CLASS
   DERIVED FROM
     "Rec. X.721 | ISO/IEC 10165-2 : 1992":top;
   CHARACTERIZED BY
     cmipsnmpProxyAgentPkg PACKAGE
     BEHAVIOUR cmipsnmpProxyAgentPkgBehaviour BEHAVIOUR
     DEFINED AS
     !This managed object class is used to represent an
     Internet agent in the proxy MIB.  Each agent that
     the proxy manages is represented by an instance of
     this object class.

     The cmipsnmpProxyAgentId attribute contains the
     administratively-assigned name of the managed
     system that contains the Internet agent.  Usually
     this is an Internet Domain Name.  This attribute
     value shall be determined by the manager when the
     object is created.

     The managementProtocol attribute specifies the
     Internet management protocol used by the proxy to
     manage devices.  It shall be the OID indicating
     SNMPv1, SNMPv2, or some other protocol. This
     attribute is assigned a value (an OID) by the
     manager that is appropriate for the Internet agent.

     The supportedMIBs attribute identifies the set of
     GDMO documents that describe the MIBs that the
     Internet agent supports. The ISO/CCITT manager may
     add elements to or remove elements from this
     attribute.
```

```
      Two object instances shall be created by the proxy
      automatically when an instance of the
      cmipsnmpProxyAgent class is created. One is the
      system object that represents the Internet
      agent. The other is the "OP1 Library Vol.
      4":capabilityObject as defined by [NMF006].

      The accessControlEnforcement attribute indicates
      where access control is applied: at the Internet
      agent, the ISO/Internet proxy, or both.

      The accessControlMechanism attribute indicates
      whether no access control, Internet access control
      as specified in [RFC1446], or ISO/CCITT access
      control as specified in [ISO10164-9] is to be used.
      The default is no access control.

      The administrativeState attribute is used to suspend
      or resume the proxy activity relative to the Internet
      agent. It is the same as the administrativeState
      attribute defined in [ISO10165-2].

      The "unlocked" state means that proxy must continue
      to perform, or resume performing, proxy activities
      on behalf of the Internet agent.

      The "locked" state means that the proxy must not
      perform, or suspend performing, proxy activities on
      behalf of the Internet agent.
      !;;

   ATTRIBUTES
      cmipsnmpProxyAgentId GET,
      {iimcManagementDoc 1}:transportAddress GET-REPLACE,
      managementProtocol  REPLACE-WITH-DEFAULT GET,
      supportedMIBs       GET-REPLACE ADD-REMOVE,
      accessControlEnforcement GET-REPLACE,
      accessControlMechanism DEFAULT VALUE
                     IimcProxyASN1.noAccessControl
                     GET-REPLACE,
      "Rec. X.721 | ISO/IEC 10165-2 : 1992":
      administrativeState GET-REPLACE;;;
   REGISTERED AS  { iimcProxyObjectClass 1 };


cmipsnmpProxy MANAGED OBJECT CLASS
   DERIVED FROM
      "Rec. X.721 | ISO/IEC 10165-2 : 1992" : top;
   CHARACTERIZED BY
      cmipsnmpProxyPkg PACKAGE
      BEHAVIOUR
      cmipsnmpProxyPkgBehaviour BEHAVIOUR
      DEFINED AS
```

```
      !This managed object class is used to contain
      objects that represent an Internet agent in the
      proxy MIB.

      The internetAlarm shall be emitted by this object
      class when the application level source of the SNMP
      Trap or Inform Request cannot be determined. The
      address field of the internetAlarm shall be set to
      the network address associated with the SNMP Trap
      or Inform Request.!;;
   ATTRIBUTES
      {iimcManagementDocMan 1}:internetClassId GET;
   NOTIFICATIONS
      {iimcManagementDocMan 1}:internetAlarm;;;
   REGISTERED AS  {iimcProxyObjectClass 2};


snmpSecurityParameter MANAGED OBJECT CLASS
   DERIVED FROM
      "Rec. X.721 | ISO/IEC 10165-2 : 1992" : top;
   CHARACTERIZED BY
      defaultSecurityParameterPkg PACKAGE
      BEHAVIOUR
      defaultSecurityParameterPkgBehaviour BEHAVIOUR
      DEFINED AS
      ! This object instance contains the default security
      parameter that is used only when such AccessControl
      field is not received during association
      establishment.!;;
   ATTRIBUTES
      "iimcManagementDoc 1":internetClassId   GET,
      snmpSecurity              GET-REPLACE
                                ADD-REMOVE ;;;
   REGISTERED AS  {iimcProxyObjectClass 3};


