# A Modeling Toolset for the Analysis and Design of OSI Network Management Objects

To deal with the complexity of network management standards and the increasing demand to deploy network management applications quickly, analysts and designers need a set of tools to help them quickly and easily model, define, and develop new network management objects.

**by Jacqueline A. Bray**

The HP GDMO (Guidelines for the Definition of Managed Objects) Modeling Toolset (GMT) is the first tool in the HP OpenView TMN (Telecommunications Management Network) developer tool chain (Fig. 1). This toolset consists of a set of integrated tools designed to aid developers in the analysis and design of OSI network object models. The key components of the toolset are a graphical modeling tool, import and export facilities, and a conformance report generator. The tools operate independently of other HP OpenView products so that network specifiers can work independently of implementers. This article provides an overview of the network modeling process, GDMO, and the modeling tools.

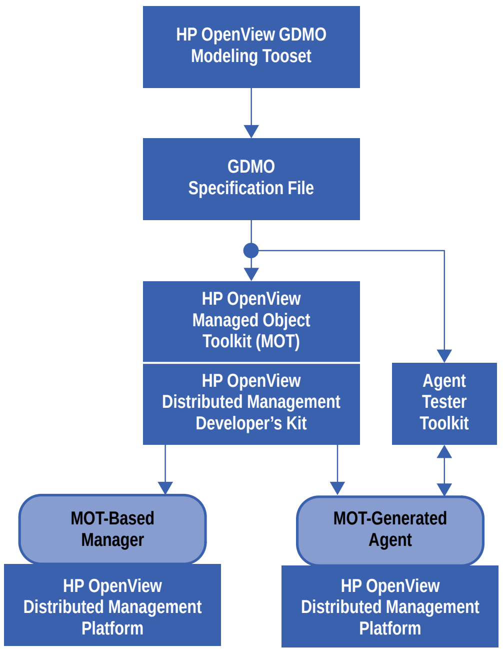

HP OpenView GDMO Modeling Tooset
GDMO Specification File
HP OpenView Managed Object Toolkit (MOT)
HP OpenView Distributed Management Developer's Kit
Agent Tester Toolkit
MOT-Based Manager
HP OpenView Distributed Management Platform
MOT-Generated Agent
HP OpenView Distributed Management Platform

***Fig. 1.*** *HP OpenView TMN developer tool chain.*

## The Modeling Process

The first step in developing a network object model is the analysis of the environment to be managed. The network and system resources to be managed are identified and their characteristics and the operations that can be performed upon them are defined. The managed resources might be physical (e.g., a router or workstation) or logical (e.g., a software process). A managed resource might also represent a collection of different resources. Other elements might be managed that are not actually resources but are required to support management functions, such as an event log. These requirements are translated into a GDMO object model, with the managed resources represented as managed objects. The managed objects define the interface to a managed resource.

## GDMO

The Guidelines for the Definition of Managed Objects is an ISO standard (ISO/IEC 10165-4 (ITU X.722))[1] that defines how network objects and their behavior are to be specified, including the syntax and semantics. This specification language allows network object designers and manager/agent implementers to communicate designs and build upon existing designs. GDMO is an object-oriented environment, using the concepts of inheritance, containment, and encapsulation. It is used to define:

- Managed object classes for managed resources
- Attributes and behaviors of a managed object
- Operations that can be performed on an attribute or object
- Notifications (events) an object might issue
- Relationships with other managed objects
- The names of object instances.

GDMO is organized into templates, which are standard formats used in the definition of a particular aspect of the object, with rules for how these templates refer to each other. A complete object definition is a combination of interrelated templates. There are nine of these templates.

- *Managed object class templates* define a model for managed object instances that share the same characteristics. The inheritance relationships with other managed object classes are specified, along with the packages that define the class characteristics.
- *Package templates* are groups of logically related sets of behaviors, attributes, attribute groups, actions, notifications, and parameters. With each attribute is a property list of valid operations (Get, Replace, Add, and Remove), initial values, and other value characteristics.
- *Behavior templates* describe, in textual form, the behavior of a component.
- *Attribute templates* define an actual data element of an object, including its syntax and behavior.
- *Attribute group templates* define a set of attributes to allow operations to be performed on the group as a whole.
- *Action templates* define additional operations for a managed object that cannot be modeled using the standard operations defined in the package template.
- *Notification templates* define unsolicited events that may be sent by the agent.
- *Parameter templates* define error conditions specific to the object and extend the definition of information used by actions and notifications. The context within which this parameter can be used is specified.
- *Name binding templates* define where an object may be located in the global containment tree, along with the attribute used to distinguish object instances. These templates also specify rules for the creation and deletion of the object instances.

