Intelligent Remote Monitoring

George Pavlou, Kevin McCarthy - University College London, UK
George Mykoniatis, Jorge Sanchez - National Technical University of Athens, Greece

Abstract. Intelligent monitoring facilities are of paramount importance in both service and network management as they provide the capability to monitor quality of service and utilisation parameters and notify degradations so that corrective action can be taken. By intelligent, we refer to the capability of performing the monitoring tasks in a way that has the smallest possible impact on the managed network, facilitates the observation and summarisation of information according to a number of criteria and, in its most advanced form, permits the specification of these criteria dynamically to suit the particular policy in hand. The OSI management metric monitoring and summarisation management functions provide models that only partially satisfy the above requirements. This paper describes our extensions to the proposed models to support further capabilities, with the intention to eventually lead to fully dynamically defined monitoring policies. The concept of distributing intelligence is also discussed, including the consideration of security issues and the applicability of the model in CORBA-based environments.

Keywords: Management, Monitoring, OSI, TMN, GDMO, CORBA

1. Introduction

Managing Quality of Service (QoS) is of paramount importance in both current and future service architectures operating over underlying broadband technologies (SDH, ATM etc.). Engineering considerations such as minimising the impact of management systems on the managed network and reacting in a timely fashion to performance degradations need to be supported by relevant computational and information models. In the TMN framework [M3010], OSI Management principles [X701] are used to structure and access management information in an object-oriented fashion. The need for generic information models supporting intelligent monitoring activities was recognised long ago, resulting in the metric monitoring [X738] and summarisation specifications [X739].

These move essential intelligence close to the managed resources, reducing the amount of management traffic and providing support for a sophisticated event-driven operation paradigm. Though of undoubted usefulness, it soon becomes apparent that the intelligent monitoring policies they support are predefined. A slightly different policy than the ones supported needs to be specified in object-oriented terms, realised and then fielded in systems supporting such facilities as pre-compiled logic i.e. it is necessary to go through the full development cycle. Ideally, fully flexible generic facilities are necessary, where new policies can be realised dynamically, without the need to alter anything at the managed end of the spectrum.

Having realised these limitations, in the context of the RACE-II ICM project we have carried out research leading in the provision of such advanced facilities. This research culminated in the specification of the Generic Support Monitoring Objects (GSMO) [Sanchez] and their object-oriented realisation as part of the OSIMIS TMN platform

[Pav95a]. In this paper, we explain the relevant issues and operational models, present two examples to show the applicability of these concepts and discuss issues related to dynamic intelligence, security and future mappings on CORBA.

2. The OSI Management and Directory Model

The Telecommunications Management Network (TMN) [M3010] provides the framework for managing heterogeneous networks and services. It proposes a hierarchical layered architecture with functional blocks communicating management information across reference points. These become interfaces when functional blocks are mapped onto distinct physical entities that exist at different network nodes.

The TMN reference points and interfaces are based on OSI application layer services. They are of a transaction-oriented nature, involving both protocols as well as associated information models. OSI Management [X701] is the main base technology, supported by the OSI Directory [X500] in order to support dynamic addressing, global naming and location and other transparencies. File Transfer Access and Manipulation (FTAM) may be also used to support bulk data transfer e.g. for software management in the context of service deployment. Finally, Transaction Processing (TP) is necessary to guarantee the consistency of complex distributed management transactions.

OSI Management projects a fully object-oriented model, with applications in agent roles "exporting" managed objects that encapsulate managed resources while applications in manager roles access these objects in order to implement management policies. Managed objects are formally specified in GDMO [X722], which is a formal object-oriented specification language with emphasis in management. The associated service/protocol (CMIS/P) [X710] has essentially "remote method call" semantics. The separation between manager and agent roles serves the purpose of the model and it is not strong in engineering terms: applications and even objects may be in both roles and this is in fact the norm in a hierarchical layered architecture such as the TMN.