-- 7.1.2 Proxy MIB Attributes


accessControlEnforcement ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
      IimcProxyASN1.AccessControlEnforcement;
   MATCHES FOR EQUALITY, ORDERING;
   BEHAVIOUR accessControlEnforcementBehaviour BEHAVIOUR
      DEFINED AS
      !The accessControlEnforcement attribute indicates
      where access control is applied: Internet agent,
      proxy, or both.!;;
   REGISTERED AS {iimcProxyAttributeID 1 };


accessControlMechanism ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
      IimcProxyASN1.AccessControlMechanism;
```

```
   MATCHES FOR EQUALITY, ORDERING;
   BEHAVIOUR accessControlMechanismBehaviour BEHAVIOUR
     DEFINED AS
     !The accessControlMechanism attribute indicates
     which access control is to be applied at the proxy
     device.  The mechanism may be no access control,
     the internet access control as defined in
     [RFC1446] or one of the ISO access control
     mechanisms as defined in [ISO10164-9].!;;
   REGISTERED AS {iimcProxyAttributeID 2 };


cmipsnmpProxyAgentId ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
     IimcProxyASN1.DistinguishedName;
   MATCHES FOR EQUALITY;
   BEHAVIOUR cmipsnmpProxyAgentIdBehaviour BEHAVIOUR
     DEFINED AS
     !This is the naming attribute for the
     cmipsnmpProxyAgent object class.  This
     attribute will name the system which
     represents the proxied device.!;;
   REGISTERED AS {iimcProxyAttributeID 3 };


managementProtocol ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
     IimcProxyASN1.ObjectIdentifier;
   MATCHES FOR EQUALITY, ORDERING;
   BEHAVIOUR managementProtocolBehaviour BEHAVIOUR
     DEFINED AS
     !This attributes specifies the Internet management
     protocol used for proxy to managed devices.  It
     shall have a value of either SNMPv1 or SNMPv2.!;;
   REGISTERED AS {iimcProxyAttributeID 4 };


supportedMIBs ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
     IimcProxyASN1.SupportedMIBs;
   MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;
   BEHAVIOUR supportedMIBsBehaviour BEHAVIOUR
     DEFINED AS
     !This attribute specifies the set of Internet OIDs
     of registered MIBs that the agent supports.!;;
   REGISTERED AS {iimcProxyAttributeID 5 };


snmpSecurity ATTRIBUTE
   WITH ATTRIBUTE SYNTAX
     IimcProxyASN1.SnmpSecurity;
   MATCHES FOR EQUALITY;
   BEHAVIOUR snmpSecurityBehaviour BEHAVIOUR
     DEFINED AS
```

```
      !This attribute specifies the default values to be used
      when other AccessControl values are not available (i.e.,
      when no AccessControl fields are received during
      association establishment or on a given CMIS
      operation).!;;
   REGISTERED AS {iimcProxyAttributeID 7 };
```

-- 7.1.3 Proxy MIB Name Bindings

```
cmipsnmpProxy-systemNB  NAME BINDING
   SUBORDINATE OBJECT CLASS cmipsnmpProxy;
   NAMED BY SUPERIOR OBJECT CLASS
     "Rec. X.721 | ISO/IEC 10165-2 : 1992": system;
   WITH ATTRIBUTE {iimcManagementDoc 1}:internetClassId;
   BEHAVIOUR cmipsnmpProxy-systemNBBehaviour BEHAVIOUR
     DEFINED AS
     !There is only one object instance of this object class
     in the proxy and this managed object instance should be
     created by the proxy process.  It is not creatable and
     deletable via CMIP management operation.!;;
   REGISTERED AS {iimcProxyNameBinding 1};


cmipsnmpProxyAgent-cmipsnmpProxyNB  NAME BINDING
   SUBORDINATE OBJECT CLASS cmipsnmpProxyAgent;
   NAMED BY SUPERIOR OBJECT CLASS cmipsnmpProxy;
   WITH ATTRIBUTE cmipsnmpProxyAgentId;
   BEHAVIOUR cmipsnmpProxyAgent-cmipsnmpProxyNBBehaviour
   BEHAVIOUR
     DEFINED AS
     !This managed object may be created and deleted
     either by management action, or by local action, of
     the proxy.