An example of each of these templates can be found in ***Appendix A***, which shows a portion of a GDMO definition for a UNIX® password file (i.e., /etc/passwd). The details of the information to be exchanged between the manager and agent are defined using ASN.1 (Abstract Syntax Notation One).[2] ASN.1 is a formal description language used to define data types to be exchanged between systems. It includes primitive data types, such as integer and Boolean, and allows new data types to be constructed from these types. The data types are grouped into one or more ASN.1 modules within a GDMO definition. In the example in Appendix A, there is one ASN.1 module named PasswordFileInfo. The GDMO templates reference ASN.1 data types by prefixing the data type with the ASN.1 module name (e.g., PasswordFileInfo.LoginNameSyntax in the loginName attribute template).

Other ISO standards also relate to the definition of management object models. For example, The Management Information Model[3] is a companion document that defines modeling concepts, principles of naming and relationships, and scoping and filtering. Another example is the standard for the Definition of Management Information.[4] This standard defines, in GDMO and ASN.1, a set of managed object classes to be used as superclasses. It includes an object class named *top*, from which every other managed object class ultimately derives. The other classes form an inheritance hierarchy, with top as the root. The object class top includes attributes for object instance naming, which the other classes inherit. The set of rules for defining managed objects is referred to as the *Structure of Management Information*.

## The Toolset

The GDMO modeling toolset stores the GDMO and ASN.1 definitions in an object dictionary, which acts as a central repository for all the tools (see Fig. 2). The toolset allows concurrent access to the tools and object dictionary and can be configured as a client/server architecture. GDMO and ASN.1 definitions are organized within documents. An import facility allows external standard and user-defined GDMO document files, such as ITU-T X.721, to be loaded into the object dictionary. (X.721 and other GDMO standards are included with the toolset.) New object classes can inherit from any object class in the object dictionary and reference other templates and data types for consistency and reuse of specifications. Within the toolset, each document has a short alias name to simplify references to documents. Object definitions can be added to new or existing documents.