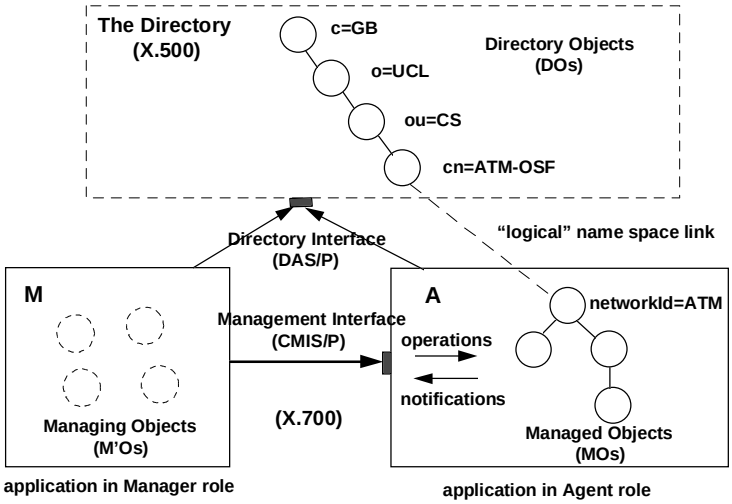

Fig. 1. The combined OSI Management and Directory model

A key difference between OSI management with other ODP-based object-oriented frameworks such as CORBA [CORBA] is that objects always come in “clusters” or “ensembles”. Objects in these have a lot of relationships but containment is treated as a primary relationship that yields unique hierarchical names. The cluster is also the unit of global distribution but managed objects and other functional components may also be distributed locally. An object-oriented database capability is offered through facilities known as scoping and filtering. Many objects may be selected through scoping while the selection may be further controlled through filtering, which allows complex assertions on attribute values. This facility may be exercised across the management interface using CMIS but also through object attributes with scope and filter syntax and semantics. In the latter case, their evaluation depends on the managed object behaviour. This capability is used by the intelligent monitoring objects.

Global distribution is based on the OSI Directory which provides a hierarchical distributed object-oriented database. Applications notify the directory of where they are and of their capabilities so that location and other transparencies can be realised. This also allows managed objects to be addressed through global names, which contain information of the logical “cluster” they belong to e.g. a particular TMN Operations System Function (OSF). OSI Management is an object-oriented technology in terms of information specification and access: objects are visible “on the wire” but the internal structure of relevant applications is not dictated and may not even not object-oriented. Platform infrastructures such as OSIMIS [Pav95a] have shown how such an object-oriented specification may be mapped onto a fully object-oriented realisation, providing high-level APIs and various transparencies [Pav94]. The Network Management Forum is currently looking to standardise such object-oriented APIs. The combined OSI Management and Directory model is shown in Figure 1.

3. Monitor Metric and Summarisation Facilities

While the manager-agent model does not in itself restrict the amount of intelligence that may be specified and realised by managed objects, most standard specifications concentrate in providing a rich enough set of attributes and actions which model information and possible interactions with the underlying resource. Notifications are also provided to report significant exceptions but they are usually generic ones e.g. object creation / deletion and attribute value change.

The OSI Structure and Definition of Management Information (SMI / DMI) specifies generic attribute types such as counter, gauge, threshold and tide-mark. Gauges model entities with associated semantics e.g. number of calls, users, quality of service etc. or the rate of change of associated counters e.g. bytes per second etc. Thresholds and tide-marks may be applied to gauges and generate QoS alarms and also attribute value changes, indicating change of the high or low “water mark”. Such activities are of a *managing* nature, in fact in the SNMP paradigm they are solely performed by management stations.

Although thresholding functions could be made part of managed objects modelling real resources (in fact, there have been such early object specifications), it does not

take long to recognise their genericity. As such, they should be better provided elsewhere so that they become re-usable. The relevant ISO/ITU-T group recognised the importance of generic monitoring facilities and standardised the metric monitor [X738] and summarisation[X739] systems management functions. By making such functions generic, it is possible to implement them once and make them part of the associated platform infrastructure.

The whole idea behind monitor metric objects is to provide thresholding facilities in a *generic* fashion. Monitor metric objects may be instantiated within an application in an agent role and be configured to monitor, at periodic intervals, an attribute of another real resource managed object. The underlying resource may be a physical or logical aspect of a network or service element. It may also be a composite one, realised through an Information Conversion Function (ICF) by accessing subordinate information models. The observed attribute should be a counter or gauge and the metric object either observes it as is or converts the observed values to a rate (derived gauge) over time. Statistical smoothing of the observed values is also possible if desired.

The main importance of this facility is the attachment of gauge thresholds and tidemarks to the resulting derived gauge which may generate quality of service alarms and indicate the high and/or low "water mark", as desired by systems using this function. In fact, the metric objects essentially enhance the "raw" information model of the observed object. The metric monitor functionality can be summarised as:

- *data capture:* through observation or "scanning" of a managed object attribute
- *data conversion*: potential conversion of a counter or gauge to a derived gauge
- *data enhancement:* potential statistical smoothing of the derived result
- *notification generation:* QoS alarm and attribute value change notifications

The metric monitoring model is shown in Figure 2.

