INTEGRATING CORBA AND JAVA FOR ATM CONNECTION MANAGEMENT

GRAHAM CHEN, JEREMY RIXON AND QINZHENG KONG

***Abstract*—The world-wide telecommunication deregulation and competition increases the demand for a better integrated service management environment which can integrate different management technology to manage different switched and transmission networks.**
CORBA as a general purpose distributed object management technology is gaining acceptance in the telecommunication industry. It has demonstrated its ability to support some important functions required in telecommunications network and service management.
Java has also demonstrated its potential as a platform independent GUI environment for telecommunication network and service management. It can be used as a fast GUI development tool and its platform independent nature and easy integration with CORBA environment offers an ideal environment for deploy customer services.
This paper discusses how CORBA and Java technology is used in developing a prototype ATM management system. This system has been demonstrated at the 1996 Network Management Forum general meeting in Barcelona. The emphasis of the demonstration was on the use of features and services provided by CORBA and the GUI implemented using Java. A simple video-on-demand service is built on top of the management platform. This paper reports our experience with building this prototype.

***Keywords*—ATM, Network Management, Service Management, CORBA, Java**

***Source of Publication*—Proceedings of the 8th International Workshop on Distributed Systems: Operation and Management (DSOM 1997), Sydney, Australia, Oct. 1997, ISBN 1 86365 344 9.**

1 INTRODUCTION

CORBA [12] is a distributed object management technology. It provides a basic framework for distributed computing. Many common object services are defined in CORBA, which provide general purpose distributed services required by a distributed platform and applications. Although CORBA is regarded as an important distributed technology, it still took a long time for the computer and telecommunication industry to accept CORBA as a major alternative development environment, especially in the network and service management area. Compared with existing network management technologies, such as CMIP and SNMP, CORBA has its own characteristics. Its distributed features, rich object services, and the ability to define complex objects make CORBA more attractive for high abstract level object management, such as telecommunication service management [1, 15].

However, although CORBA provides most distributed features required for the development of distributed applications, it does not provide a development environment normally required by most application development. Java [7, 8, 9], on the other hand, provides some of this support. Its platform independent feature has attracted more attention in the computer and telecommunication network and service management industry. The fast growing acceptance of Java and its GUI development environment, and its integration with many underlying management platforms, often make Java the first choice of the GUI based management application development languages.

In [3], a telecommunication service management platform based on the integration of the following three environments is proposed:

- Java-based customer management and service delivery environment
- CORBA-based service management environment
- CMIP, SNMP-based network and element management environment

It also discussed the importance of the seamless integration among these technology to provide a solution for service provisioning and management.

In [10], a discussion was presented on how the integration between CORBA and OSI technology can be achieved to bridge the gap between TMN service and network management. In [4], a demonstration based on this integration was presented to validate the integration strategy.

This paper discusses how CORBA and Java can be used to add value to this integrated service management platform. To support the integrated architecture presented in [3], a prototype ATM management system is built and reported here. The purpose of the prototype is to show how the features and object services supported by CORBA can be used to develop integrated network and service management applications. In particular, the research examines how CORBA can add value to the management solutions in the area of distribution, scalability and integration with the traditional network management technology. It also demonstrates the use of Java in management application development. A demonstration of this system has been given at the Network Management Forum (NMF) in Barcelona in October 1996.

The demonstration allows users to view, navigate and query the current topology of a simulated ATM network as well as provision Permanent Virtual Circuits (PVCs) between hosts in the network. The demonstration is concerned with network topology management rather than network element management. To further demonstrate the service integration, a Video-on-demand (VOD) service is supported so users can order videos delivered over the ATM network PVCs. The demonstration was built as an application of HP OpenView CORBA platform and used its services.

The rest of this paper is organised as follows: Section 2 briefly review the ATM and its management technology, in particular, its connection management. Section 3 discusses how CORBA can be used as the management architecture for ATM connection management. Section 4 and 5 discusses the demonstration including its management architecture, implementation and scenarios demonstrated using VOD service. Section 6 discusses the issues and future work related to build an end-to-end broadband connection management transparent to the underlying transmission technology and the extension of this to a more general service management platform.