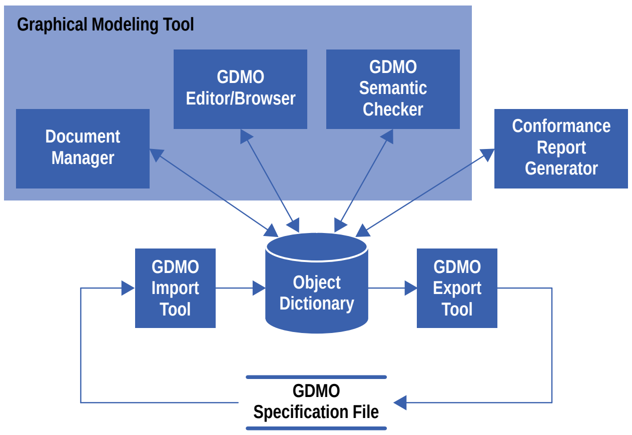
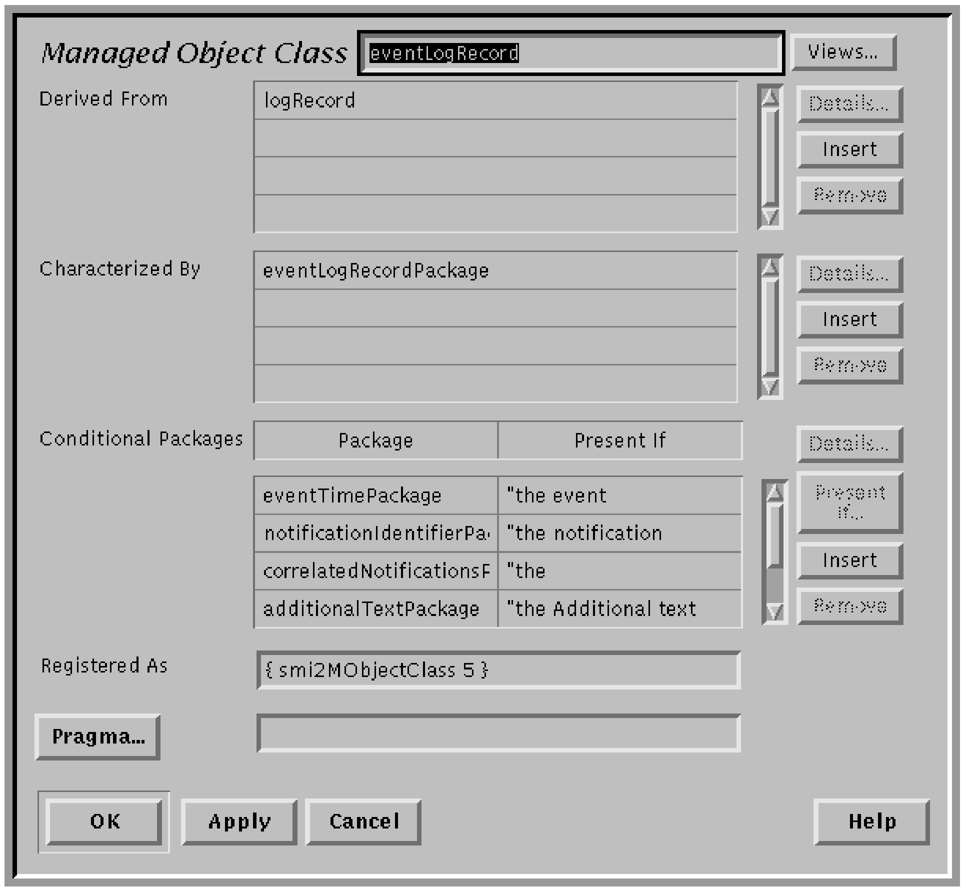


***Fig. 2.*** *The GDMO modeling toolset.*

The graphical modeling tool can be used to learn the syntax of the GDMO language, to explore existing GDMO documents, and to create new ones. A template window exists for each of the nine GDMO template types. These windows show the details of an existing template or guide users in creating new templates. For example, Fig. 3 shows the template for a managed object class, with the GDMO keywords along the left (requiring no entry) and the specific entries for this managed object class entered in the table. Clicking the name of one of the referenced templates, such as the Characterized By Package template, and clicking on the Details button, displays the window for that particular template. The Package template window displays entries for several template types, including Attributes. Clicking on Details for an attribute in that window would display its Attribute template. In this way, successively more detailed information can be viewed until the lowest level, the ASN.1 definition, is reached.



Managed Object Class: eventLogRecord

Derived From: logRecord

Characterized By: eventLogRecordPackage

Conditional Packages:

| Package | Present If |
|---|---|
| eventTimePackage | "the event |
| notificationIdentifierPa | "the notification |
| correlatedNotificationsF | "the |
| additionalTextPackage | "the Additional text |

Registered As: { smi2MObjectClass 5 }

***Fig. 3.*** *The window for a Managed Object Class template.*

Browsers for each of the template types can be invoked to display a list of all available entries of that type. A filter function is available in the browsers to display subsets of the complete list. Template entry names can be copied into another template window without retyping the name by selecting an entry with the mouse and then clicking the Insert button in the appropriate template. The Details button described above also functions in the same way while in the browser windows. This detailed drill-down capability is available throughout the tool.

Two useful features for cross-reference checking the object model are the Viewpoint and Inherited Characteristics options. Clicking the Viewpoint button, available on all template and Viewpoint windows, displays the viewpoint for a selected item. In the Viewpoint of Managed Object Class window in Fig. 4, the selected object is represented by the vertical bar, the templates on the left reference the selected object, and the templates on the right are referenced by the selected object. The boxes on either side of the vertical bar contain the appropriate GDMO keywords. Above each template name is the template type (e.g., MOC (managed object class), PKG (package), etc.). The Viewpoint window is helpful when making changes to an object to verify that those changes will not adversely affect other objects that are dependent upon it.