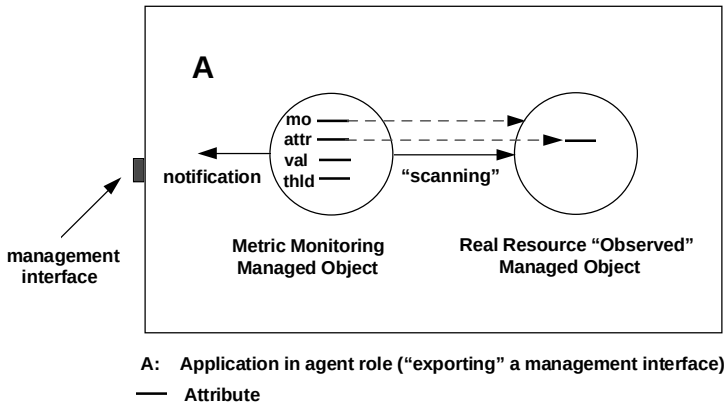

Fig. 2. The metric monitoring model

Gauge thresholds are always specified in pairs of values: a triggering and a cancelling threshold. The former will generate a notification when crossed only if the latter has

been previously crossed in the opposite direction. This prevents the continuous generation of notifications when the measured value oscillates around the triggering threshold and is known as the *hysteresis* mechanism. Figure 3 shows both high (e.g. "over-utilisation") and low thresholds (e.g. "under-utilisation") applied to the observed value. Typically, a normalised value of around 5% is used as the hysteresis interval.

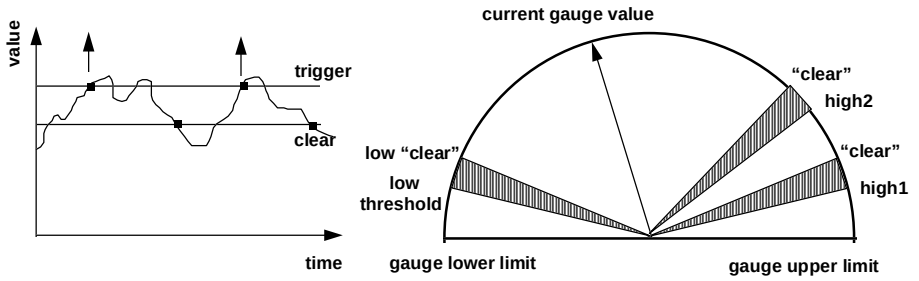

Fig. 3. Thresholding with hysteresis

The metric objects offer the OSI management power through event reporting and logging, even if the "raw" observed management information model does not support such notifications. More importantly, they obviate the use of rates, thresholds and tidemarks in a way tied to specific managed objects but they allow the same flexibility and power dynamically, whenever a managing system needs it. Such a monitoring facility reduces the management traffic between applications and their impact on the managed network by supporting an event-based operation paradigm. It should be emphasised that these facilities are of "managing" nature but offered within a managed object cluster across a management interface.

The summarisation objects extend the idea of monitoring a single attribute to monitoring many attributes across a number of selected managed objects. They offer similar but complementary facilities to metric objects. In this case, there are no comparisons / thresholding but only the potential statistical smoothing and simple algorithmic results of the observed values (min, max, mean and variance). These are reported periodically to the interested managing systems through emitted notifications. The observed managed objects and attributes can be specified either by supplying explicitly their names or through CMIS scoping and filtering. The observed values may be raw ones, modelling an underlying resource, or enhanced values as observed by metric objects. They can be reported either at every observation period or after a number of observation periods (buffered scanning).

The major importance of this facility is that a number of values a managing system needs can be specified once and then reported as mentioned, without the managing system having to send complex CMIS queries periodically or having to perform the statistical smoothing etc. Intelligence in this context has to do with the periodic scanning and reporting of diverse information automatically, after the criteria for its assembly have been specified once. Such criteria may be expressed through CMIS scoping and filtering, providing a lot of flexibility in summarising information of dynamic

nature (e.g. traffic across certain ATM virtual path connections etc.) Any other complex computations on the summarised information are left to be performed by the managing system. Such logic is usually of static, pre-compiled nature.

4. Advanced Intelligent Monitoring Facilities

Rationale

While the metric monitor and summarisation facilities can be combined to provide a lot of flexibility, they still leave the task of complex calculations and comparisons to managing systems. In particular, in most cases it is not one attribute value that needs to be monitored and compared to thresholds but the combination of more than one attributes, possibly across different managed objects. The derived value is thus the result of the application of a *mathematical operation* on the observed values. Such an operation could be the *sum(a_1+...+a_n)*, *div(a/b)*, *diff(a-b)* etc. The combination of more than one operations may be used to construct arbitrary expressions, albeit at the cost of multiple intelligent monitoring objects.

In short, a facility is needed that combines the properties of the metric monitor (thresholding on a derived result) and summarisation objects (observation of arbitrary objects and attributes). This combination is effected through a mathematical operation that is performed on the observed values to yield the derived result. We have recognised the need for such a facility early on and have specified and implemented a number of GDMO classes that provide this functionality, termed collectively *Generic Support Monitoring Objects* [Sanchez]. Since then, the relevant ITU-T standardisation committee has also recognised this need and provided amendments to the metric and summarisation functions offering similar functionality.

