ATNP JWG? IP -
June 23, 1998

**Aeronautical Telecommunication Network Panel**
Joint Working Group (Network Management)
Utrecht, Netherlands

Information Paper
Draft

# Proposed ATN Management Systems Platform OSIMIS Model

**Prepared by: Pam Tupitza**

**Presented by: Pam Tupitza**

**Summary**

This paper provides the joint working group with a new proposed architectural model to resolve network and system management issues in global domains. This model is proposed from research developed by OSI Management Information Service (OSIMIS).

# Table of Contents

# 1.0 Introduction

ATNP has the task of defining a model for network and systems management that provides a common worldview of the ADL network. ATNP has adopted OSI standards as the base model for their SARPS as they are internationally accepted. The OSI model for the definition of network management is an object-oriented model. This is a departure from the standards developed for other areas of the communication standard. This follows the needs of today's systems to perform remote configuration of network elements, assess and optimize the performance of other elements, and offer intelligent reasoning capabilities for diagnosing network faults.

There are many issues surrounding the actual level of definition for the ADL network management model. There are debates on the need for SARPs both inter-domain and intra-domain. There are concerns in the domains on the loss of proprietary information, the cost of developing 'another' network management system for the ADL in addition to their own domain network management systems. Given the many concerns expressed this paper proposes an alternative network management model. This proposed model would provide a generic set of network management functions and information, which would 'hide' the underlying protocol complexity and provide a powerful information model. It can also 'hide' distributed information access by using virtualization made possible by the use of object-oriented techniques.

This model is based on the implementation of an object-oriented management platform based on the OSI model and developed by the University College of London. It has been used for a variety of research projects including RACE ICM, NEMESYS, ESPRIT MIDAS. In other words, the 'theory' of the model has been proven by other implementations.

The unique set of needs and concerns ATNP faces in the definition of network management might find a solution in this proposed model. OSIMIS provides a model that shows how 'the rich OSI Management functionality can be exploited and to serve as a generic OSI management platform.' OSIMIS was designed from the beginning with the intent to support the integration of existing systems with either proprietary management facilities or different management models. Different methods for the interaction with real managed resources are supported; encompassing loosely coupled resources, as it is the case with subordinate agents and management hierarchies. OSIMIS provides already a generic application gateway between CMIS and SNMP while a similar approach for integrating OSI management and the OMG CORBA framework may be pursed in the future. Using OSI management as an end-to-end integrating model is also in line with the NMF OMNIPoint approach which suggests a multiple technology architecture for future heterogeneous environments.

OSIMIS provides a model that fits the ATNP problem. The ability of the platform to integrate with proprietary management facilities addresses the concerns of developing dual SM applications. This would also provide a solution that potentially resolves the contradiction of requirements that the ATNP would use OSI standards and at the same time provide a simple protocol solution.

### 1.1 References

ISO/IEC 10165-1:1993 *Information technology - Open Systems Interconnection - Structure of Management Information: Management Information Model*
ISO/IEC 10165-4:1992 *Information technology - Open Systems Interconnection - Structure of management information - Part 4:Guidelines for the definition of managed objects.*
ISO/IEC 10164-4: *Information technology - Pen System Interconnection - Systems Management - Part 4: Alarm Reporting Function*
ISO/IEC 10164-5: *Draft Recommendation - Event Report Management Function*
ISO/IEC 10164-11*: Information technology - Monitor Object MO classes*
OSIMIS 8.0: *The OSI Management Information Service Platform. University College London. Http://www.cs.ucl.ac.uk/research/osimis*
Object-Oriented Networks, *Subodh Bapat, Prentice-Hall, Inc.*

## 2.0 OSIMIS Platform Functionality

This model focuses on the idea of a generic network management platform as opposed to the traditional network systems. A platform provides a basic set of functionality of network management for many different network components and systems and a generic protocol to allow the access of management information between systems. In addition it offers object-oriented agent support and higher-level manager support to hide the complexity of CMIS services. The System Management SARPs need to specify the standard for a generic open network management platform to provide a generic functionality for managing the variety of ATN devices and systems. States can use these generic standards for the ATN platform in developing their own management systems. This will provide system interoperability while allowing states to keep their proprietary solutions.

