AUTOMATING THE OSI TO INTERNET MANAGEMENT CONVERSION THROUGH THE USE OF AN OBJECT-ORIENTED PLATFORM

G. Pavlou, S. N. Bhatti, G. Knight

Department of Computer Science
University College London
Gower Street, London WC1E 6BT
England

Abstract

OSI provides a powerful object-oriented management model that is infinitely scalable and extensible but is only now beginning to see widespread support. The Internet model has followed a less powerful but simpler approach which has led to an already large installed base of manageable equipment. In the future, even if OSI becomes ubiquitous, the two solutions will have to coexist as the level of investment on Internet management technology is already high. A natural path for this coexistence is to provide a conversion from the Internet to the OSI model and use proxy systems that will provide an OSI view of all existing Internet-manageable resources. This paper discusses how the generation of such proxy systems can be automated through a set of well defined conversion rules and suitable object-oriented infrastructure in the form of a generic management platform. The approach followed for the proxy system is mostly stateless, with optimised cacheing for scoping of transient objects and filtering. Complex issues are highlighted.

Keywords: Network operation and management, Local and Metropolitan Area networks, Internetworking.

1. Introduction

OSI provides a powerful object-oriented management model that is both scalable and extensible. OSI management was one of the last applications to be standardised by ISO and as such it received extensive scrutinisation. This led to a powerful object-oriented model which took a long time to be completed and offer the infrastructure for workable solutions. On the other hand, a different approach was followed by the Internet community in which a simple solution was chosen with the primary goal of having minimal impact on the systems to be managed. This approach proved to be very successful in the short term, with many vendors of networking equipment offering SNMP-manageable devices. Recently, the OSI management approach has started gaining ground, with the first implementations becoming available. The current investment on Internet management technology is already high enough to suggest that a coexistence of the two solutions should be expected in the years to come.

The scene for this coexistence was first set by the specification of the Internet MIB in OSI form [1] two years ago which suggested that such a translation was possible. We actually designed and implemented immediately an OSI agent providing a view of the "standard" Internet MIB in OSI fashion in order to experiment with that approach. That design and implementation was based on our generic object-oriented management platform in which we had researched issues of generic infrastructure and extensibility through well defined object-oriented Application Program Interfaces (APIs).

Our approach was a "Dual Agent" one and despite its limitations (see section 3) served to highlight that OSI management implementations could be simple and highly efficient, as certified by comparative performance measurements against the equivalent SNMP agent implemented using the same basic infrastructure. Soon though it became apparent that an application gateway approach would have many additional advantages and this was pursued next. In such an approach, the OSI agent acts as a gateway that accesses the equivalent Internet agent from its back-end in a proxy fashion.

What became apparent during that work was that if a set of deterministic rules for Internet to OSI Management Information Base conversion was specified, the production of such application level gateways could be automated to a large extent. We derived those rules based on the standard Internet MIB translation and designed and implemented such a system. In doing that, we realised that the support our platform was providing already tackled the most complex aspects of such the conversion, leaving very little to be done. In fact, revisiting that work we realised that such a conversion could be completely automated with a minor exception, the automation starting simply from the formal specification of an Internet MIB.

It should be added that work on setting up the rules for MIB conversion and the architecture of such a gateway system are currently under way [2] [3] [4]. Our approach for the conversion is similar but our system architecture is different as it takes the whole procedure a step further by suggesting mechanisms to automate it completely. Also this work is based on SNMP version 1 and it may need to be modified in the light of SNMP version 2 developments.

2. Management Models

The two different approaches taken by OSI management and Internet management are characterised by their very different Management Information Models (MIMs) i.e. the way that they present the management information. Although both use abstract representation of resources to be managed known as Managed Objects (MOs) and a manager-agent access paradigm, their respective specifications for the Structure of Management Information (SMI) follow different philosophies. As a first step on approaches towards their integration, the two models are explained and compared.

2.1 The OSI Model

The OSI Model provides a powerful and general management infrastructure by adopting a fully-fledged object-oriented approach. Logical or physical resources to be managed are manipulated through managed object abstractions which are handled by applications in agent roles and are accessed by applications in manager roles that realise management policies. Communication between manager and agent applications is through the OSI Common Management Information Service/Protocol (CMIS/P) [5] [6], which conveys information of operations to be performed on managed objects. Management information about resources to be managed is specified in Management Information Base (MIB) definitions while an agent application contains a MIB instance that may pertain to a multitude of resources in the local or other systems.