In most cases, simple mathematical calculations are enough to express real-world problems. For example, if connection acceptance and rejection counts are kept, the connection rejection ratio is given by the simple formula below. In a similar fashion, if PDU rejected and sent counts are kept, the error rate across that transmission path is given by their ratio. Such arithmetic operations can be either specified statically, through a relevant attribute of the summarisation object, or specified dynamically by combining the observed attribute values with operands. In the latter case, ultimate flexibility is provided in applying arbitrary mathematical operations.

```
 Connection Rejection Ratio          Transmission error rate
        Sum(connRej)                        Sum(pdusRej)
---------------------------                 ------------
Sum(connAcc) + Sum(connRej)                 Sum(pdusSent)
```

Model

The underlying concepts for the Generic Monitoring Support function are based on the combination of the Metric Monitoring and Summarisation functions. This function provides for the ability to aggregate attribute values, apply operations on them and provide statistical information about the aggregated result of the operation. The function is realised by the Generic Support Monitoring Objects (GSMO) and provides for:

• the ability to monitor information provided by single or multiple attribute types

• the selection of the observed attribute values either explicitly, through the names of the containing objects, or through scoping and filtering

• the identification of an operation (e.g. sum) to be applied on the observed attribute values

• the identification of an algorithm (e.g. uniform weighted moving average) used to smooth the derived result

• the ability to emit notifications when the derived result crosses a threshold or "pushes" a tide-mark or when any of its attributes change

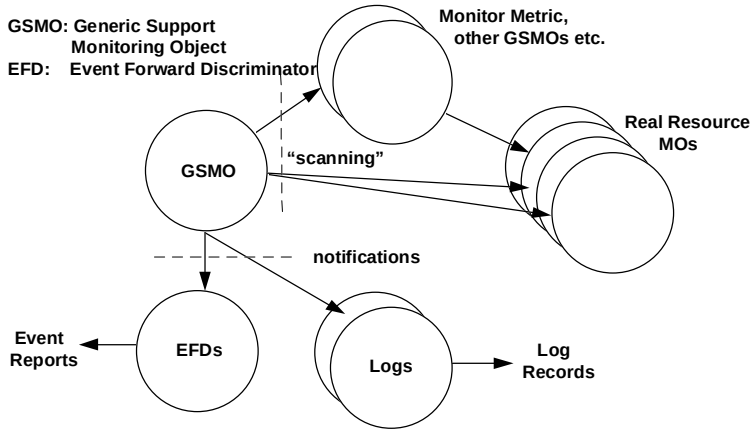

Fig. 4. Intelligent Monitoring and Summarisation model

The model for the operation of the GSMO objects is similar to that of summarisation and is shown in Figure 4. Information is obtained by observing attributes of other objects, including managed objects representing a management view of an underlying network or service resource, metric objects, or other GSMO objects. The attributes of those "observable" objects are accessed at the object boundary, triggering associated behaviour in the same fashion as across a management interface.

A particularly useful type of intelligent monitoring object is the probability estimator. This is a buffered homogeneous object that will scan one observed attribute and will calculate the probability that its value is greater than a preset threshold:

Prob[A>T] = Number of appearances of the event {A>T} / Number of observations

For example, calculating the probability that the connection rejection rate is greater than T involves two intelligent monitoring objects: one calculating the connection rejection ratio as *connRej / (connAcc + ConnRej)* and a probability estimator observing the output of the first one according to the above formula.

While there is a lot of flexibility in specifying attributes and objects to be observed, either explicitly or through scoping and filtering, the full set of available mathematical operations is statically specified and realised when the GSMO information objects become engineering computational objects using relevant platform infrastructure. The

introduction of a new mathematical operation implies new specification and subsequent realisation. An additional drawback relates to the fact that any more complex mathematical operation needs to be decomposed to simple generic ones that are supported, necessitating the existence of more than one such objects. This can pose a scalability problem regarding activities at entities with 10's of thousands managed objects (e.g. a mediation function for an access network).

The above observation has led us to research the possibility for specifying dynamically the operation to be applied on the observed attributes. This is possible by merging the relevant operations and operands with the observed attributes in a GSMO attribute with a complex, recursive, tree-like syntax. ASN.1 is very powerful and particularly suitable for such complex representations. The notion of "global attributes" is also introduced as a tuple that consists of a global object name, extending across both the directory and management name spaces, and an attribute type.

Through this type of object, it is possible to specify dynamically any arbitrary mathematical operation as dictated by the management policy, without the need for any additional specification or implementation. The drawback is that the observed "global attributes" need to be explicitly specified as they are referenced in the mathematical operation i.e. the flexibility of using scoping and filtering is lost. This can be compensated by using these in conjunction to the previous types of GSMO objects.