The ATN network management platform should provide standards for the following:

- High level object-oriented API's
- Generic applications such as browsers, gateways, directory services, etc
- Generic management applications

OSIMIS is built as an environment using ISODE. Following are just a few of the services it offers:

- Full CMIS capability including an agent object to handle incoming CMIS requests and select the target MO's according to Distinguished Name scoping and filtering constructs
- Upper layer OSI stack that conforms fully to the relevant ISO/CCITT recommendations.
- Extensible full Directory Service (X.500) and File Transfer (FTAM)
- Lightweight presentation layer runs directly on top of TCP
- Generic MIB with a common data dictionary for generic attributes including generic MO with generic methods for selecting objects and accessing attributes
- Generic system management functions including object and state management

The network management platform provides the generic functionality for all managed devices and systems. It also provides a SMASE to handle the protocol for ATN dedicated SM manager to manager communications.

## 2.1 Generic API's

The Generic Managed System (GMS) and the Remote MIB (RMIB) are standard API's that are used by the applications when acting in agent and manager roles respectively. There are also standard API's for coordination, and ASN.1 support. The Directory Support Service (DSS) provides more sophisticated searching, discovery and trading facilities for address resolution and directory access.

### 2.1.1 GMS API

The GMS API provides the full functionality of the OSI management model including scoping, filtering, access control, linked replier and cancel-get. It supports Object Management, Event Reporting, and Log Control System Management Functions (SMF). The GMS is comprised of three classes:

- CMISAgent that provides OSI agent facilities
- Generic MO
- Meta-class MO supporting meta-class information for the MO

Additionally there is generic attribute types such as counter, gauge, counterThreshold, gaugeThreshold and tideMark.

### 2.1.2 RMIB API

The Remote MIB (RMIB) support service offers a higher level API which provides the abstraction of an association object. This handles association establishment and release, hides object identifiers, hides the complexity of CMIS distinguished names and filters, provides a high-level interface to event reporting which hides the manipulation of event discriminators and error handling. There are two API's in the RMIB:

- RMIBAgent which is the association object
- RMIBManager which provides call-backs for asynchronous services offered by the RMIB Agent.

OSIMIS offers a simpler high-level approach that could be used to meet the requirement of the remote MIB as a piece of abstracted "equipment" (per CONOPs). This facility is provided by the Shadow MIB (SMIB) support service, which offers the abstraction of object in local address spaces, 'shadowing' the real managed object handled by remote agents.

### 2.2 Generic Applications

There are two types of generic applications:

- semantic-free manager ones that may operate on any MIB without changes, and
- gateways for other management models (i.e. between CMIS/P and the Internet SNMP)

#### 2.2.1 **Generic Managers**

These are a class which are semantic free and these are usually referred to as MIB browsers as they allow movement in a management information tree, retrieval and alteration of attributes, perform actions and created and delete managed objects. These browsers owe their genericity to the generic CMIS facilities. The facilities being empty local distinguished name for the top MIB object, the localClass facility and scoping.

#### 2.2.2 **Generic Gateways**

Generic gateways between OSI CMIP and the Internet SNMP are possible without any semantic loss for the conversion from CMIS to SNMP as SMNP operations and information model are a pure subset of OSI. This work involves a translator between Internet MIBs to equivalent GDMO ones and a special back-end for the GDMO compiler, which will produce run-time support for the generic gateway. This provides a way to process current or future MIBS without changes to the gateway.

## 3.0 Application Program Interfaces

### 3.1 RMIB Manager

The Manager application provides a set of abstract classes providing virtual methods. Each manager class redefines the virtual methods specifying how it will receive and process the incoming event reports and asynchronous results. The use of virtual object classes addresses the need of the ADL to provide both security and common usability (or interoperability).