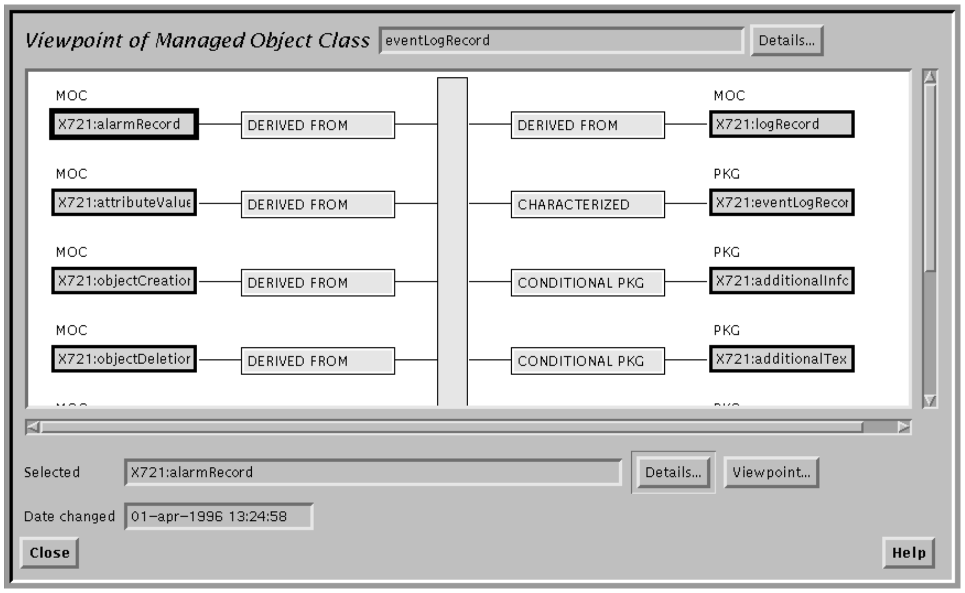


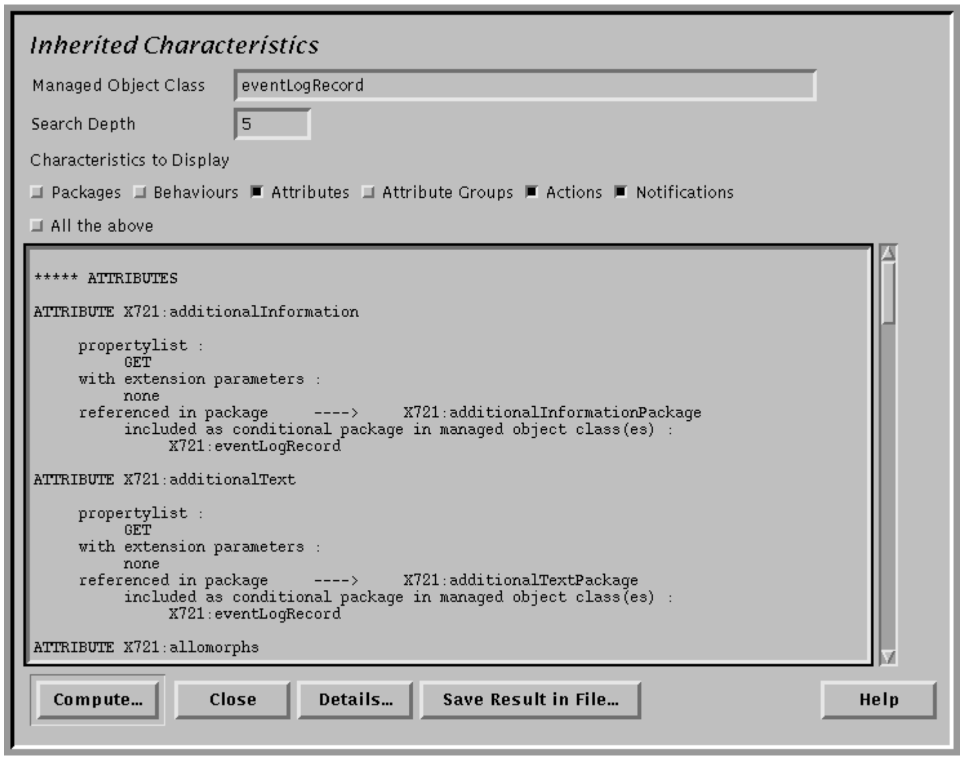
***Fig. 4.*** *A GDMO tool for cross-reference checking: the Viewpoint of Managed Object Class window.*

Clicking Views and then the Inherited Characteristics button, available only on managed object class template windows, displays the window shown in Fig. 5. The characteristics available to a managed object class, whether specialized in that object class definition or inherited from an ancestor, can be displayed by selecting the characteristics of interest (Attributes, Notifications, etc.) from the row of buttons under Characteristics to Display and then clicking Compute. The scrolled window lists all of the inherited characteristics available and where they were referenced. Clicking on a characteristic and then the Details button displays its template.