Since managed object names can be global, all the types of GSMO objects can be used for the intelligent summarisation of information which exists logically or physically at different clusters and locations. In this case, the observed objects and attributes need to be explicitly specified. The advantage of monitoring within the same node and reducing management traffic is lost and this defies somehow the purpose of GSMO.

What emerges as a possibility though is the inclusion of further intelligence so that a top-level operation is broken down to sub-parts based on observed values within the same node or domain. Each such sub-part will be delegated to a subordinate GSMO object which will be created in that domain. All the buffered scanning and smoothing will take place locally while resulting values will be reported back to the "master" GSMO object to produce the final value, perform comparisons and trigger notifications. The result of using such a facility is effectively "downloading" intelligence as close to the managed elements as possible, reducing the overhead of management traffic and increasing the observed information timeliness by reducing network latency. The mapping of managed object names to nodes or domains could be through the directory. Further research is necessary to realise such intelligent monitoring objects.

Security considerations

An important aspect of any distributed management environment is security. In the OSI management model, security services comprise authentication, stream / connectionless integrity and access control. The first two rely on information exchanged at association establishment and as such are orthogonal to management information. Access control enables authorised initiators i.e. applications in managing roles or human users driving such applications to have different views of the "exported" man-

agement information by the target application. The level of enforced access control could be at the class / instance level or at the individual attribute, action and notification level [X741].

Access control is enforced by special support objects which identify authorised initiators and targets i.e. information to be accessed and the rules of the access control policy. Typically, protected managed objects (targets) are unaware of the access control policy they are subject to. The generic part of the agent application authorises or refuses requests based on the access control objects and the relevant policies. The behavioural logic of managed objects (including the access control ones) is totally unaware of the access control function. This is particularly important, since it allows the access control and management policies to be kept separate.

Unfortunately, this does not hold for intelligent monitoring objects - their sophisticated functionality comes at a price. Intelligent monitoring objects act essentially as managing objects within an application in agent role, accessing other real resource objects at the object boundary. According to the standard access control model, unauthorised initiators may be prevented from creating such objects. If though creation rights are granted, this essentially means that the initiator in question has read access rights to the whole object cluster. This is because intelligent monitoring objects contain attributes whose values point to other objects and attributes and the latter can *not* be checked by the access control function in a generic fashion.

Since it is very common for applications in managing roles to have only partial access to information across a management interface, it is essential to allow for such access control policies in the context of intelligent monitoring. The solution is to make the relevant monitoring objects aware of the access control policy and check that the latter is not violated. This check is performed when they are created or whenever the monitored targets change e.g. through a set operation. The drawback of this approach is that the access control infrastructure becomes non-transparent, though its visibility can be eliminated through object-oriented realisation. This is considered a small price to pay given the rich functionality such intelligent objects provide. In addition, due to their genericity, they are implemented only once and become subsequently part of the relevant platform infrastructure, so the “problem” is eliminated.

Realisation

Implementing OSI management based information specifications can be a daunting proposition without suitable supporting infrastructure. In our case we have implemented standard metric monitoring, summarisation and intelligent monitoring (GSMO) objects using the OSIMIS object-oriented TMN platform [Pav95a]. In OSIMIS, GDMO information objects become C++ engineering objects. A GDMO/ASN.1 compiler was used to produce stub managed object infrastructure, which hides completely all access details, leaving only behavioural aspects to be implemented.

An important platform aspect for realising such objects is the ability to evaluate scoping and filtering locally, to address managed objects by name and to access their attributes at the object boundary. OSIMIS is designed with flexibility in mind and

these functions are supported through easy to use APIs. Finally, accessing objects can be through the same API regardless of location, offering useful transparency services. The resultant implementation has become an integral part of the OSIMIS platform. The monitoring objects exist as pre-compiled knowledge at every application in agent role, ready to be instantiated by managing systems to offer their services.

5. Applicability Examples

Let's examine now the applicability of the previously described intelligent monitoring facilities through two representative examples.

FDDI Network Utilisation

As the first example, we will consider a performance monitoring case study for a FDDI Metropolitan Area Network (MAN). The management policy in this case is to continuously monitor the utilisation of the network and generate both early warnings and "red light" quality of service alarms, indicating levels of congestion. In addition, we would like to keep the history of network utilisation over periods of time and, in particular, the highest congestion levels reached over those observation periods.

A FDDI MAN has ring topology and packets are "injected" in the ring at a particular source node, to be subsequently forwarded across the ring by other nodes until they reach the specified destination node. Usually FDDI rings are used as backbones, carrying traffic of other Local Area Networks (LANs) that are attached to them. At every node, the number of octets (bytes) received and sent through the two FDDI interfaces of that node are known. The network utilisation can be thus calculated by averaging the throughput in terms of outgoing traffic through every node on the ring.