The advantages of using a virtual object class are:

- An external object may query a virtual object exactly as it queries any other object. The external object may request that the virtual object return the value of any of its externally visible attributes or it may request that the virtual object execute any of its externally visible functions. The external object issues these requests in the same protocol that it uses on any other object; it does not know whether the queried object is actual or virtual. A virtual object class participates in all the relationships defined for its supporting actual base class.

- Valuable for security. All instances of a virtual class must be instances of its supporting base class. Although the virtual class can have instances, it can not be instantiated. It is not possible to create a new object using the virtual object class as a template. An object can only be created as an instance of an actual object class. Once it is created, it may happen to automatically become a member of one or more virtual object classes supported by that actual class**. It is not possible to destroy an instance of a virtual object class**. If the object is destroyed as an instance of the actual object class it is automatically destroyed as an instance of all virtual object classes supported by that base class. A virtual object can never be modified**: it is not possible to change the value of an attribute of a virtual object**. If a change must be made, it must be made to the supporting actual instance. This use of virtual objects will address the issue of a remote manager from one domain inappropriately altering the data of another domain's agent.

An example follows[1]:

Consider a router-based internetwork designed for interconnecting various local-area networks over several geographical dispersed sites. The routers use a link-state-based algorithm to exchange routing information. Because the internetwork is large, it is configured to be a multi-tier network for the purposes of routing. The network is partitioned into separate areas. Each area connects to the wide area backbone using a special router called the designated router or a level-2 router. Each area may also contain other routers which route traffic within the area, called level-1 routers. Each designated router converses with designated routers in other area to convey information about the network nodes reachable in its own area [ISO 10589]. The designated router for each area is chosen either by assignment or by election among routers. There may be no special built-in differentiation between level-1 routers and designated routers; it is possible that all router devices may be exactly alike. Although a designated router may choose to invoke some built-in FUNCTIONS that level-1 routers may not invoke, it need not intrinsically possess any additional functionality that level-1 routers do not already have. All routers may be instances of the same object class. It is also possible that a designated router may automatically become a level-1 router if a new election is held among the routers

[1] Object-Oriented Networks, Models for Architecture, Operations, and Management by Subodh Bapat

following a topology reconfiguration. For these reasons, it is not good modeling practice to define designated-Router and level-1-router as subclasses of router. If these classes are created and populated with instances, instances of designatedRouter would have to be deleted and new instances would have to be added to level-1-router each time a new election was held. The routers should have the ability to autoclassify themselves. Therefore.. both designatedRouter and level-1-router can be defined as selected virtual classes supported by the base router class. The selector basis of virtualization can use the value of the level attribute of the router class. The selector for designatedRouter constrains this attribute to the value 2; the selector for level-1-router constrains it to the value 1. Each time a new election is held, the routers reclassify themselves if necessary merely by changes the value of their level attribute. When the membership of the designatedRouter class is queried, only those router objects whose level attribute happens to have the value 2 *at that time* are returned. The entire membership of designatedRouter is re-evaluated by applying the selector basis of virtualization to every member of its base router class. Of course, all explicitly assigned designated routers will always be members of the designatedRouter class.

The manager application will include:

- RMIB support service offers high-level CMIS abstraction with idea of association object, which encapsulates the management association and underlying data using object-oriented method. The API sets on top of the MSAP service using the M_GET.

- The association object offers high-level management operations where MO naming and filter expressions are realized through string based notations hiding: CMIS dialogue, protocol access and ASN.1

- Event Reporting interface including the manipulation of the EFD object, filter assertions to manipulate event discriminator constructs, limited error handling

- Management op interface can be synchronous in RPC fashion or asynchronous using call-back functions. In RPC, whole application is blocked until reply or error is received until wait routine times out.

- RMIB Manager has 2 ‘virtual’ methods used for receiving and interpreting asynchronous results.

- Notion of MO’s as abstractions of real resources. Not a strong separation between managing and managed systems so they can play both roles.