The graphs available in the GDMO toolset represent three distinct and independent tree structures used in OSI system management. These graphs are the inheritance graph, the registration graph, and the name binding graph. The inheritance graph (Fig. 6) shows the inheritance hierarchy of all the managed object classes in a selected GDMO document, along with any superclasses derived from other documents. (All of the tool windows handle referencing across documents. When a template is referenced from another document, the template is prefaced with the document alias.) Object class nodes can also be added or deleted on this graph. GDMO and the GDMO toolset both support multiple inheritance, which allows classes to inherit properties from more than one superclass.

The registration graph (Fig. 7) shows part of the registration tree of object identifiers defined in ITU-T Recommendation X.721.[4] An object identifier is a unique ASN.1 data type that is a sequence of nonnegative integers representing a particular object. GDMO describes the registration tree structure adopted in the OSI system management standards for allocating globally unique identifiers to components of managed object definitions.[5] Objects can be registered via the registration browser or the registration tree. Registration is typically done in the last phase of GDMO modeling, when document definitions are stable.

The name binding graph (Fig. 8) displays the containment relationships defined via the name binding template. The name binding template specifies a subordinate (contained) object and a superior (containing) object, along with an attribute of the subordinate object that will be used to name instances of that class. The name binding template also specifies whether object instances can be created and deleted via remote management, along with any limitations on those actions. For example, it may specify that an object instance can be deleted via remote management, but only if that object instance does not contain other objects. This containment hierarchy represents the structure of the Management Information Base (MIB). It shows the objects an agent contains and the hierarchy and containment of those objects, which are used not only to define the MIB structure but also as a means of unambiguously referencing object instances.[6]



Inherited Characteristics

Managed Object Class: eventLogRecord

Search Depth: 5

Characteristics to Display

☐ Packages ☐ Behaviours ■ Attributes ☐ Attribute Groups ■ Actions ■ Notifications

☐ All the above

```
***** ATTRIBUTES

ATTRIBUTE X721:additionalInformation

     propertylist :
          GET
     with extension parameters :
          none
     referenced in package     ---->     X721:additionalInformationPackage
          included as conditional package in managed object class(es) :
               X721:eventLogRecord

ATTRIBUTE X721:additionalText

     propertylist :
          GET
     with extension parameters :
          none
     referenced in package     ---->     X721:additionalTextPackage
          included as conditional package in managed object class(es) :
               X721:eventLogRecord

ATTRIBUTE X721:allomorphs
```