2 ATM NETWORK AND ITS MANAGEMENT

ATM (Asynchronous Transfer Mode [14]) is a broadband transmission technology, which takes the advantage of the reliability and fidelity of modern digital facilities to provide fast packet switching.

ATM is a packet-oriented transfer mode. It allows multiple logical connections to be multiplexed over a single physical interface.The information flow on each logical connection is organised into fixed-size packets (cells). The logical connections in ATM is referred to as Virtual Channels. A Virtual Channel is set up between two end users through the network, and a variable-rate, full-duplex flow of fixed-size cells is exchanged over the connection. Virtual Channels are also used for user-network exchange (control signalling) and network-network exchange (network management and routing). A Virtual Path is a bundle of Virtual Channels that have the same endpoints. Thus all of the cells flowing over all of the Virtual Channels in a single Virtual Path are switched together.

ATM management aims to manage the physical ATM network, and to manage the services provided over the logical connections of the ATM network. Examples of ATM management functionality include setting Permanent Virtual Circuits (PVCs) over the ATM network, to view the topology of the complete or part of the ATM network and to support services over the ATM network. These features are supported by the demonstrator.

3 CORBA AS THE MANAGEMENT PLATFORM

The OMG *Common Object Request Broker Architecture* (CORBA) and its supported object services provide a solid base for the shift of network management platform from peer-to-peer communication paradigm to a distributed object paradigm. It also provides the basic technology for developing distributed service management platforms and applications.

ORB, the kernel part of the CORBA architecture, provides the communication channel between Clients and Object Implementations. Different from ordinary message passing channels, ORB provides many distributed features, such as transparency of object implementation.

CORBA object interfaces are defined in IDL. IDL is an abstract definition language for modelling general objects that provides templates for object class definitions. The General and Internet inter-ORB Protocols (GIOPs and IIOPs) define the data transfer syntax and message formats for object invocations within and across ORB domains.

A CORBA object has the following features:

- A CORBA object has a set of attributes, and is the target of operations.
- Unlike OSI managed objects, CORBA does not specify object behaviour in the IDL object definition.

- CORBA object interfaces are used to group objects that share the same attributes and operations.
- A CORBA object interface can be derived from another object interface and inherit its features, such as attributes and operations.

Each object instance is identified by an Interoperable Object Reference (IOR). The IOR is an opaque handle to objects and used to invoke operations on objects. Through ORB Services, CORBA provides object location transparency when using IORs to invoke objects.

While CORBA IDL defines the operations for an object, this definition only represents the interface to the object. There may be many different actual implementations of this interface.

3.1 CORBA SERVICES

OMG's Common Object Services [13] include a set of CORBA services which are very useful in providing distributed ATM Network Management. The following services are particularly useful in ATM management applications and are used in the development of this demonstrator.

- Transaction Service
  The transaction service provides a method safely performing a series operations on objects which may be distributed across many hosts. A transaction has the ACID (atomic, consistency, isolation and durability) properties.
- Relationship Service
  The relationship service stores information about the relationships between CORBA objects. Each object in a relationship plays a certain *role* in the relationship. For example, in a *containment* relationship, an ATM location contains an ATM node, the object ATM node plays a *contained* role, while the object ATM location plays a *contains* role.
  The CORBA Relationship Service is a very general service. Any types of object relationships can be described and maintained using the facilities supported by this general service. In some cases, it is required to capture a special type of object relationships, such as the topological relationships among different network elements. A specific relationship service, Topology Service, is part of the HP OpenView DM CORBA platform which provides a better solution to the network management problems.
- Notification Service
  The notification service allows objects (notification producers) to emit notifications when an event occurs. Clients can receive these notifications if they have previously registered to receive notifications from that notification producer. Arbitrary data can be passed along with the notification.
- Logging Service
  The logging service provides a facility for distributed clients to log messages to a unified log. The log presents the illusion of being a single log but is actually stored in a distributed manner across all hosts of the installation. The messages can be retrieved later, with ordering maintained correctly.