### 3.1.1 Manager Virtual Methods

- createResult(CMISobject, RMIBAgent) it receives the create operation result for a single MO

- generalResult(CMISobjectList, RMIB Agent) is used for Get, Set Action and Delete where a number of Mo's can be returned.
- EventNotification method includes a parm(RMIBAgent* agent) so event reports can be separated from one another in an application having more than one management associations
- CMISobject List has a method getOperation which can identify the type of operation invoked and can be used in RMIB Manager call-back.
  GeneralResult(CMISobjectLIST, RMIBAgent) to correlate results against original request.

### 3.1.2 Management Functions

Specify minimal and essential management functions. The types of SM functions that should be specified for the management platform include: functions providing generic definitions of MO's or attributes, attributes or notifications and those that provide system definitions necessary to systems management, event reporting, log control

Function Importance are as follows:

1. Configuration and performance monitoring:
- System functions like event reporting and log control
- Object/state management
- qosAlarm, MetricSummarisation

2. Security
- Access Control
- SecurityAlarm Reporting/ Audit Trail

The two most fundamental functions to provide OSI management power through event reporting are:

- qosAlarm
- securityViolation

#### 3.1.2.1 State Management

Define generic state attributes, model for transitions between states and provide state change notification (state transition diagrams?)

#### 3.1.2.2 Object management

1. Passthrough service to CMISE from SMASE
2. object creation
3. object deletion
4. attribute value change

### 3.1.2.3 Management of Event Reporting

Methods to log, filter, and forward relevant network events in a distributed environment. Description includes control messages, event reporting messages and retrieval messages. The ATN manager application needs the possibility of remote systems to receive event reports from a target systems simultaneously and for the event reporting criteria (thresholds) to be individually tailored. This requirement is in contrast to conventional requirements in which it is only necessary to report events to a single remote link.

Requirements:

- Definition of a flexible event report control service allowing systems to select which event reports are to be sent to particular managing system.
- Specification of the destinations to which event reports are to be sent
- specification of a mechanism to control the forwarding of event reports, by suspending and resuming forwarding (may address by monitoring MO)
- ability of an external managing system to modify the conditions used in the reporting of events
- ability to designate a backup location to which event reports can be sent if the primary location is not available

The event report management model provides for remote event reporting and local processing of potential event reports. The model also describes the control message, event reporting messages and retrieval messages.

Conceptual event processing receives local notifications and forms the potential event reports. These potential event reports are distributed to all Event Forwarding Discriminators that are contained within the local open system. The Event Forwarding Discriminator is used to determine which event reports are to be forwarded to a particular destination during specified time periods. It may also be used to specify the mode (confirmed or non-confirmed) for forwarding events. Each event forwarding discriminator may contain a scheduling capability determining the intervals during which event reports will be selected for forwarding. Each event forwarding discriminator contains a scheduling capability determining the intervals during which event reports will be selected for forwarding. The EFD is itself a MO and can emit notifications. These notifications are processed as potential event reports for all event forwarding discriminator including the one that generated the notifications.

Event reports are generated as a result of the notification that an event has occurred e.g. a threshold violation or a change in configuration status. The event forwarding management function provides the capability of identifying the destinations to which selected event reports are to be sent. Event reporting management provides the means by which discrimination and forwarding can be initiated, terminated, suspended, or resumed and through which the attributes of the EFD can be read and modified.

Event reporting management function provides the capability for setting up a long-term event reporting relationship between two open systems. While the EFD is in the unlocked state, the reporting open system forwards event reports to the specified destination given that the operational state is enabled and any schedule is not off-duty

## 3.2 RMIB Agent

A Remote MIB Agent is an association object representing the management association with a remote agent. The RMIBAgent class when instantiated at runtime become a RMIBAgent object, which can be used to activate a management association. It allows operations to be performed between remote Agent/Manager and use the event reporting facility. One RMIB Agent is used per association. More than 1 may be present in an application and possibly active at one time.