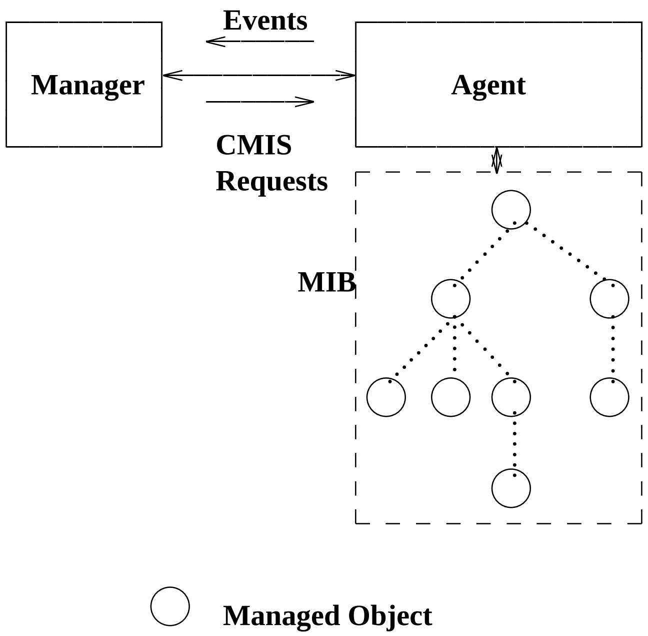

Figure 1. The OSI Management Model

A managed object may contain attributes, perform actions, emit notifications and exhibit behaviour at its boundary which encapsulates the underlying resource. Managed object classes are formally specified in an ASN.1 template language known as Guidelines for the Definition of Managed Objects (GDMO) [7]. There are two main aspects in management information modeling: inheritance and containment. All managed object classes form a global inheritance tree whose root is the `top` object class that contains self describing information for every object instance. Polymorphism is expressed through a set of access methods common to all instances, the same as the management protocol operations. Also, allomorphism is a limited form of polymorphism, that provides the ability to view a managed object as an instance of any of its parent classes in the inheritance tree. This is a powerful mechanism to cater for information model extensions without hindering interoperability.

Containment is used to model managed object relationships for the purpose of naming. Managed objects in a MIB instance form a Management Information Tree (MIT) according to these containment relationships, which are also known as "name bindings". Every managed object instance instance has a Relative Distinguished Name (RDN) which is unique within the scope of the containing object. This is formed through the type and value of a naming attribute. The sequence of relative names from the top object of the local containment tree until an actual object form the object’s Distinguished Name (DN), this is unique within the local system (agent). The top object of the containment tree is always of class `system`, which represents the system (agent) being managed as a whole.

CMIS/P supports the following operations:

CMIS operation	Purpose
M-Get	retrieve information
M-Cancel-Get	cancel a previous M-Get
M-Set	set attribute values of MO(s)
M-Action	invoke an action on MO(s)
M-Create	create a MO instance
M-Delete	delete a MO instance(s)
M-Event-Report	send an event report

The M-Event-Report is an asynchronous operation that sends a notification emitted by a managed object to an interested manager. The other operations are performed in the opposite direction and, apart from M-Cancel-Get, express operations to be performed on managed objects. The M-Get, M-Set, M-Action and M-Delete may be performed on multiple objects of a containment subtree through scoping. Furthermore, elimination is possible through filtering i.e. expressing logical assertions on attribute values.

2.2 The Internet Model

The Internet Model is can be seen as a simple version of the OSI one in which object-oriented aspects are restricted to a bare minimum. This is even reflected in the terminology used, as the term "object class" is carefully avoided with "object type" used instead which has different connotations. In this model, objects are effectively single value items i.e. pieces of information that may be read or written, affecting the operation of the associated resource. They are formally specified using a single OBJECT-TYPE ASN.1 template. The only objects that may have multiple instances are table entries while there is no notion of either inheritance or containment relationships. Instead, a simple architecture is used for naming, imposing a lexicographically ordered sequence on all the objects in an MIB instance.

All object types are registered in the global Object Identifier (OID) naming tree, in the same way as OSI classes, attributes, actions, notifications and name bindings. Objects with a single instance are named by suffixing the type (class) identifier with a zero (".0") while multiple instance objects (fields of table entries) are suffixed by the value(s) of the "key", or index, field(s) for the entry in an object identifier form. This scheme guarantees uniqueness of names and ensures that all the MIB instances are lexicographically ordered. MIB instances are held by agent applications and are accessed by manager ones. Management information is conveyed by the Simple Network Management Protocol (SNMP) [8] which supports the following operations:

SNMP Operation	Purpose
Get	retrieve information
Get-Next	retrieve information by MIB traversal
Set	set variable values or create/delete table entries
Trap	inform manager of an event

The Trap is an asynchronous operation similar to the OSI M-Event-Report while the rest express read and write operations to be performed on managed objects. The Get-Next is different to Get as it does not require full object instances but any object identifier and it will return the next lexicographically ordered object instances. This is necessary in order to find-out if an object is supported by an agent and also to traverse tables. Creation and deletion of table entries is possible by emulation through Set.