     A side effect of the creation/deletion of this object
     shall be the creation/deletion of the ISO system
     managed object associated with the Internet agent.

     The cmipsnmpProxyAgentId contains the DN of
     the proxied device.!;;

   CREATE WITH-REFERENCE-OBJECT;
   DELETE;
   REGISTERED AS {iimcProxyNameBinding 2};


snmpSecurityParameter-cmipsnmpProxyNB  NAME BINDING
   SUBORDINATE OBJECT CLASS snmpSecurityParameter;
   NAMED BY SUPERIOR OBJECT CLASS cmipsnmpProxy;
   WITH ATTRIBUTE {iimcManagementDoc 1}:internetClassId;
   BEHAVIOUR snmpSecurityParameter-cmipsnmpProxyNBBehaviour
   BEHAVIOUR
```

```
     DEFINED AS
     !This managed object is created automatically when the
     containing cmipsnmpProxy object is created.!;;
   REGISTERED AS {iimcProxyNameBinding 3};
```

7.2 Proxy MIB ASN.1 Module

```
IimcProxyASN1 {iimcManagementModMan 3}
DEFINITIONS IMPLICIT TAGS ::= BEGIN
IMPORTS
     iimcManagementProxy, iimcManagementModMan,
     iimcManagementDocMan, iimcIIMCProxy
          FROM IimcAssignedOIDs {iimcManagementModMan 1}

     ObjectIdentifier, AccessControlInfo
          FROM IimcCommonDef (iimcManagementModMan 2}

     DistinguishedName
          FROM InformationFramework { joint-iso-ccitt ds(5)
          modules (1) informationFramework (1) };

-- for ISO/CCITT-Internet proxy specific extension
iimcProxySpecificExtension OBJECT IDENTIFIER
             ::={iimcManagementProxy specificExtension(0)}

-- for ISO/CCITT-Internet proxy object class
iimcProxyObjectClass OBJECT IDENTIFIER
             ::={iimcManagementProxy managedObjectClass(3)}

-- for ISO/CCITT-Internet proxy package
iimcProxyObjectClass OBJECT IDENTIFIER
             ::={iimcManagementProxy package(4)}

-- for ISO/CCITT-Internet proxy attribute
iimcProxyAttributeID OBJECT IDENTIFIER
             ::={iimcManagementProxy attribute(7)}

-- for ISO/CCITT-Internet proxy name bindings
iimcProxyNameBinding OBJECT IDENTIFIER
             ::={iimcManagementProxy nameBinding(6)}

-- SNMP Protocol version
iimcSnmpProtocol OBJECT IDENTIFIER
          ::= { iimcProxySpecificExtension snmpProtocol(1) }

-- the OID for SNMPv1
snmpV1 OBJECT IDENTIFIER ::= {iimcSnmpProtocol snmpV1(1) }

-- the OID for SNMPv2
snmpV2 OBJECT IDENTIFIER ::= {iimcSnmpProtocol snmpV2(1) }

-- the OID for IIMC errors
```

```
iimcErrors OBJECT IDENTIFIER
          ::= { iimcProxySpecificExtension errors(2) }

snmpTooBig     OBJECT IDENTIFIER ::= { iimcErrors 0 }
snmpBadValue   OBJECT IDENTIFIER ::= { iimcErrors 1 }
snmpReadOnly   OBJECT IDENTIFIER ::= { iimcErrors 2 }
snmpGenErr     OBJECT IDENTIFIER ::= { iimcErrors 3 }
noResponse     OBJECT IDENTIFIER ::= { iimcErrors 4 }
cannotDelete   OBJECT IDENTIFIER ::= { iimcErrors 5 }
notImplemented OBJECT IDENTIFIER ::= { iimcErrors 6 }
wrongLength    OBJECT IDENTIFIER ::= { iimcErrors 7 }
wrongEncoding  OBJECT IDENTIFIER ::= { iimcErrors 8 }


AccessControlUsed ::= INTEGER {
                    noAccessControl (0),
                    internet (1),
                    iso (2)}


AccessControlEnforcement ::= INTEGER {
                         agent   (0),
                         proxy   (1),
                         both    (2)}


SupportedMIBs ::= SET OF CHOICE {
                    oid OBJECT IDENTIFIER,
                    ps  PrintableString }


ErrInfo ::= INTEGER {
                       transport(0),
                       authenticationProtocol(1),
                       privacyProtocol(2) }