The manager-agent ratio can be many to one. The managers may communicate with same remote agent sequentially. The interface can be synchronous or asynchronous depending on the application. Synchronous block will occur until result or error is received. The ATN Remote Agent application may wish to receive event reports and asynchronous requests. This is achieved using RMIB Manager through callbacks. In the asynchronous case, the coordinator object is needed to provide a central listening facility. A single coordinator is required to receive external events and async results on behalf of the registered Knowledge Source. Coordinator also supports real time alarms for polling.

The RMIBAgent is a specialized Knowledge Source to receive event reports and via the applications coordinator.

With each async request the RMIBManagerobject is passed explicitly in the call to register with the RMIBAgent object. The Agent object is responsible for executing the call-back using the correct RMIBManager. When the event arrives through the coordinator when accessing the remotes MIB and MITree the MO on which the operation is to be performed is specified using the class name and the instance name (or Distinquished Name)

In the RMIB interface, the MO class name and distinquished name are passed in strings and mapped onto their OID and DN equivalent before requesting MSAP interface.

Note: Remote MIB provides CMIS interface for interacting with agents of the remote MIBs

### 3.2.1 Agent Generic Functions

This agent provides generic functions such as object addressing, scoping, filtering and error handling. Also provides generic management functionality:

- event reporting
- event logging

- access control

## 3.3 GMS Application

The Generic Managed System offer tools for building agents that use the full functionality of the OSI management model. SMF's supported include:

- Object Management
- Event Reporting
- Log Control
- QualityofServiceAlarm notification of the Alarm Reporting SMF
- Partial Access Control, Metric and Summarization Objects
- High level API for the integration of new managed object classes where only semantic aspect (behavior) need to be implemented

These functions are realized as special MO's, generic attribute and notification types which can be instantiated or invoked. Notification can be emitted through a special method call and all the subsequent notification processing is carried out by the GMS in a transparent fashion. Object management is accomplished by using GDMO definitions together with the GMS to hide completely the emission of object creation and deletion notification and the attribute change when something is changed through CMIS.

### 3.3.1 CMIS Agent

The CMISAgent is a specialized knowledge source. Functions supported are:

- Manage associations based on authentication information
- Validate operation parameters
- Find the base object for the operation
- Apply scoping and filtering
- Check access control rights
- Apply the operation on the target managed object
- Return the result/error

The specific managed objects make use of a common meta-class for all the managed object classes (MOClassInfo). Access to the meta-class information is checked for correctness and authorization before the behavior code that interacts with the real resource is invoked.

GMS offers support for all CMIS Get requests through the coordination mechanism.

# 4.0 Generic Application Level Gateway

Considerable investments have been made into network solutions based on the two dominant management protocol architectures. It is necessary to provide a NM platform solution that will allow both SNMP and OSI CMIS/P to coexist to achieve global inter-working across heterogeneous platforms. OSMIS provides a 'semantically rich reference model as the basis for this inter-working.'

The gateway offers generic functionality from the OSI world including:

- Event-Reporting SMF- permits the managing application to create Event-forwarding-discriminators at the agent, which control the selection and destination of all event reports that the agent generates.

- Log-Control SMF – which permits event logging according to manager configurable criteria.

- Metric Monitor and Summarization SMFs – permit manager application to configure agents to undertake localized polling, threshold checking, data summarization and statistical analysis.

- X.500 Directory- allows distributed transparencies, such as faults, replication and location transparency to be achieved

## 4.1 Protocol Mapping

There exists a one to many mapping when translating CMIS operations to SNMP. OSIMIS offers a solution that minimizes the number of SNMP requests. Then CMIS requests (M-"Create, M-Delete, M-Set) operations are all mapped to SNMP Set requests.

Note: M-Set requests that would cause the creation or deletion of multi-instance SNMP objects are prevented to ensure that the semantics of the original CMIS requests are not infringed.

SNMP traps are mapped to an 'internetAlarm' CMIS Event-Report. This notification contains the list of name/value pairs that are provided by the Trap's list of variable-bindings. The gateway is also required to determine the Distinguished Name of the object instance that is associated with each variable-binding. The completed event report must then be forwarded to the manager that has request such reports, and may also be locally logged.