3.2 CORBA FOR DISTRIBUTED NETWORK MANAGEMENT

The added value to network management technology which CORBA can provide lies in its distribution capability and generic object services. CORBA's distribution mechanism plays a number of major roles:

- It can be used to distribute management function
- It can be used to construct distributed management applications based on the management function
- It can be used to distribute management domain (OSI networks, databases and other resources).

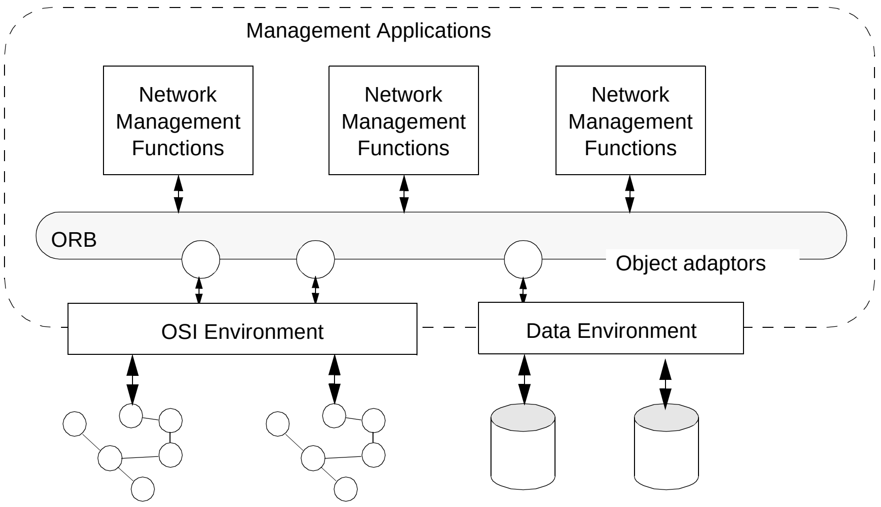
Figure 1 depicts this architecture.

The key to integrating the OSI environment is to build a set of object adaptors that map abstract CORBA requests to OSI operations [4, 10]. With this support in place, the management functions such as the ATM connection management can be built as CORBA objects.

4 OVERVIEW OF THE ATM MANAGEMENT ARCHITECTURE

The CORBA based ATM management architecture contains three major components:

- A management platform
- A set of management objects


Management Applications
Network Management Functions
Network Management Functions
Network Management Functions
ORB
Object adaptors
OSI Environment
Data Environment
Figure 1: Distributed Network Management Architecture Based on CORBA

- A management application.

The ATM management architecture is illustrated in Figure 2 and its components are described below:

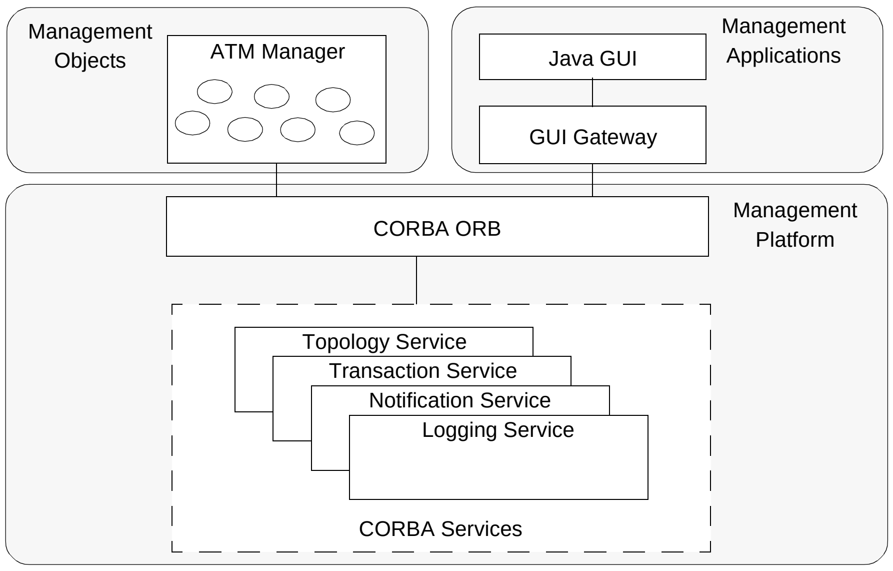


Figure 2: ATM Management Architectural Components

In this management architecture, CORBA is selected as the management platform for the ATM network management system. In addition to the ORB, a number of CORBA Services are also important in the ATM management. This platform is the kernel part of the distributed ATM management system

The ATM manager is a collection of CORBA objects which represent and manage an ATM network. These objects reflect the ATM networks topology and status. It is the integration point to the managed physical ATM network elements.

There are three basic types of objects in the ATM Manager:

- Objects which represent physical elements of the network such as switches, hosts and physical connections.
- Objects which represent logical elements of the network such as PVCs and locations.
- Objects which offer management functions for the ATM network such as PVC construction and deletion.

Each object has a unique name. The relationship between different objects is maintained in the topology service.

A GUI based management application is the management interface for operators to issue management requests and perform management operations on the managed ATM network.

Java was chosen for the GUI development because of a number of strong features:

- GUI construction in Java is faster, easier and more robust.
- Java GUIs are inherently very distributable.
- The bridging software between Java and underlying CORBA is available which enables easy development of a GUI gateway to the CORBA environment.

Examples of the management functionality provided by the management application include:

- display the ATM network topology
- create and delete PVCs
- query PVC path information
- update topology to reflect the change in the network

5 IMPLEMENTATION

To demonstrate the distributed nature of the ATM management, the management system is implemented on multiple hosts. In the implementation, two HPUX machines are used. Each host simulates and manages an ATM subnetwork. It contains the following components:

- A CORBA environment
- An ATM manager
- Multiple Java GUIs and a GUI gateway

5.1 ARCHITECTURE

The physical architecture of the system is illustrated in Figure 3:

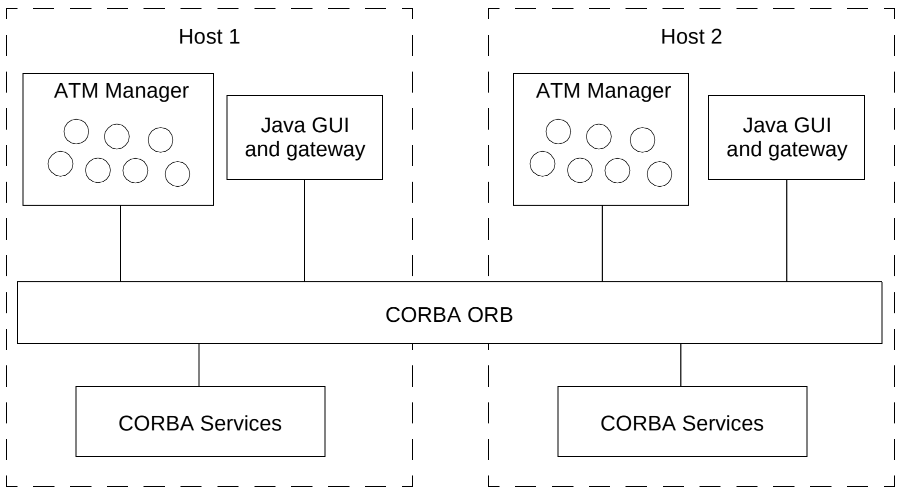


Figure 3: Physical ATM Management Architecture

5.2 OBJECT TOPOLOGY SERVICE

HP OpenView DM CORBA is selected as the CORBA platform for the implementation. The main reason for using HP OpenView DM CORBA is its Topology Service. As mentioned earlier, Topology Service is a special Relationship Service. It focuses on the network topology aspects of the object relationship. In the case of network management, a major requirement is to remember the connections between different network devices and other elements and to be able to display this type of information graphically. The Topology Service provided by the HP DM CORBA supports the storage of the network topology information, and provides a flexible query engine for retrieving this information.