***Fig. 5.*** *A GDMO tool for cross-reference checking: the* Inherited Characteristics *window.*

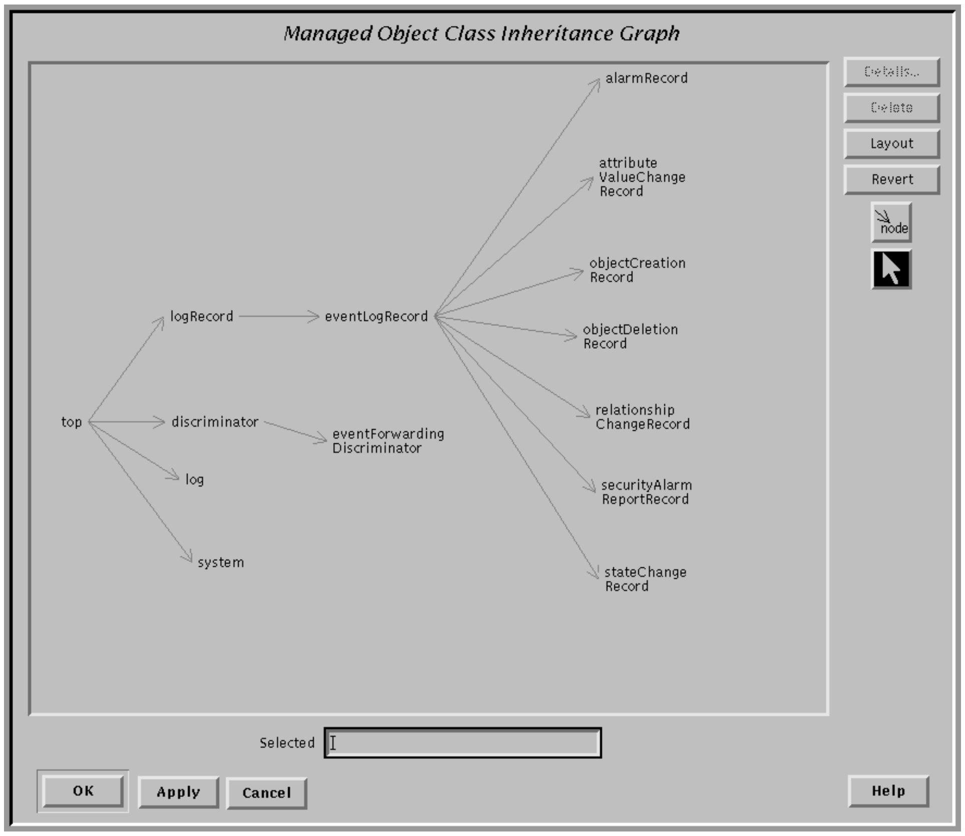


***Fig. 6.*** *A managed object tree structure in a* Managed