OSIMIS has developed an efficient mapping that minimizes the number of generated SNMP requests (since MIB transversal using the Get-Next primitive necessitates a wait state until the response to the current retrieval request is received before a further request can be emitted.) The managed objects that are present in the scoped MIT subtree must be

refreshed in a top-down manner. The filter can then be applied to their state to permit selection of those instances that will have the current CMIS operation applied.

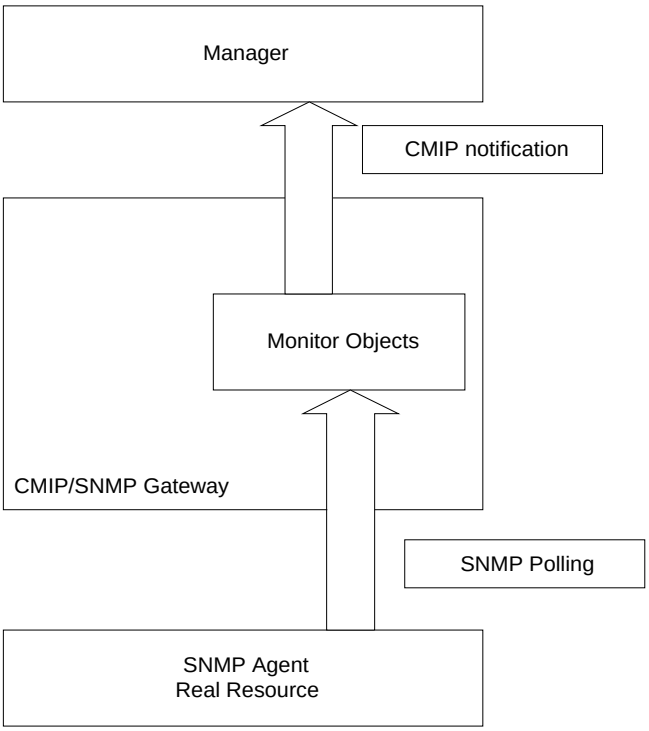
### 4.1.1 Monitor Objects

A monitor object can perform activities on data, contained in attributes of other MO's

1. data capture
2. data conversion
3. data enhancement
4. data analysis
5. generate notifications

This object observes an attribute in an observed MO. This attribute must be of type Counter or Gauge. The info is stored in Monitor Object and converted to rateGauge. Thresholds may be set to determine a qosAlarm (iso10164-4) notification. The remote application can set/change threshold boundaries used in Monitor objects, on/off at run time. [ISO 10164-11].

Monitor objects can be used to eliminate some amount of polling traffic in the network. Without monitor objects a manager must poll the SNMP agent to get info from the real resource. If you use Monitor objects all polling is isolated to agents local network as the Monitor objects can take care of polling and only emit notifications to Manager when appropriate. This helps eliminate unnecessary traffic on the Manager's network




## 5.0 Generic MIB

Management of real resources takes place through instances of managed object classes. The collection of managed objects related to the management of an open systems in known as a managed system. Managed objects are organized in a Management Information Tree (MIT) according to containment relationships. This is necessary for naming purposes so managing applications can access them. Containment relationships between classes are specified by Name Bindings.

### 5.1 Event MO's (Generic definition)

The super class is a discriminator object class. The discriminator is a managed object that allows a managing system to control management operations and event reports that may be forwarded, by a managed system. Discriminators can be created, deleted, read and modified. The discriminator provides the specification of conditions that shall be satisfied prior to allowing the management operation or notification associated with the discriminator to proceed, these conditions are:

- identification of scheduling packages that determine when discriminator processing will occur
- the criteria for discrimination
- the administrative and operation state of the discriminator
- specific subclass conditions