2.3 Model Comparison

The two models are similar to some extent but also have many differences. The main difference is that the OSI model uses an event-driven approach as opposed to the Internet one which follows a polling approach. Traps in SNMP are not configurable and also are not guaranteed to be received by managers as an unreliable connectionless transport mechanism is used (the Internet UDP [9]). As such, managers have to resort to low frequency polling. This difference has repercussions on the complexity of managing and managed systems (managers/agents), SNMP agents being simple and shifting all the complexity to manager applications, which is one of reasons for the current proliferation of SNMP-manageable devices.

Get-Next provides some of the functionality of CMIS scoping but with a lot more interactions which may consume bandwidth. Filtering has no counterpart in the SNMP model. The same holds for M-Cancel-Get and M-Action: the former is meaningless in the absence of scoping while the latter may be emulated through the SNMP Set and suitable information modelling. M-Create and M-Delete in the Internet model can be emulated through Set for table entries only, and not always successfully as there are weak semantics. Creation may be emulated by setting all the fields of a table entry while deletion by setting one of the fields to a value rendering the entry invalid.

3. Integration Model

There are two basic models for integrating the OSI and Internet management approaches, integration at the manager or agent level.

3.1 Integration by Managers

One possible approach is to have managers that understand both the OSI and Internet models and know which protocol they should use for each instance of communication. However, this approach requires extra complexity to be built in managers and assumes "dual-stack" support as the manager will need separate communications infrastructure for both the OSI and Internet worlds. Such an approach is adopted by the X/Open XMP [10] Application Program Interface (API) which provides uniform access to CMIS and SNMP services. However, it is impossible to conceal which model the manager deals with as the two information models have important differences.

3.2 Integration by Agents

This approach assumes that a translation is possible from an Internet to an equivalent OSI MIB and there are two methods for providing such a solution, the dual agent or application gateway approach. In this case, a dual stack is needed by agents but managers have a single view of managed resources, based on one of the models.

3.2.1 The Dual Agent Approach

This approach assumes that each MIB is implemented separately, in both OSI and Internet fashion. This means that each system supports two agents, every Internet agent being re-implemented in an OSI fashion. This approach requires of course a significant amount of new investment. An additional problem is that the same real resources can be accessed independently through those two agents and this may result in concurrent access problems that will need to be resolved. In this approach, a manager may take either an Internet or OSI view.

3.2.2 The Application Level Gateway Approach

A better approach is that of an application-level gateway or proxy agent. In this case, an OSI agent can present an OSI view of resources managed by an Internet agent by accessing the resources not directly but through the latter. The mapping between Internet and equivalent OSI agents is logically one-to-one but at the physical level one OSI agent may represent many Internet ones. The first advantage over the dual agent approach is that the total number of agents is not double. This reduces the systems which need a dual stack and also eliminates the problem of concurrent access to the real resources. In this approach, OSI managers have an OSI view of resources while Internet managers may operate locally. The real advantage of this approach though lies in the fact that the similarities between the two models allow one to define a set of deterministic rules for the conversion between the two. In this case, the production of these application gateways can be almost completely automated as this paper argues.

3.2.3 Limitations of the Application Level Gateway Approach

Though this approach adds manageability to a whole world of resources with Internet management capabilities, it does little to exploit the event-driven nature of the OSI management model. In fact, the usage of the OSI model is undermined and restricted only to the semantics of the Internet one. For example, an OSI application that needs to perform remote monitoring of resources in a remote local-area network connected to a wide-area backbone will have to use polling on a wide-area scale. In this particular case, the OSI management power is clearly underused.

A better approach would be to enhance the information model that resulted from the conversion with additional information (thresholds, tide-marks etc.) which will support the generation of notifications or to associate additional generic notifications such as object creation, deletion etc. In this case one could still maintain the highly automated generation of the proxy system through augmented GDMO which could be used by a suitable compiler to produce a proxy system that will periodically poll the Internet agent(s) to support the additional notifications. Polling in this case is restricted to the local environment as proxy systems should be kept close to the resources for which they present an OSI view through the Internet back-end.

4. Management Information Base Conversion

As pointed out in the previous section, the two information models have similarities but are different enough to pose a number of problems when trying to convert one to the other. The main difference is that the OSI model follows a fully object-oriented event-driven approach while the Internet one is based on simpler remote-debugging paradigm.