The topology service stores information about the relationships between CORBA objects. *Associations* are formed between objects. Each object in an association has a *role* in that association.

The functionality offered by the topology service can be split into three areas: metadata, data, and queries.

Metadata is the *type* information of the object relationships stored in the topology service. Every object managed by topology has a type. Types exist in a type hierarchy (in most cases, the topological type of an object will simply be its IDL interface name). Metadata is typically created during the installation of an application.

Topology metadata restricts how objects may be related to each other by specifying the minimum and maximum (which can be infinity) cardinality of roles in associations. For example, an ATM physical connection object must be associated with exactly two ATM node objects.

Once the metadata is established, relationship information between object instances can be stored.

All operations on the topology service are performed within a transaction. If no transaction is active when the operation is called, one will be started by the Topology Service. Any restrictions specified on how objects may be related are enforced by the Topology Service. The restrictions are checked at the end of a transaction, and the transaction is rolled-back if the state which would be committed is not consistent with the metadata restrictions. This ensures that relationship information stored in topology remains consistent and it is also consistent with the object model.

While the relationship information can be retrieved using typical *get me the things related to this* type of calls, the Topology Service offers a much more powerful and flexible interface for accessing this information: the topology query language (TQL).

TQL allows topology information to be retrieved using a text string as the query. For example, the following TQL finds all of the **`ATMNode`** objects which are associated with **`ATMLocation`** objects which are associated with the specific object named **`ATMWorld`** (in other words, it finds all the **`ATMNodes`** in the world):

```
AENAME “ATMWorld” - ATMLocation - ATMNode
```

The results for the above query is a list of (**`ATMLocation`**, **`ATMNode`**) pairs satisfying the query. Alternatively, the results can be returned simply as the union of all the objects in the list.

More complex queries are possible which make use of association types/roles, logical operators or repeating units.

5.3 OBJECTS IN ATM MANAGER

The ATM Manager contains mainly a set of ATM management objects which represent the ATM network. Following objects are defined in the ATM manager. The topological information about these objects are stored and manipulated in the topology service.

- **`ATMFactory`**
  An **`ATMFactory`** object exists in all ATM Managers. It creates and deletes all of the other objects in the manager. It also adds the objects to topology on creation and removes them on deletion. When an object is removed from the Topology Service all associations related to that object are automatically deleted.
- **`ATMNode`**
  An **`ATMNode`** object represents a node in the managed ATM network. It is an abstract base type with two concrete subtypes: **`ATMHost`** and **`ATMSwitch`** which are used for hosts and switches respectively.
- **`ATMPhysConn`**
  An **`ATMPhysConn`** object represents a physical connection between two nodes in the ATM network. It is an abstract base type with two concrete subtypes: **`ATMSSConn`** and **`ATMHSConn`**, which represent switch to switch connections and host to switch connections respectively (host to host connections are not possible in an ATM network).
- **`ATMPVC`**
  An **`ATMPVC`** object represents a Permanent Virtual Circuit (PVC) in the ATM network. A PVC is a logical route for data between two hosts. A PVC must have two hosts as endpoints and traverse at least one switch.
- **`ATMLocation`**
  An **`ATMLocation`** object represents a logical location in the ATM network. An ATM network would be divided into locations based on geographic distribution. All **`ATMNode`** objects are associated with one **`ATMLocation`**. This provides a starting point for operations specific to a location.
- **`ATMConnMgmt`**
  An **`ATMConnMgmt`** object creates and deletes PVCs.

PVCs are created by specifying the two host endpoints, the name of the new PVC and, optionally, the path for the PVC. The path for the PVC is specified as a list of **`ATMSwitch`** objects. If a path is not specified, the **`ATMConnMgmt`** will execute a query in the Topology Service to find the shortest path between the endpoints.