Each discriminator has an operational state and an administrative state. Operational states in 10165-2…Operational state changes reported using the state change notification. The notification will be process by the affected discriminator before it enters the disable state, or after it enters the enabled state. The operational state are enable and disabled. The administrative states are locked and unlocked. State changes are the result of intervention by a managing system or local administrative activity. Notifications result from change in the administrative state or creation/deletion. The discriminator contains a construct that is a filtering mechanism on inputs. Discriminator attributes are:

| | |
|---|---|
| Discriminator Id | uniquely identifies the instance of a discriminator |
| Discriminator Construct | specifies tests on information to be processed by the discriminator |
| Administrative State | unlocked- processing is permitted by a managing system<br>locked - processing is not permitted by a managing system. |
| Operational State | enabled - operable<br>disabled - inoperable |
| Notifications | state change<br>attribute value change<br>object creation<br>object deletion |
| Scheduling Packages | Conditional packages related to scheduling |

| | |
|---|---|
| | are defined for the event forwarding discriminator |
| Availability status | A conditional package present if the other scheduling related packages are instantiated. |

Conditional package we should consider is the duration package. This package allow the user to automatically control the time that a managed object starts and stops functioning through the use of the start time and stop time attributes. Could be useful to transmit 'logged' events and data at non-peak times. (10164-5)

### 5.1.1 Event Forwarding Discriminator

Allows specification of conditions to be satisfied by potential event reports related to MO before the event report is forwarded to a particular destination(s). It is a subclass of discriminator.

| | |
|---|---|
| Destination | destinations to which the discriminator forwards event reports. May be a single /multiple application entity title |
| Backup destination Package | Two attributes which contain back and active destinations. |
| Backup destination list | An ordered list of application entity titles. The application entities identified in the list are AE titles designated to be used if the destination specified by the destination attribute fails Detection of AE failures is a local matter |
| Active destination | a single application entity title. |
| Mode package | Attribute may be set by the managing system at object creation time. It may be confirmed or unconfirmed. |
| EFD behavior | In addition to the superclass behavior new behavior testing attributes |

### 5.1.2 Event Logs

The log records events chronologically in a readable format. The platform writes information to the log about any known network events and can generate its own events.

### 5.1.3 Event Report Control

Report control and thresholds as MO's in their own right so they can be created and deleted dynamically and many to many relationship could exist between defined events, threshold and report control MO's

Need the possibility of remote systems to receive event reports from a target systems simultaneously and for the event reporting criteria (thresholds) to be individually tailored.

This requirement is in contrast to the conventional one in which it is only necessary to report events to a single remote link.

### 5.1.4 Generic Notifications

#### 5.1.4.1 Event Type

Communications alarm type. Associated with procedures/processes used to convey information from one point to another. ( Classify protocol errors here?)
QOS alarm type. Associated with degradation in the quality of service.
Processing error alarm type. Associated with a software of process fault
Equipment alarm type. Associated with an equipment fault.
Environmental alarm type. Associated with a condition relating to an enclosure which the equipment resides.

#### 5.1.4.2 Event information

1. Probable cause. Probable cause values for notifications shall be indicated in the behavior clause of the object class definition.

Do we what to use the one defined in the standard??? Probably just a subset.

2. Perceived severity. Defines how the capability of the MO has been affected.

Cleared: Indicates the clearing of one or more previously reported alarms. It will clear all alarms for a managed object that have the same Alarm type, probable cause. Multiple associated notification may be cleared by using the Correlated notifications parameter. (The clearing of alarms does not need to be reported???)
Indeterminate: Level of severity can't be determined
Critical: Indicates a services affecting condition has occurred and an immediate corrective action is required. (ex MO is totally out of service and must be restored)
Major: Indicates a service affecting condition has developed and an urgent corrective action is required.
Minor: Indicates the existence of a non-service affecting fault condition and that corrective action should be taken in order to prevent a more serious fault.
Warning: Indicates the detection of a potential or impending service affecting fault, before effects have been felt. Action should be taken to further diagnose and correct the problems to prevent it from becoming a service affecting fault

3. Back-up status
4. Specifies whether the object emitting the alarm has been backed-up, and services provided to the user have not been disrupted. This field with the severity field can diagnose the ability of the system to continue providing services.
5. Back-up object. Specifics the instance that is providing back-up services. This parameter is related to the back-up object relationship attribute (10164-3)
6. Trend indication (is it needed)
7. Threshold Information. Present when the alarm is the result of crossing a threshold.