Object Class Inheritance Graph *window.*

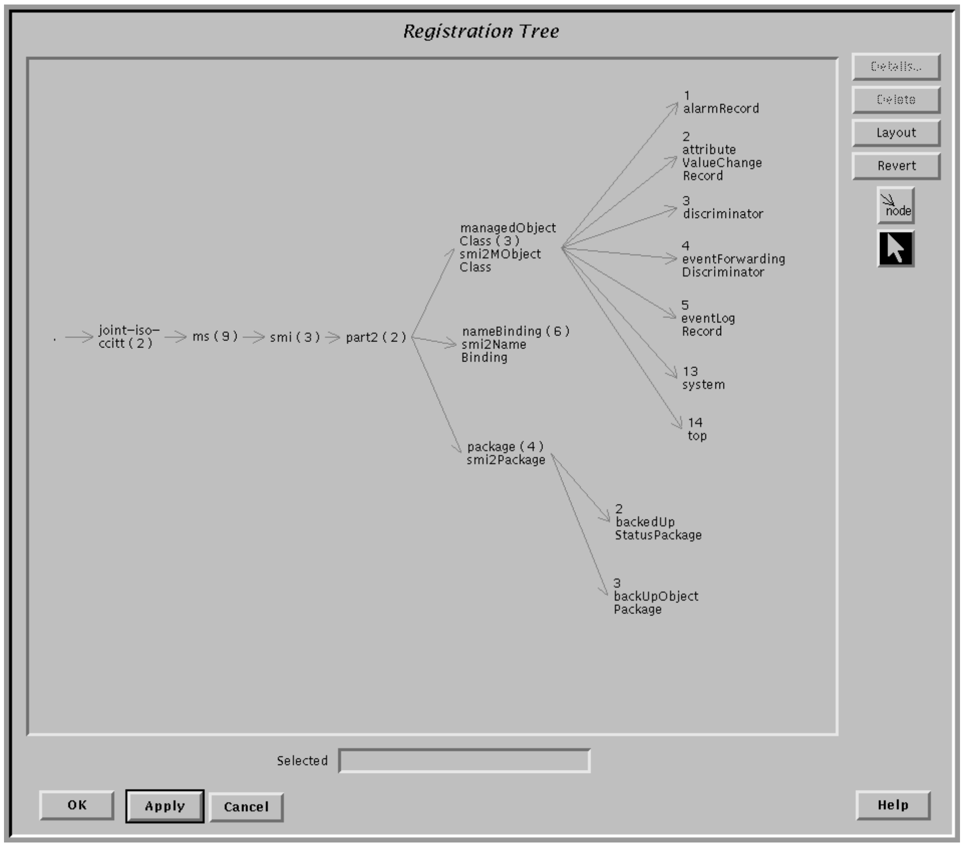


*Fig. 7.* *A registration tree graph.*

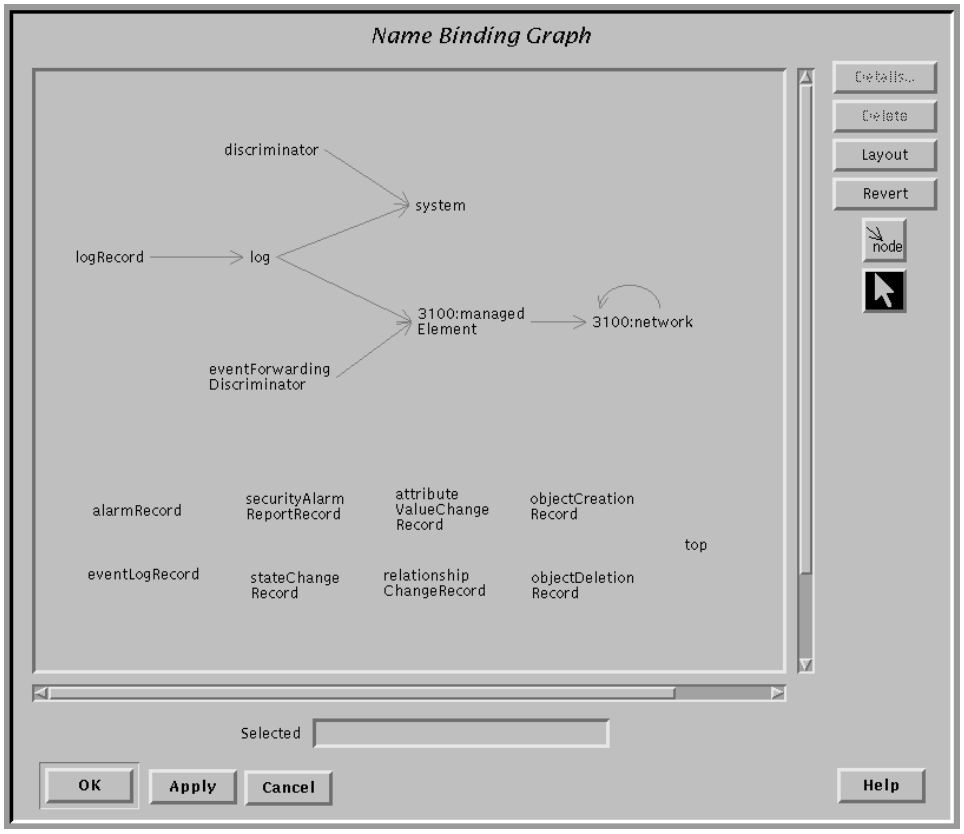
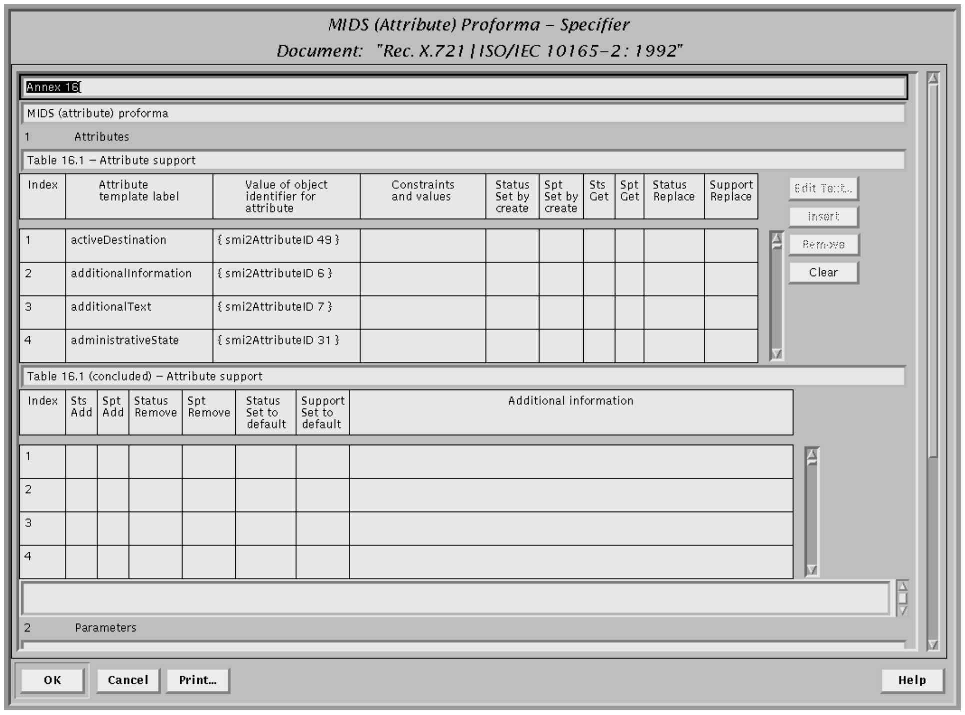