Most FDDI rings come with SNMP-based management interfaces at every node. The latter support attributes related to the activity of every physical interface of that node. This can be considered as "raw" information available to managing systems for data conversion, enhancement, thresholding, history etc. In SNMP-based management systems, all these functions take place in centralised management stations which retrieve this information periodically across the network. This mode of operation is in accordance with the SNMP *fundamental axiom* of physically separating managing and managed functionality in order keep network elements simple.

Given the fact that SNMP agents do no support sophisticated intelligent monitoring facilities, there is not much that can be done to "push" intelligence to the network element level. This intelligence has to be deployed instead at the next level of the management hierarchy, which has to be kept as close to the managed elements as possible. In our case study, this is the adaptation / mediation layer which is conceptually part of the element management layer of a TMN hierarchy. We thus operate a generic CMIS/P to SNMP application gateway or Q-Adaptor Function (QAF) [McCar], which provides automatically an OSI view of SNMP managed objects. In engineering terms, one such device will adapt for all the FDDI nodes, being both a TMN Q-Adaptor (QA) and a Mediation Device (MD). This should operate at one of the FDDI nodes, keeping all the monitoring traffic in the FDDI domain.

Averaging the throughput across the FDDI ring is a function that can be easily provided by the intelligent monitoring support objects. An instance of such an object should be created within the CMIS/P-SNMP QA/MD application with the task to monitor locally the FDDI interface objects, calculate the mean, convert it to a throughput gauge, smooth it statistically and compare it to threshold values specifying early warning and red-light (congestion) conditions. In addition, the highest and lowest throughput levels will be recorded through relevant tide-mark attributes. Note that monitoring locally within the QA/MD will translate to monitoring over the FDDI ring across the SNMP interface of the nodes. Despite this, the polling traffic will be limited in the FDDI domain while the result of the intelligent monitoring object will be presented to higher-level functions in an event-driven fashion, through emitted QoS alarm and attribute value change notifications. Event forwarding discriminator and possibly log objects will have to be created to disseminate notifications or log them locally.

The attributes of the monitoring object set by the managing object or application to support this function are:

```
scannerId           “fddi-thput”
granularityPeriod   10 (secs)
algorithmType       EWMA (exponentially weighted moving avg)
estimateOfMeanThld { Low=0.8 Switch=Off High=0.85 Switch=On ,
                     Low=0.9 Switch=Off High=0.95 Switch=On }
operation           mean     (of observed values)
baseManagedObject   {}       (the top MIT object)
scope               3ndLevel (interface entries are there)
filter              (objClass=interfaceEntry & ifType=fddi)
attributeId         ifOutOctets
```

Note that the interface objects containing the attributes to be monitored are specified using the scoping and filtering facilities. That way, exact knowledge of the FDDI ring topology in terms of its number of nodes is not necessary. An alternative, less flexible way to specify those objects could be through an instance list. Note also that the observed value is statistically smoothed after the calculation in order to reduce problems with rapid fluctuations. Finally, the threshold values are shown normalised: 85% and 95% are the early warning and red-light values respectively, with 5% hysteresis each. In reality, these would have to be specified based on the units of the measured traffic (octets) and the maximum speed of the network (150 Mbits / sec).

The class of monitoring object used above supported a number of operations to be applied on the observed values, from which the *mean* was used. As we already described, the limitation of this approach lies in the fact that the number of supported operations are finite and pre-defined. Our case study was served well by the existing operations, but it would be nicer to be able to supply a coefficient to be applied to the mean so that threshold values can be normalised. Bearing in mind we are observing bytes / sec, the coefficient for FDDI should be $(8 / 150 * 10^6)$. For example, if our FDDI has four nodes, with b the byte counter at each node and c the coefficient, the observed value is given by the expression:

e.g. b_1 = { {elemId=fddi-1, interfacesId=null, ifEntryId=2} ifOutOctets }

In this case, the monitoring support object specifies through one attribute both the observed object instance / contained attribute and the operation to be applied on the observed values. Note that the absence of scoping and filtering implies knowledge of the interface object name. This can be obtained through a get operation with scope and filter, before the monitoring object is created.