Triggered threshold. The identifier of the threshold attribute that caused the notification
Threshold level. In the case of a gauge the threshold level specified a pair of threshold values, the first being the value of the crossed threshold and it's corresponding hysteresis (In case of a counter the threshold specifies on the threshold value)

Observed value: The value of the gauge or the counter which crossed the threshold. This may be different from the threshold value.
Arm time: For a gauge threshold, the time at which the threshold was last re-armed, namely the time after the previous threshold crossing at which the hystersis value of the threshold was exceeded again permitting generation of notifications when the threshold is crossed. For a counter threshold, the later of the time at which the threshold offset was last applied, or the time at which the counter was last initialized.

8. Notification identifier. Provides an identifier for the notification, which may be carried in the correlated notifications parameter of future notification. Notification identifiers must be unique across all notifications of a particular MO throughout the time that the correlation is significant
9. Correlated notifications: Contains a set of Notification identifiers and their associated MO instance name. (Do we need a source object instance??? Would correlation be needed from a MO instance other than the one in which the Correlated notification parm appears???
10. State change definition: Used to indicate a state transition (10164-2) associated with the alarm
11. Monitored attributes:
12. Proposed repair actions

## 5.2 Generic MO

Define the generic MO. This class includes methods for selection of objects and access to attributes.

### 5.2.1 Use of packages

The functionality of MO classes come through grouping of attributes, actions, notifications an behavior known as packages. When referring to the functionality of a class you are referring to the functionality of its basic package. A class has 1 basic package and 0 or more conditional ones.

Conditional packages offer capability to tailor functionality of an object instance at run time, by allowing the manager to control the existence of conditional packages when creating that instance

The importance of packages is the ability of run-time tailoring as opposed to compile time supported by inheritance.

This is important for aircraft situations where the additional functionality may incur a space or processing overhead for managed devices and should be used only when needed. In which case the manager may delete and re-create an instance with more packages that will supply information and behavior it needs. Object deletion and recreation is needed to modify the configuration of packages.

Note: Mo's will need to be classified as safety or not??? Or just a queriable attribute?? The need is to be able to retrieve logged information for safety related stuff. Do I need a separate ASE and protocol for safety related SM Function (because of potentially degraded system capabilities?? Or just a separate agent??

### 5.3 MIB Browser

The MIB Browser is a management function that allow the MIB to be read and used. A generic browser is needed. A generic browser would need no prior knowledge of MO classes present in the target managed system, so it can cope with the evolution of the MIBS. The generic MIB browser needs the following to be defined: mapping between classes, attribute and event identifiers and their name and syntax as defined in OSI standards. (Could use global OSI directory as a repository when information can not be locally found)

Containment relationships between classes can be done by name bindings. Each object class has a naming attribute in a name binding with a containing class which together with the valued uniquely identifies it. (This is known as a Relative Distinguished Name) The sequence of the RDN from the top of the containment structure to its MO constitute its DN

## 6.0 Summary

OSIMIS takes advantage of the OSI Management Model that hides the complexity of the underlying communications protocol (CMIS/P). It provides a method to condense the associated information model into simple to use Application Program Interfaces (APIs). It combines the 'thoroughness of the OSI models and protocols with advanced distributed systems concept pioneered by ODP to provide a highly dynamic distributed information store….it also combines seamlessly the OSI management power with the large installed based of Internet SNMP capable network elements'.

This paper recommend ATNP adopt a similar method of solving the unique set of problems the ATN faces in network and systems management. ATNP could adopt the idea of a generic management systems platform. This paper proposes the initial specification include initial components of the platform, such as:

- a generic 'summary' MIB with a common data dictionary
- an agent to create and update the MIB (similar to the GMS)
- a Remote Manager to access the MIB