Once the PVC object is created by the **`ATMFactory`**, associations are created in topology between it and all of the nodes and physical connections along its path.

PVC deletion is handled by calling **`ATMFactory`** to delete the object. This takes care of removing all associations to the PVC automatically.

`ATMConnMgmt` objects are notification producers. When a PVC is created or deleted, a notification is emitted. The data sent with the notification is a single text string which is suitable for forwarding without any processing to the Java GUIs.

- **`ATMWorld`**
  There is only one **`ATMWorld`** object. All **`ATMLocation`** objects are associated with it, this provides a starting point for global operations.

5.4 JAVA GUI AND ITS FUNCTIONALITY

The GUI developed in Java provides a management information to operators.

When the Java GUI starts up, it makes a socket connection to a GUI Gateway with the hostname and port number of the gateway passed to the GUI on invocation. It requests a list of all **`ATMLocation`** objects and a list of all ATM objects in those locations and displays a unified view of the whole ATM network. It then waits for an operator to issue management operations. The Java GUI can perform the following actions:

- Network topology display
- PVC creation and deletion
- Query PVC path information
- Asynchronous updating

The GUI can be used to display either a unified view of the entire ATM network or an subnetwork at a specified location. Switches, hosts, physical connections and PVCs can be shown or hidden as desired. The switches and hosts are shown in two concentric circles with the switches in the inner ring.

Figure 4 shows a unified view of a small ATM network.

Lines between hosts are PVCs and other lines are physical connections.

PVC creation and deletion operations can be issued by operators. A PVC can be created by selecting the menu option **`Add PVC`**. The operator is then required to provide the name of the PVC to be created and click on the source and destination hosts. A PVC can be deleted by simply selecting the menu option **`Delete PVC`** and click on the PVC to be deleted.

Information about the paths which PVCs follow through the network can be displayed in three ways:

- PVC Path
  A window showing the hosts, switches and physical connections which a PVC traverses can be displayed by selecting **`Show PVC Path`** from the menu and clicking on a PVC.
- PVCs through switch
  A window showing all the PVCs which traverse a switch and their host endpoints can be displayed by selecting **`Show PVCs through switch`** from the menu and clicking on a switch.
- PVCs through physical connection
  Similarly, a window showing all the PVCs which pass through a physical connection and their host endpoints can be displayed by selecting **`Show PVCs through physconn`** from the menu and clicking on a physical connection.

When a PVC is created or deleted by another operator, a notification is produced by the **`ATMConnMgmt`** object which performed the operation. When the GUI gateway receives this notification, it sends a message to all of the connected GUIs so they can update the current display to reflect the changes.

5.5 GUI GATEWAY

The GUI Gateway is the glue between the Java GUI and the underlying CORBA platform, which allows a Java GUI to communicate with CORBA objects and services. It accepts socket connections from multiple