Summarising, the functionality described for the FDDI case study is a matter of deciding on the management policy and instantiating the type of intelligent monitoring object suitable for the policy in hand. Semantic-free (generic) applications such as browsers, event monitors, loggers etc. may be used to receive the QoS alarms or to retrieve and plot history data and trends. In short, it may be a matter of minutes (or hours at most) to implement sophisticated monitoring policies, as opposed to the weeks (or months) that would be required otherwise for the full development cycle

ATM Network Monitoring

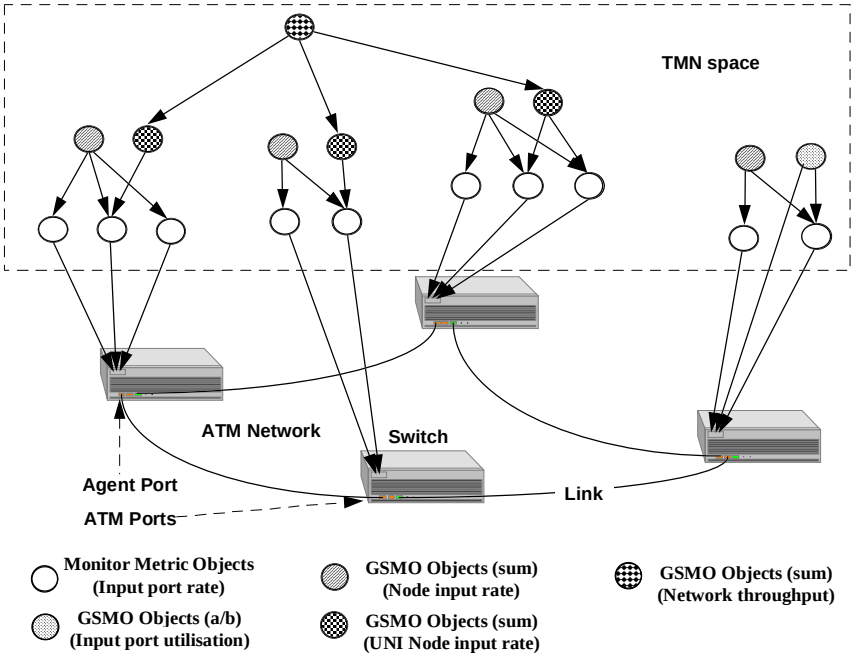

Fig. 5. ATM intelligent monitoring

As a second example, we will consider monitoring the activity of a high-speed network of ATM technology. In this example we will examine how a “raw” attribute (numberOfCellsReceived in a ATM port) that is available at switches with management

interfaces can be used to compute the following:

- link / port throughput (rate) and utilisation
- node input rates, including the User-to-Network (UNI) and Network-to-Network (NNI) input rate
- network throughput

As a first step, monitor metric objects should be instantiated at each switch agent (NEF) for the computation of the link or port throughput (numberOfCells over time). As the link/port throughput and capacity are known (the latter is an attribute of the link object), it is possible to calculate the link utilisation through an intelligent monitoring object with a division operation (link rate / link capacity). The overall node input rate can be calculated by another intelligent monitoring object, observing all the link input rates and adding them up (sum operation). Finally, the network throughput can be calculated by a top-level monitoring object, observing all the intelligent monitoring objects that calculate UNI link input rates and adding them up. The relevant hierarchy of monitoring objects on a 3-node network is shown in Figure 5.

Instantiating all these intelligent monitoring objects can be done in a way that does not need pre-defined knowledge of the network topology but "discovers" it from information registered in the directory. Instantiating the intelligent monitoring objects can be done, for example, using an interpreted access API such as the Tcl-MCMIS offered by the OSIMIS platform. A small script e.g. 50-100 lines of interpreted object-oriented logic is enough to start-up the above monitoring function, supported subsequently by generic tools such as browsers, event monitors etc.

6. Discussion

The intelligent monitoring facilities presented try in essence to move intelligence to the agent part of the manager-agent model, reducing network traffic and allowing managing entities to focus in the realisation of management policies, being supported by a rich event-based paradigm. Their generic nature makes it possible to implement them once and make them part of relevant platform infrastructure. Their use can enhance "raw" information models and pays real dividends when used in conjunction with simple information models resulting from SNMP to GDMO translation. Arbitrary expressions combining attribute values are possible, while the use of scoping and filtering provides greater flexibility in specifying information with dynamic properties.

All the above facilities, from the metric monitor objects to the fully dynamic distributable intelligent summarisers, try in essence to break away from the static, compiled knowledge of monitoring intelligence. In fact, they are steps towards fully dynamic intelligence that could be downloaded to the managed resources and operate autonomously as much as possible, reporting back to a master managing object which will have a global view of the domain in which it operates the policy. Such intelligence may have intrusive (control) as well as monitoring aspects in order to react to pre-defined conditions that express the management policy in hand.