SnmpSecurity ::=  SEQUENCE {
                snmpOperation    SnmpOpType,
                snmpSecurityInfo AccessControlInfo }


SnmpOpType ::= ENUMERATED {
              getReqPdu(0),
              getNextReqPdu(1),
              getRespPdu(2),
              setReqPdu(3),
              trapPdu(4),
              getBulk(5),
              informReq(6)     }


END
```

8 Conformance Requirements

8.1 Management Communication Requirements

An implementation of the ISO/Internet proxy shall satisfy the following management communication requirements.

- The ISO/Internet proxy shall conform to ISO/CCITT CMIP in the agent role, as specified by [ISO9596-1] and [ISO9595], and profiled by AOM12 [ISO11183-1,2].

- The ISO/Internet proxy shall conform to either Internet SNMPv1 or SNMPv2 in the manager role, as specified by [RFC1157] or [RFC1448].

- The ISO/Internet proxy shall conform to all mandatory security requirements specified by [IIMCSEC] for each supported version of SNMP (SNMPv1 and/or SNMPv2). If proxy supports only SNMPv1, then enforcement of access control and Party MIB support is optional.

8.2 Management Function Requirements

An implementation of the ISO/Internet proxy shall satisfy the following management function requirements.

- The ISO/Internet proxy may optionally conform to ISO/CCITT system management functions in the agent role, as specified by either [ISO10164-5] or [ISO10164-6], and profiled by AOM221 [ISO12060-4] or AOM231 [ISO12060-5].

8.3 Management Information Requirements

An implementation of the ISO/Internet proxy shall satisfy the following management information requirements.

- The ISO/Internet proxy shall support the "system" object specified by [ISO10165-2], in the agent role.

- The ISO/Internet proxy shall support the translated MIB-II "internetSystem" object specified by [IIMCMIB-II], in the manager role.

- The ISO/Internet proxy shall support the IIMC Proxy MIB definitions specified by section 7 of this document, in the agent role.

- The ISO/Internet proxy shall support the IIMC Party MIB definitions specified by [IIMCSEC], in the agent role.

- The ISO/Internet proxy shall support the Internet Party MIB definitions specified by [RFC1447], in the manager role.

- The ISO/Internet proxy shall support one or more translated MIBs, derived in accordance with the procedures specified by [IIMCIMIBTRANS]. For each supported MIB, the ISO/CCITT GDMO translation shall be supported in the agent role, and the Internet MIB shall be supported in the manager role.

- The ISO/Internet proxy shall comply with the information models specified by [ISO10165-1,4] and either [RFC1155], [RFC1212], or [RFC1442].

- The ISO/Internet proxy may optionally support the "OP1 Library Vol.4":capabilityObject" specified by [NMF006], in the agent role.

8.4 Service Emulation Requirements

An ISO/Internet proxy implementation that claims conformance to this specification must state the level of CMIS service emulation that it supports. Two levels are defined:

a) a basic proxy which emulates CMIS kernel services, plus support for scoped CMIS Get; and

b) an enhanced proxy which emulates all CMIS services, including scoping and filtering on all applicable CMIS services.

As noted in section 8.1, the proxy requires support for the Enhanced Management Communications Profile AOM12. That is, the proxy is required to support CMIP kernel, multiple object selection, filtering, and multiple reply functional units. However, a basic proxy is not required to request the filtering functional unit. Furthermore, a basic proxy is not required to carry out scoping for CMIS services other than M-GET, and returns the CMIS error "complexity limitation" for any other CMIS service request which contains a Scope parameter value not equal to "base object alone". These limitations do not apply to the enhanced proxy, which is required to carry out both scoping and filtering for all CMIS service requests.

Editor's Note: [Deletion of Conceptual Table Translation, as proposed in Appendix B of [IIMCIMIBTRANS], would require the basic proxy also to support an M-GET Filter value consisting of a single AVA on the objectClass attributeId.]

9 Acknowledgments

The following individuals have contributed to this effort.

Bob Aronoff - NIST
Jon Biggar - NetLabs
Mary Brady - NIST

April Chang - NetLabs
Ken Chapman - Stratus Computer Inc.
Alice Chen - Boeing
Christopher Crowell - Cabletron Systems
Jock Embry - Opening Technologies
Ian Emsley - Bull S.A
Paul Golick - IBM
Ulrich Gremmelmaier - University of Stuttgart
Pramod Kalyanasundaram - University of Delaware
Ken Hunter - Hewlett-Packard
Lee LaBarre - The MITRE Corporation
David Liu - Northern Telecom
Jim MacLeod - U S West
Reece Markowsky - OSIWare
Subrata Mazumdar - IBM
Keith McCloghrie - Hughes LAN Systems
Owen Newnan - U S West
Steve Ng - MPR Teltech
Yasuhiro Ohara - NTT
Jong-Tae Park - KyungPook National University
George Pavlou - University College of London
Lisa Phifer - Bellcore
Jim Reilly - Technical Rsch Ctr of Finland
Tom Rutt - AT&T
Adarsh Sethi - University of Delaware
Raj Sirsikar - University of Delaware
Baltej Singh - OSIWare
Mark Smith - Hewlett-Packard
Einar Stefferud - Network Management Associates
Mark Sylor - Digital
Hector Trevino - Bellcore
Huy Truong - Tandem
Al Vincent - U S West
Dean Voiss - NetLabs
David Waitzman - BBN
Graham Wisdom - Timeplex
Yoshi Yamashita - NKK Corporation

References

[ISO7498-4] ISO/IEC IS 7498-4, Information Processing Systems - Open Systems Interconnection -Basic Reference Model Part 4 - Management Framework, 1989.