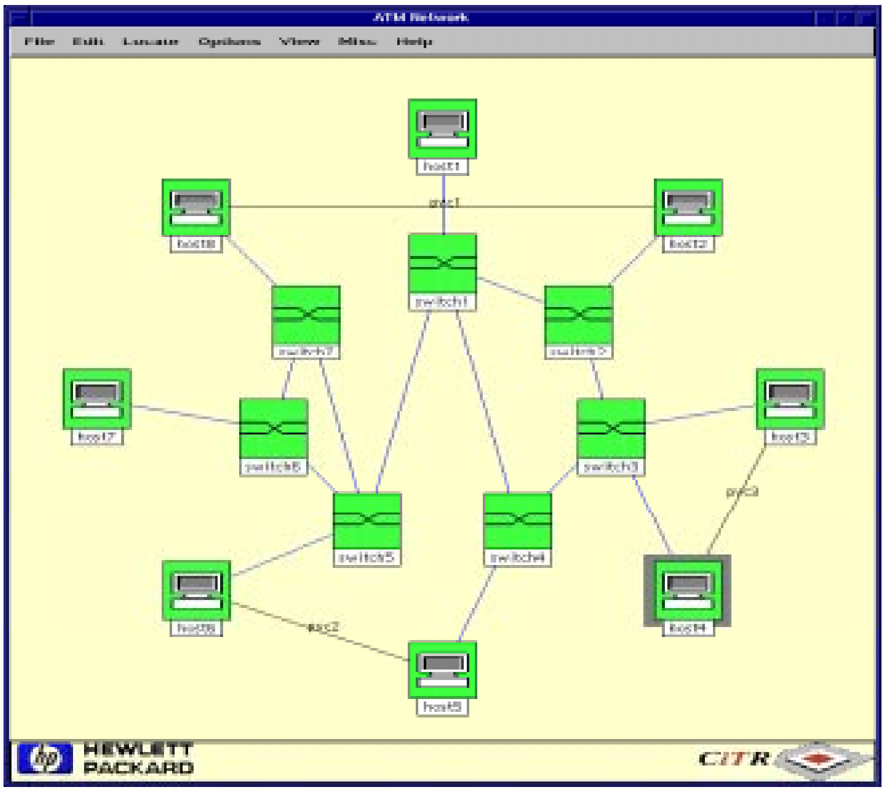


Figure 4: A unified view of an ATM Network

GUIs and has three basic functions: responding to commands from the GUI, passing on notifications from the ATM managers and logging interesting events.

ATM Objects are always referred to by their name when communicating with the GUI.

5.6 A SERVICE EXAMPLE—VIDEO ON DEMAND

A service example, a simulated VOD service, is provided with the implementation of the system in order to demonstrate the management capability of the CORBA based ATM network management system. This example uses the functionality supported by the system and demonstrates the integration between different technologies used in developing the system.

A CORBA service object, **`ATMVideo`**, is defined to represent a video which can be sent over the ATM network to a viewer. When a request is received to show the video, this **`ATMVideo`** object performs the following functions:

- Determines the ATM host where the video is to be sent. This is given to the **`ATMVideo`** object in the request.
- Queries the Topology Service to find the host where the video currently resides.
- Calls the **`ATMConnMgmt`** object to create a PVC between the two hosts.
- Waits for a notification from the **`ATMConnMgmt`** object that the PVC has been created.
- Plays the video.
- Waits for the video to stop playing.
- Calls the **`ATMConnMgmt`** object to remove the PVC

6 DISCUSSIONS

CORBA as a distributed object management platform can be used to manage telecommunication networks and services, especially for managing high level management objects at the network and service level. The distributed features and the rich services dramatically reduces the development time. The video on demand service showed that applications built using the CORBA platform are easy to extend and to integrate with other applications.

CORBA as a platform itself does not provide the development environment required for management applications. However, by selecting Java as the GUI development mechanism, the management applications can be quickly developed based on the available integration technology between Java and CORBA.

The ATM Manager is an important component in the management architecture. It provides an abstract layer of ATM management network and acts as an integration point to underlying network and element management layers which use different management protocols, such as CMIP and SNMP.

The following experiences are gained from this study in the area of using CORBA for broadband network and service management.

First, a set of standard adaptor objects are required. This layer of objects acts as a gateways between CORBA and the underlying communication environments, providing some generic functions used by many service management applications. One such object adaptor in the demonstrator is the ATM Connection Management object. This object represents a high level abstraction of the physical connections based on a network of ATM switches. However, this is a sufficiently general object that can be used in many other applications. This layer of objects is best defined by industry standards bodies, and to be offered as a set of standard class libraries. In fact, TINA-C's definition of base level service management objects and NMF's component sets [11] are exactly these kinds of objects. There is still lack of well accepted, standardised definitions in this area.