The difference in the expressive power and complexity of the two models means that there may be many ways to perform the conversion between them and also that semantics may be lost in such a conversion. This is true for converting from the more complex (OSI) to the simple one (Internet). The opposite conversion though is possible, as the Internet can be treated as a subset of the OSI model and moreover, it may be performed according to a set of deterministic rules, and so the construction of proxy systems performing the conversion can be automated to a great extent.

4.1 Managed Objects

The main difference between the two models lies in their use of the managed object concept. In both models, managed objects model physical or logical real resources to be managed but the degree of object-orientation and scale is different. For example, a whole protocol machine is an OSI managed object but an Internet "object group", with information held by it (e.g. pdusSent etc.) being Internet managed objects. This observation leads to the natural conversion rule that a SNMP object becomes an OSI attribute that belongs to an object class, which in turn is defined by the containing SNMP group. The only exception to this rule relates to tables contained in a group, the reason being that each table entry should be addressable as a separate object. The rule can be extended to model tables and table entries as separate managed object classes.

4.2 Naming and Containment

As explained, there is no notion of containment in an Internet MIB so a means must be found to name a converted object class and specify its containment relationships (name bindings) to other classes. When a group is converted to an OSI object class, a naming attribute needs to be defined for use in its relative name. This should be a new attribute as there is no general rule for using an existing one apart from table entry objects, but even in this case the values of more than one attributes may be needed and this is not allowed for relative names in OSI management (unlike the OSI Directory). A general rule is to devise a new attribute whose value should be "nil" for single instance objects and equal to the value(s) of the key/index attribute(s) for multiple instance ones i.e. tables entries. A simple rule for the name of that attribute is to use the class name with the string "Id" as a suffix, as is common practice in OSI information modelling.

Regarding the containment relationships of converted object classes, the following rules may apply. The Internet system group is converted to an object class that may be positioned anywhere in an OSI tree, what is important is the relative position of other converted classes with respect to this one. A simple solution is to have classes corresponding to other Internet groups bound to the Internet system object, with the exception of tables and table entries. Table entries are bound to tables while tables are bound to the class modeling the group they belong to or to the Internet system if they constitute a group of their own. If in the future Internet MIBs model groups within a group, this will be naturally modeled by containment relationship of the corresponding OSI classes.

4.3 Notifications

In OSI, a managed object may emit notifications according to the operation of the real resource it represents. These are separate managed objects (specific event log records according to the event type) and their information may be transmitted to a manager through an event report management operation or they may be stored locally in logs according to the Event Report [11] and Log Control [12] functions in the agent.

In the Internet model, traps are roughly equivalent to notifications but these are few and their emission cannot be dynamically controlled by a manager. They are essentially part of the protocol and not of the information base which poses a problem in converting them to notifications, as the class to which they should be associated cannot be determined in a generic fashion. It will always be possible to associate each of them to one of the converted classes but there is no explicit information in an Internet MIB to facilitate the automation of this mapping.

4.4 Registration

Information in both Internet and OSI MIBs is registered in the global registration tree in the form of object identifiers i.e. a series of numbers from the root to the registered leaf entity. In the case of the Internet model the registered information is object groups and types (classes) and the concept is extended to provide a naming architecture for object instances through a suffix that ensures name uniqueness as explained.

When converting an Internet to an OSI MIB, the Internet identifiers could be re-used to identify classes (groups) and attributes (objects) but new identifiers should be specified for non-existing Internet entities such as name bindings, naming attributes and notifications. For registration coherence, a better solution would be to generate new object identifiers for everything. These may be generated from the class identifier by replacing an initial Internet-specific portion with a predetermined value and suffixing it with a unique one that will retain the last part of the group/objects, while it will cater for unique name binding and notification identifiers. There are many ways to generate unique new identifiers, the important aspect is to standardise one approach which will be carefully selected to to avoid future name space collisions but also it should be simple enough to ensure the efficient identifier translation by proxy systems.

4.5 Automating the Conversion

The conversion from an SNMP MIB specified using the OBJECT-TYPE ASN.1 template to the equivalent OSI MIB in GDMO format can be automated through a translator that follows the above rules. The identification of groups, tables and table entries is not explicit in an Internet MIB but can be accomplished in the following fashion. Groups can be identified as they share a common prefix identifier which is known for each MIB Module. Tables can be identified as they are the only objects whose syntax is ASN.1 SEQUENCE OF while table entries are the only objects containing an INDEX clause for naming their instances. Containment reationships between groups, tables and entries can be identified by examining the prefix of the registration identifier.