There is a lot of ongoing research towards defining formally arbitrary policy objects. These will contain (part of) the management policy in the form of an attribute containing a script in an interpreted policy language which may access other local or remote objects in a transparent fashion, very much like the intelligent monitors. Temporal constraints should be possible as well as facilities to structure the policy as in programming languages (control loops, procedures etc.) We are actually considering such implementations based on the TCL interpreted language and its object-oriented extensions, while the active policy objects are specified in GDMO [Vassila]. There are a number of problems to be solved though, the main one being concerned with preventing the arbitrary consumption of computing resources that may lead the managed system containing the misbehaved policy object to starvation.

All this work has been based on the OSI management framework. Given the advent of ODP-based approaches, an interesting consideration is to investigate the mapping of those concepts onto OMG CORBA, regarding the latter as the pragmatic counterpart of ODP [X901]. The intelligent monitoring objects are specified in GDMO/ASN.1 from an information perspective and realised as C++ engineering objects, using suitable flexible infrastructure provided by the OSIMIS platform. In theory, it should be possible to map them onto CORBA IDL from a computational perspective and realise them as C++ engineering computational objects on a CORBA platform.

Though this is in theory possible, first considerations reveal a number of hurdles to overcome. First, there is no notion of clusters of objects in CORBA as in OSI agents or access to them with database-like facilities such as scoping and filtering. References to objects are possible but management traffic will be generated, defying partly the purpose. Another more important limitation is that the notion of unique global names does not (yet) exist in CORBA. As such, it is not possible to specify intelligent monitors that span a particular domain administered by one ORB. Finally, it is not easy to map onto CORBA IDL recursive ASN.1 definitions like the one used by intelligent monitors to specify arbitrary operations on observed attributes.

These considerations relate to the current state of CORBA. Trading services may be used in the future to emulate scoping and filtering while the issue of global object references is going to be addressed. The problem of mapping complex recursive ASN.1 syntaxes to equivalent IDL ones remains but there exist workarounds through non-generic mappings. We will report our findings in the future.

7. Conclusions

We explained in this paper the issues behind intelligent remote monitoring and the use of the OSI management access and information models to provide flexible generic support object specifications, the Generic Support Monitoring Objects (GSMO). We also showed the applicability of those objects in the TMN context through two simple but representative case studies. Given the future integration of service execution and management infrastructures in a unifying framework [Pav95b], it is of paramount importance that support for such powerful concepts is maintained.

Acknowledgements

This paper describes work undertaken in the context of the RACE II Integrated Communications Management project (R2059), which is partially funded by the CEU.

References

[M3010] ITU-T M.3010, Principles for a Telecommunications Management Network

[X701] ITU-T X.701, Information Technology - Open Systems Interconnection - Systems Management Overview

[X500] ITU-T X.500, Information Processing - Open Systems Interconnection - The Directory: Overview of Concepts, Models and Service, 1988

[X901] ITU-T X.900, Information Processing - Open Distributed Processing - Basic Reference Model of ODP - Part 1: Overview and guide to use

[X710] ITU-T X.710 / X.711, Information Technology - Open Systems Interconnection - Common Management Information Service/Protocol, Version 2

[X722] ITU-T X.722, Information Technology - Structure of Management Information - Part 4: Guidelines for the Definition of Managed Objects, 1991

[X738] ITU-T X.738, Information Technology - Open Systems Interconnection - Systems Management - Part 11: Metric Objects and Attributes, 1994

[X739] ITU-T X.739, Information Technology - Open Systems Interconnection - Systems Management - Part 13: Summarisation Function, 1994

[X741] ITU-T X.741, Information Technology - Open Systems Interconnection - Systems Management - Part 9: Objects for Access Control, 1995

[CORBA] Object Management Group, The Common Object Request Broker Architecture and Specification (CORBA), 1991

[Sanchez] Sanchez, J., G. Mykoniatis, J. Reilly, G. Pavlou, K. McCarthy, Intelligent Monitoring Functions in OSIMIS, ICM/WP3/NTUA/0097, 1994

[Pav94] Pavlou, G., T. Tin, A. Carr, High-level APIs in the OSIMIS TMN Platform: Harnessing and Hiding, in Towards a Pan-European Telecommunication Service Infrastructure - IS&N ‘94, pp. 219-230, Springer-Verlag ‘94

[Pav95a] Pavlou, G., K. McCarthy, S. Bhatti, G. Knight, The OSIMIS Platform: Making OSI Management Simple, in Integrated Network Management IV, pp. 480-493, Chapman & Hall, 1995

[McCar] McCarthy, K., G. Pavlou, S. Bhatti, N. DeSouza, Exploiting the Power of OSI Management in the Control of SNMP-capable resources, in Integrated Network Management IV, pp. 440-453, Chapman & Hall, 1995

[Vassila] Vassila, A., G. Knight, Introducing Active Managed Objects for Effective and Autonomous Management in the TMN, this volume

[Pav95b] Pavlou, G., D. Griffin, Issues in the Integration of IN and TMN, this vol.