Second, a set of higher level objects with more service semantics can be defined using standard adaptor objects. With this layer of objects, the differences in the connection management of the physical broadband network can be encapsulated. Although the demonstrator supports ATM connection management, there is no reason why the same principle cannot be used to apply to the SDH and other network management for the connection management thus to achieve the end-to-end broadband connection management. This abstraction of objects enables us to model a whole or part of a service, rather than a generic function. There is very little standards effort focused on this area. This is due to the fact that the service offerings and their management technology represents the real competition and the revenue side of the value chain. There is resistance to standardise this layer of services. However, there is still a requirement for more generic components. One example is the NMF's solution sets.

Third, more abstract service objects can be defined to represent service offerings and higher level management functions. In the demonstration, the CORBA relationship service in the form of topology is used to add more semantics to existing objects. The operations defined by service objects can also use a transaction service to add transaction semantics in order to provide better support for user transactions. At this level, we see much less industrial standards activities, as this set of objects are more business oriented and therefore pose a wider range of variations. These are the true areas of competition of service offering—anyone, without owning any part of the network infrastructure, can provide services in the form of business objects, because one only needs to build on top of technology independent solution sets.

In addition to the research direction to support more integration with other broadband technology, the work reported here can also take another important direction. The demonstrator only provides the connection management for the ATM network. There are, however, other functional areas of service management which can be supported by using the similar architecture. These include the well-defined TMN functional areas such as fault, performance, accounting and security management, as well as NMF's business processes such as ordering, trouble ticket and service level agreement management. The CORBA distributed infrastructure and its services provide an environment to develop these functions and to integrate these in a service management application.

ACKNOWLEDGEMENTS

The authors would like to thank the many people who have contributed to this research. In particular: Greg Hall, Peta McRae, Robert Coote, Norm Lawler and Jason Etheridge.

REFERENCES

[1] E. K. Adams and K. J. Willetts, *The lean communications provider—Surviving the shakeout through service management excellence*, McGraw-Hill, 1996.

[2] W. J. Barr, T. Boyd, and Y. Inoue, The TINA initiative, *IEEE Communication Magazine*, March 1993.

[3] G. Chen and Q. Kong, Integrated TMN Service Provisioning and Management Environment, to appear in the *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management* (ISINM 97).

[4] G. Chen, M. Neville and Q. Kong, Distributed network management using CORBA/TMN, *Proceedings of the 7th IFIP/IEEE International Workshop on Distributed Systems Operation and Management* (DSOM), October 1996.

[5] J. P. Chester and K. R. Dickson, Standards for integrated services and networks, *Integrated Network Management IV*, A. S. Sethi, Y. Raynaud and F. Faure-Vincent (eds.), Chapman & Hall, 1995.

[6] M. Flauw and P. Jardin, Designing a distributed management framework—An implementer's perspective, *Integrated Network Management IV*, A. S. Sethi, Y. Raynaud and F. Faure-Vincent (eds.), Chapman & Hall, 1995.

[7] JavaSoft, The Java Language: An Overview, http://www.javasoft.com/doc/Overviews/java/.

[8] JavaSoft, The Java Language Environment White Paper, http://www.javasoft.com/doc/language_environment/, May 1996.

[9] JavaSoft, Java Beans: A Component Architecture for Java, http://splash.javasoft.com/beans/WhitePaper.html, July 1996.

[10] Q. Kong and G. Chen, Integrating CORBA and TMN Environments, *Proceedings of IEEE/IFIP Network Operations and Management Symposium*, Kyoto, April 1996.

[11] Network Management Forum, NMF Solution & Component Sets, http://www.nmf.org/sets2.htm.

[12] Object Management Group, The Common Object Request Broker: Architecture and Specification, Revision 2.0, July 1995.

[13] Object Management Group, CORBA Services: Common Object Services Specification, Revised edition, OMG document number 95-3-31, 1995.

[14] M. De Prycker, *Asynchronous Transfer Mode—Solution for Broadband ISDN*, 2nd Edition, Ellis Horwood, 1993.

[15] W. Widi, *Standardisation of Telecommunication Management Networks*, Ericsson Review, No. 1, 1988