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

[ISO8825] ISO/IEC 8825: Information Technology - Open System Interconnection-Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1),1990.

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

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

[ISO10164-5] ISO/IEC 10164-5: Information Technology - Open Systems Interconnection - Systems Management - Part 5: Event Report Management Function, 1991.

[ISO10164-6] ISO/IEC 10164-6: Information Technology - Open Systems Interconnection - Systems Management - Part 6: Log Control Function, 1991.

[ISO10164-9] ISO DIS 10164-9, Information Processing Systems - Open Systems Interconnection - Structure of Management Information - Part 9: Objects and Attributes for Access Control, ISO/IEC JTC1/SC21/N7661, March, 1993.

[ISO10165-1] ISO/IEC 10165-1: Information Technology - Open Systems Interconnection - Structure of Management Information - Part 1: Management Information Model, 1991.

[ISO10165-2] ISO/IEC 10165-2: Information Technology - Open Systems Interconnection - Structure of Management Information - Part 2: Definition of Management Information, 1992.

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

[ISO11183-1] ISO/IEC ISP 11183-1, Information Technology - International Standardized Profiles AOM1n OSI Management - Management Communications Protocols - Part 1: Specification of

ACSE, Presentation, and Session Protocols for the use by ROSE and CMISE, May 1992.

[ISO11183-2] ISO/IEC ISP 11183-2, Information Technology - International Standardized Profiles AOM1n OSI Management - Management Communications Protocols - Part 2: AOM12 - Enhanced Management Communications, June 1992.

[ISO12060-4] ISO/IEC DISP 12060-4, Information Technology - International Standardized Profiles AOM2n OSI Management - Management Functions - Part 4: AOM221 - General Event Report Management, July 1992.

[ISO12060-5] ISO/IEC DISP 12060-5, Information Technology - International Standardized Profiles AOM2n OSI Management - Management Functions - Part 5: AOM231 - General Log Control, July 1992.

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

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

[RFC1212] RFC1212, M. Rose, K. McCloghrie - Editors, Concise MIB Definitions, March 1991.

[RFC1213] RFC1213, K. McCloghrie and M. Rose - Editors, Management Information Base for Network Management of TCP/IP-basedinternets: MIB-II, March 1991.

[RFC1441] J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Introduction to version 2 of the Internet-standard Network Management Framework, April 1993.

[RFC1442] J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2), April 1993.

[RFC1446] J.M. Galvin, K. McCloghrie, J.R. Davin, Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2), April 1993.

[RFC1447] J.D. Case, K. McCloghrie, M.T. Rose, S.L. Waldbusser, Party MIB for version 2 of the Simple Network Management Protocol (SNMPv2), April 1993.

[RFC1448] J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2), April 1993.

[RFC1450] J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Management Information Base for version 2 of the Simple Network Management Protocol (SNMPv2), April 1993.

[RFC1452] J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Coexistence between version 1 and version 2 of the Internet Network Management Framework, April 1993.

[IIMCIMIBTRANS] ISO/CCITT and Internet Management Coexistence (IIMC): Translation of Internet MIBs to ISO/CCITT GDMO MIBs, Draft 2, May 1993.

[IIMCMIB-II] ISO/CCITT and Internet Management Coexistence (IIMC): Translation of Internet MIB-II (RFC1213) to ISO/CCITT GDMO MIB, Draft 2, May 1993.

[IIMCSEC] ISO/CCITT and Internet Management Coexistence (IIMC): ISO/CCITT to Internet Management Security, Draft 2, May 1993.

[IIMCOMIBTRANS] ISO/CCITT and Internet Management Coexistence (IIMC): Translation of ISO/CCITT GDMO MIBs to Internet MIBs, Draft 2, May 1993.

[NMF006] Network Management Forum: Forum 006, OMNIPoint 1 Library Volume 4, Issue 1.0, August, 1992.

[NMFTR107] NM Forum and X/Open, ISO/CCITT and Internet Management: Coexistence and Interworking Strategy, Issue 1.0, October, 1992.

Appendix A (Normative) Managed Object Conformance Statements (MOCS)

Editor's Note: [This section will be filled in prior to publication. When completed, this section will contain a tabular representation of the managed object classes, attributes, notifications, and name bindings defined in this document. The format of these proforma tables will be as defined by ISO/IEC 10165-6.]

Appendix B (Informative) Example Operation

Operation: CMIS M-GET request
Object class: Internet MIB-II ip
Object instance:{systemTitle = "NetLabs"/internetClassId = ip}
Scope: 2nd level down
Filter: ipRouteType == indirect
Attribute id list: ipRouteDest

Example Internet MIB-II 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 CMIS M-GET request based on the example Internet MIB-II ipRouteEntries.

1. Check the existence of the base object

The ISO/Internet proxy issues an SNMP GetNext 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 GetNext Requests.

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

b) ipRouteEntry: The definition for this object class contains the attribute specified in the filter (ipRouteType),therefore the ISO/Internet proxy issues SNMP GetNext Requests to retrieve the object instances.

The SNMP requests issued and the responses received would be as follows:

```
GetNextRequest  {ipRouteDest, ipRouteType}
GetResponse     {ipRouteDest.192.95.93.1 =
                              192.95.93.1,
                 ipRouteType.192.95.93.1 = direct}

GetNextRequest  {ipRouteDest.192.95.93.1,
                ipRouteType.192.95.93.1}
GetResponse     {ipRouteDest.192.95.93.2 =
                              192.95.93.2,
                 ipRouteType.192.95.93.2 =
                              indirect}

GetNextRequest  {ipRouteDest.192.95.93.2,
                 ipRouteType.192.95.93.2}
GetResponse     {ipRouteDest.192.95.93.3 =
                              192.95.93.3,
                 ipRouteType.192.95.93.3 =
                                    invalid}

GetNextRequest  {ipRouteDest.192.95.93.3,
                 ipRouteType.192.95.93.3}
GetResponse     {ipRouteDest.192.95.93.4 =
                              192.95.93.4,
                 ipRouteType.192.95.93.4 =
                                    other}

GetNextRequest  {ipRouteDest.192.95.93.4,
                ipRouteType.192.95.93.4}
GetResponse     {ipRouteDest.192.95.93.5 =
                              192.95.93.5,
                 ipRouteType.192.95.93.5 =
                              indirect}

GetNextRequest  {ipRouteDest.192.95.93.5,
                ipRouteType.192.95.93.5}
GetResponse     {ipRouteIfIndex = 5,
                 ipRouteProto = 1}
```

c) ipNetToMediaEntry: The definition for this object class does not contain the attribute specified in the filter (ipRouteType), therefore the ISO/Internet proxy concludes that there are 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 CMIS M-GET operation is performed.

3. Form the response

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

```
CMIS M-GET Response (first linked reply PDU):
     Object class: ipRouteEntry
     Object instance:/systemTitle =
  { "NetLabs"/
  internetClassId = ip/
  internetClassId = ipRouteTable/
  internetClassId = ipRouteEntry.192.95.93.2 }
     Attribute list: ipRouteDest = 192.95.93.2


CMIS M-GET Response (second linked reply PDU):
     Object class: ipRouteEntry
     Object instance: { /systemTitle =
  "NetLabs"/ internetClassId = ip/
  internetClassId = ipRouteTable/
  internetClassId = ipRouteEntry.192.95.93.5 }
     Attribute list: ipRouteDest = 192.95.93.5

CMIS M-GET Response (final empty response PDU)

    {internetClassId = ip }
```