Having produced the equivalent OSI MIB in GDMO format, it is possible to automate completely the generation of a proxy system for that MIB through a special GDMO MIB compiler that will produce runtime support for a platform already providing a lot of the functionality of a proxy system (see later). In order for this to happen, additional information is needed in the GDMO MIB for table and table entry objects as these need to be treated specially. This information needs to identify which objects are tables and table entries and for the latter the names of the attributes whose values are used for naming and the name and value of the special attribute that serves for entry deletion. This information may be produced by the MIB translator in the form of an augmentation in the BEHAVIOUR clause for the name binding in the GDMO.

5. The Generic Managed System

Building an OSI proxy system for Internet agents is a difficult task because the power of the OSI management model comes at the price of some complexity. The real power of the OSI model lies in the features known as scoping and filtering. Implementing such issues and also synchronisation, Event Reporting [11] and other generic systems management functions is a difficult issue for OSI agents in general. Proxy systems pose an additional problem because of the interactions required through the network at the agent's back-end to access the "loosely-coupled" resources. Proxy systems in which the back-end protocol is connectionless may require retransmissions to ensure delivery and time-outs to avoid resulting in the OSI manager hanging-up in the case of a subordinate agent failure.

Since OSI uses an object-oriented specification methodology, using object-oriented design methods and implementation technology can result in harnessing its power and complexity behind well though-out Application Program Interfaces (APIs) and providing a generic and extensible management platform. The OSI Management Information Service (OSIMIS) [13] platform does exactly that, and the support provided for the development of management agents is known as the Generic Managed System (GMS).

5.1 Support for event-driven system organisation

Using object-oriented design methodology, the Coordinator/Knowledge Source abstraction in the GMS may be used to organise communication with the real resource. An OSI agent can be implemented as one operating system process with one Coordinator object instance listening to all external communication endpoints or internal timer alarms and exercising a fully event-driven policy by serialising all requests and ensuring they will be taken to completion. Knowledge Sources are general application objects that may register external communication endpoints on which they expect to receive information and also request "wake-up" calls periodically to implement a polling regime.

5.2 Support for transparent abstract syntax handling

ASN.1 compilers may be used to produce methods for manipulating abstract syntax values. A general ASN.1 object may be defined encapsulating these methods and automating aspects such as encoding, decoding, filtering etc. For example, decoding, checking the range of permitted values and setting a management attribute can be fully automated, leaving only the interaction to the real resource to be implemented. More importantly, the same is true for filtering as compare methods may be used to evaluate any CMIS allowed filtering assertion (equality, ordering, set comparison).

5.3 Support for CMIS agent functionality

Managed objects in an OSI agent are autonomous entities that should only be concerned with the interactions with the associated real resource and not with details of access through the management protocol. The knowledge of the protocol and associated issues such as parameter checking, object addressing, access control, scoping, filtering, synchronisation and handling of errors and linked replies can be encapsulated by an instance of an object known as the OSI Agent.

Managed object instances are represented internally as C++ [14] object instances linked in a containment tree. Operations such as scoping and filtering are handled by the agent which may request that an object refreshes its subordinates if these are transient objects or refresh the contents of an object before filtering. The knowledge for evaluating the assertions in the filter exists already in the attributes within a managed object. Optimisation mechanisms are applied to ensure that information is not unnecessarily requested. Atomic synchronisation of operations across managed objects is implemented with a double pass mechanism and an internal two phase commit protocol.

5.4 Managed Object support

Managed object instances are represented internally as C++ object instances linked in a containment tree. Generic behaviour for access through the management protocol is inherited. A lot of checks can be automated such as the validity of an operation and values involved. A very simple interface for the interaction with the associated real resource can be presented to MIB implementors. Class knowledge can be automatically produced through the use of a GDMO compiler. Additional support includes the set of generic attribute types such as counter, gauge, threshold and tide-mark and support for systems management functions such as event reporting and logging. Also various methods of access to the associated resource can be used such as fetch-upon-external CMIS request, cache-ahead through polling or updating through asynchronous reports for clever resources. The Knowledge Source functionality may be used to receive information from loosely coupled resources as for example subordinate agents.

6. Service Mapping Using the GMS

As explained in the previous section, the GMS infrastructure will allow incorporation of existing management facilities into the agent implementation. In our implementation of the CMIS to SNMP gateway, we will use specially enhanced Knowledge Source objects to allow interaction with the SNMP agent.

This model has three main features:

1. It is a semi-stateful approach in that the OSI agent maintains a MIB instance that is effectively a MIB Image of the information held at the SNMP agent.

2. The OSI agent has no knowledge of SNMP. The use of SNMP is as a back-end protocol that provides interaction between the object instances and the real resource - in this case the real resource role is taken by the SNMP agent. The object instances effectively act in a SNMP-manager role.