*Fig. 8.* *A name binding graph.*

Once the GDMO document is complete, the semantic checker verifies that the specifications are complete and correct. It checks references throughout the object dictionary, including the detection of templates that are defined but unreferenced. The document can then be exported to an ASCII file for subsequent use by code generators, such as the HP OpenView Managed Object Toolkit and other tools. The ASCII file contains both the GDMO definitions and ASN.1 modules. The Managed Object Toolkit is described in ***Article 6***.

The remaining tool, the conformance report generator, generates printed reports that conform to the ISO standard: *Requirements and Guidelines for Implementation Conformance Statement Proformas Associated with Management Information* (ISO/IEC 10165-6 (ITU–T X.724). Fig. 9 is an example of a proforma. The designer annotates the components in the report with any additional information that will be needed by the agent developer implementing the components. The agent writer will indicate in the proforma report the level of conformance to the definitions.



MIDS (Attribute) Proforma – Specifier

Document: "Rec. X.721 | ISO/IEC 10165-2: 1992"

Annex 16

MIDS (attribute) proforma

1 Attributes

Table 16.1 – Attribute support

| Index | Attribute template label | Value of object identifier for attribute | Constraints and values | Status Set by create | Spt Set by create | Sts Get | Spt Get | Status Replace | Support Replace |
|---|---|---|---|---|---|---|---|---|---|
| 1 | activeDestination | { smi2AttributeID 49 } | | | | | | | |
| 2 | additionalInformation | { smi2AttributeID 6 } | | | | | | | |
| 3 | additionalText | { smi2AttributeID 7 } | | | | | | | |
| 4 | administrativeState | { smi2AttributeID 31 } | | | | | | | |

Table 16.1 (concluded) – Attribute support

| Index | Sts Add | Spt Add | Status Remove | Spt Remove | Status Set to default | Support Set to default | Additional information |
|---|---|---|---|---|---|---|---|
| 1 | | | | | | | |
| 2 | | | | | | | |
| 3 | | | | | | | |
| 4 | | | | | | | |

2 Parameters

***Fig. 9.*** *An example of a proforma, which is the output from the GDMO developer toolkit's conformance report generator.*

## Conclusion

Developing a network object model and the associated GDMO specifications is a complex process. The GDMO Modeling Toolset aims to reduce the complexity and time involved in this definition by providing intuitive graphical tools and object model verification.

## Acknowledgments

The author would like to acknowledge the contributions of many individuals who participated in the development and deployment of the GDMO toolset, including Paul Stoecker and Mark Smith for their technical contributions.

## References

1. *Guidelines for the Definition of Managed Objects*, ITU-T Recommendation X.722, 1992.
2. *Specifications of Abstract Syntax Notation One (ASN.1)*, ITU-T Recommendation X.208, 1993.
3. *Management Information Model*, ITU-T Recommendation X.720, 1993.
4. *Definition of Management Information*, ITU-T Recommendation X.721, 1992.
5. J. Westgate, *Technical Guide for OSI Management*, NCC Blackwell Limited, 1992.
6. W. Stallings, *SNMP, SNMPv2, and CMIP, The Practical Guide to Network-Management Standards*, Addison-Wesley Publishing Co, 1993.

▶ Go to Appendix A
▶ Go to Article 6
▶ Go to Table of Contents
▶ Go to HP Journal Home Page