3. Traps are initially handled separately, "outside" the MIB instance. A special Trap-Sink function must act as an SNMP manager accepting Traps and effectively demultiplex them to the correct MO instance where they will be emitted as notifications.

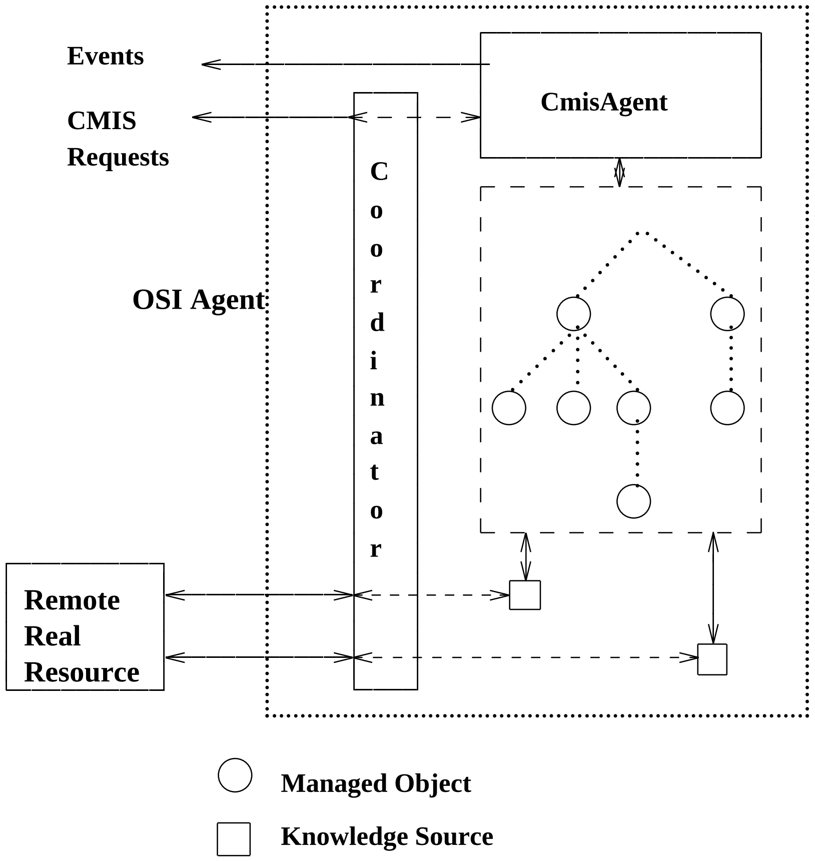

Figure 2. The GMS Architecture

6.1 The MIB Image

A SNMP MIB would be automatically translated to a Converted MIB in GDMO format. An OSI agent maintains an MIB instance that is produced from the Converted MIB using a suitable GDMO compiler. This instance acts as a MIB Image of the information from the SNMP agent. When object instances are selected to be involved in management operations, the values of their attributes may be updated by obtaining "fresh" information, so that they can reflect accurately the state of the SNMP agent.

In the GMS, when a CMIS request is received, a scope operation is performed on the object instances, during which Table object instances would "refresh" their Table-Entry objects by interacting with the SNMP agent. Once the scoping operation has been done, the remaining objects are refreshed only if an operation requiring their attribute values is performed, e.g. applying a filter or for the execution of a CMIS M-Get operation. The exceptional refresh behaviour for Table objects is to allow for Table-Entry objects that are often of a very dynamic nature.

When objects are updated in the GMS implementation, several possible methods are used for refreshing information. The choice between which method is employed is determined at compile time, and is an attempt to optimise interactions with the SNMP agent. If the object instance is a Table object, it refreshes its selected subordinates, which are Table-Entry objects. Any objects that are not Table objects or Table-Entry objects refresh themselves. If a Table object has any attributes other than a naming attribute, it will need to refresh itself as well as its subordinates.

Note that when a Table object is updating its Table-Entry objects, it should first check if the corresponding SNMP table entry has not been "deleted", i.e. declared invalid, by checking the information received from the SNMP agent. For example, if an ipRouteEntry object is deleted it has a value of *invalid(2)* for ipRouteType.

6.2 Interacting with the SNMP Agent

In interacting with the SNMP agent, a object instance in the OSI agent effectively assumes the role of a SNMP manager. To interact with the SNMP agent, the object instance must identify the SNMP objects that it needs to access. Recall that a significant effect of the MIB Conversion operation was to map SNMP objects to OSI attributes. Hence an object instance must map the identifiers of its attributes to identifiers of SNMP objects. After that, the object instance is ready to map CMIS to SNMP requests. Having established the SNMP object identifiers for the attributes involved in the operation, the object instance can map the CMIS request to one or more SNMP requests as explained below. The only CMIS operation that is meaningless in this context is M-Action as these are not present in the Converted MIB.

6.2.1 Mapping CMIS M-Get and M-Cancel-Get

For non table-entry objects, a M-Get is mapped to a SNMP Get for the requested attributes. Before a request for a table entry is serviced, the containing table refreshes the entry when the latter is addressed either through scoping or explicit naming. All the attributes of each entry are requested (cacheing) to avoid accessing it again for a filter or M-Get operation. This involves an overhead in the case of a M-Set operation but this design choice is justified by the fact that Set operations are much less frequent than Get operations.

A table may be requested to refresh a single entry, referenced by its relative name or all its subordinate entries as a result of a scope operation. In the former case, a SNMP Get operation is used with the identifier suffix being formed from the value of the naming attribute present in the relative name. In the latter case, the SNMP Get-Next operation is used to traverse the whole table retrieving a row at a time. In general, if an error is reported in a SNMP Get-Response, it should be mapped to a CMIS error as follows:

SNMP Error	CMIS Error	Specific CMIS Error Information
tooBig	processingFailure	-
noSuchName	getListError	noSuchAttribute
genErr	processingFailure	-

However, a noSuchName error will never be returned as the validity of the requested attributes in the operation will be checked by the object instance in the OSI MIB Image. In fact, the GMS will take care of this automatically through the knowledge of the Converted MIB. Also, a tooBig error can occur only once, as knowledge will be retained in the object instance and subsequent SNMP requests will be resized appropriately, i.e. the tooBig error is handled locally.

The CMIS M-Cancel-Get will be serviced by the OSI agent. However, it may not be possible to prevent further interactions with the SNMP agent as there is no equivalent SNMP service.

6.2.2 Mapping a CMIS M-Set

When an M-Set is received, this will map to a SNMP Set request using the same identifier mapping as described above. In general, if an error is reported in a SNMP Get-Response, it should be mapped to a CMIS error as follows:

SNMP Error	**CMIS Error**	**Specific CMIS Error Information**
noSuchName	setListError	noSuchAttribute
badValue	setListError	invalidAttributeValue
tooBig	processingFailure	-
genErr	processingFailure	-

The noSuchName and tooBig errors are handled as in the case of M-Get. The badValue error should never occur as semantic knowledge is available in the Converted MIB. The value will be decoded in the OSI object instance and its correctness will be checked, so the SNMP Set operation will only be issued after this check.

There are two special cases to cater for. These are a consequence of the use of the SNMP Set for the creation and deletion of table entries. This create and delete behaviour should not be allowed via CMIS as it destroys the M-Create and M-Delete semantics respectively. The mapping of the CMIS M-Create and M-Delete to SNMP is described below.

6.2.3 Mapping CMIS M-Create and M-Delete

In SNMP only table entries can be created and deleted. To create a table entry, the SNMP agent must be sent a Set request with values for each column of that entry. So, the CMIS M-Create must specify values for all the attribute values in the object instance. To delete a table entry in an SNMP agent, the value of a designated column is set to a value to indicate that this entry is "invalid" i.e. effectively deleted. If an error is reported in the SNMP response, it is mapped to a CMIS error as described above.

6.2.4 Mapping a CMIS M-Event-Report

This mapping is different to the ones described so far as it is from agent to manager and not manager to agent. The processing of the incoming Trap involves identifying the Trap type, deducing the SNMP "object" which it relates to, and then identifying the MO instance in the MIB image to which it must be forwarded. The automatic mapping of SNMP Traps to OSI Event-Reports is still under investigation.

7. Complex Issues

Although much could be automated in the SNMP SMI to OSI GDMO conversion process, some issues regarding the "Generic Mapping" and also the implementation of the gateway service require further investigation. These issues are presented in this section, along with a description of the problems involved in their respective realisations.

7.1 Synchronisation and Atomicity Restrictions

Although SNMP forces "all or none" in Get and Set operations, the situation can be complicated by the introduction of the gateway from the OSI world to the SNMP world. Consider the following situation; a manager makes a request, say a M-Set, operation with atomic synchronisation, and, after scoping and filtering, two MO instances are selected. In the suggested architecture, each MO instance must interact with the SNMP agent independently, so it is difficult to perform the operation atomically. The OSI agent could try to roll-back if one of the responses to the SNMP Set indicates an error. However, the ability to roll-back would rely on the operation being reversible in the SNMP MIB. An alternative solution could be that all MO interactions with the SNMP agent must be through a single SNMP-Manager function for the MIB image. However, this would require additional complexity in multiplexing and demultiplexing to and from the SNMP-Manager function and object instances in the MIB Image.

7.2 Converting Traps to Notification Types

It has already been explained that a SNMP trap is part of the protocol and not of the MIB definition. When converting an Internet to an OSI MIB though, traps become notifications associated to a particular object class according to their semantics. However, there is no generic mechanism to invoke the equivalent notification when receiving a trap by simply examining its contents and the knowledge of the Converted MIB. A table needs to be maintained in the trap-sink function within the gateway to enable this mapping. The generation of this table could be automated by introducing manual augmentations in the Converted MIB.

7.3 Access Control

Access control in the Internet Model is by use of "community names". A community name is a name identifying a particular community that is assigned a particular view of the SNMP MIB, and is an OCTET STRING transmitted in every SNMP Message. For this to be used via CMIP, the community name could be sent in the access control parameter of a CMIS call. However, a method for performing this function is currently undefined. Also, the much stronger OSI Access Control [15] is weakened by the use of SNMP, so it may be pointless to use the OSI Access Control model on the Converted MIB at the gateway when the access to the SNMP agent is not as strongly protected.

7.4 MIB Images in the Gateway

One can envisage a scenario where the OSI agent is acting as a gateway to many SNMP agents all having the same SNMP MIBs. Using the methods described in this paper, this would result in a separate MIB Image for each SNMP agent. This may lead to exhaustion of resources at the OSI agent’s host, and restricts the number of SNMP agents accessible through that gateway. In this case, it may be more resource efficient to use just one MIB Image in the OSI agent, and service all the requests for management operations by updating this single MIB Image, effectively using it as a cache from the appropriate SNMP agent. However, the implications of this approach have yet to be fully investigated.

8. Conclusions

We have explained in this paper how the Internet to OSI management conversion can be completely automated through the use of a well defined set of rules for information model conversion and suitable generic object-oriented OSI management infrastructure. The use of the latter is essential as it can take care for most of the complex aspects of the mapping such as the CMIS scoping and filtering. It can also provide a generic mechanism for integrating MIBs with diverse communication needs to managed resources, as is the case with proxy systems.

This approach opens a whole world of management possibilities by making possible to manipulate through OSI management the whole set of Internet-manageable resources and this with the minimum additional investment. The only problem is that the polling-based Internet management paradigm is transferred on OSI environments. This could be rectified by allowing enhancements of the converted information model to increase the level of notifications and still maintain the almost completely automated procedure described. This is a research topic we will pursue in the future.

Acknowledgements

This paper was produced under the ICM (Integrated Communication Management) and MIDAS (Management In a Distributed Application and Service environment) research projects funded respectively by the European Commission RACE (Research in Advanced Communications in Europe) and ESPRIT (European Strategic Program for Research in Information Technology) research programmes.

REFERENCES

[1] L. LaBarre (Editor), *OSI Internet Management: Management Information Base*, RFC 1214, April 1991.

[2] L. LaBarre, *ISO/CCITT and Internet Management Coexistence (IIMC): Translation of Internet MIBs to ISO/CCITT GDMO MIBs*, Internet Draft, October 1992.

[3] A. Chang, *ISO/CCITT and Internet Management Coexistence (IIMC): ISO/CCITT to Internet Management Proxy*, Internet Draft, October 1992.

[4] P. Kalyanasundaram, A. Sethi, *An Application Gateway Design for OSI-Internet Management*, 3rd International Symposium on Integrated Network Management, San Francisco, April 1993.

[5] ISO/IEC 9595, *Information Technology - Open Systems Interconnection - Common Management Information Service Definition*, 1990.

[6] ISO/IEC 9596, *Information Technology - Open Systems Interconnection - Common Management Information Protocol*, 1990.

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

[8] J. Case, M. Fedor, M. Schoffstall, J. Davin, *A Simple Network Management Protocol (SNMP)*, RFC 1157, May 1990.

[9] J. Postel, *User Datagram Protocol*, RFC 768, November 1980

[10] X/Open Preliminary Specification, *Systems Management: Management Protocols API (XMP)*, The X/Open Company Limited, July 1991.

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

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

[13] G. Pavlou, S. N. Bhatti, G. Knight, *The OSI Management Information Service User’s Manual*, Version 1.0, February 1993.

[14] M. A. Ellis, B. Stroustrup, *The Annotated C++ Reference Manual*, May 1991, Addison Wesley.

[15] ISO/IEC CD 10164-9, *Information Technology - Open Systems Interconnection - Systems Management - Part 9: Objects and Attributes for Access Control*, February 1992.