Towards TMN-based Integrated Network Management Using CORBA and Java Technologies

Jae-young Kim, Hong-Taek Ju and James Won-Ki Hong
Dept. of Computer Science and Engineering, POSTECH
{jay, juht, jwkhong}@postech.ac.kr

Seong-Beom Kim
Telecommunication Network Research Lab.
Korea Telecom
sbkimn@kt.co.kr

Jong-Tae Park
School of Electronic and Electrical Engineering,
Kyungpook National University
park@ee.kyungpook.ac.kr

Abstract

Today’s complicated and heterogeneous telecommunication network environments need fully-integrated, cost-effective, user-friendly management systems. When developing such management systems in distributed environments, many telecommunication companies are carefully adopting CORBA and Java technologies. CORBA technology enables developers to create and manipulate distributed management system components easily while Java technology liberates human users from complicated management system interfaces. In this paper, we propose a TMN-based integration framework for distributed network management by combining both CORBA and Java technologies. The framework generalizes a layered architecture from the management agents layer to the Java GUI layer. It also provides a generalized TMN management interface on top of gateway systems to coordinate heterogeneous management protocols. By using this interface, the TMN management functionality can be easily constructed to provide powerful TMN management services to administrators via user-friendly Web browsers. On the basis of this framework, we have designed and implemented a TMN alarm surveillance system which validates our framework.

[Keywords: TMN-based Network Management, Management Framework, CORBA, Java, Web-based Management, Integrated Network Management]

1. Introduction

The rapid growth of networking technologies in recent years has created much more complex and heterogeneous network environments. To manage network devices and services in such complex
environments, various organizations have developed management platforms using different kinds of management protocols (e.g., SNMP [1], CMIP [2], TMN [3]).

Since there are many different network management schemes which can not be easily integrated, there have been a lot of research efforts to harmonize them with new technologies. Joint Inter-Domain Management (JIDM) [4] is one of the organizations doing such integration. The goal of JIDM is to integrate CMIP, SNMP and CORBA technologies [5, 6].

To cover both computer and telecommunication network management at the same time, TMN is chosen as a good solution because it provides a systematic and consistent management architecture. However, the TMN framework does not provide implementation details in its standard documents. The realization of a TMN framework has become a main issue in the network management community.

Common Object Request Broker Architecture (CORBA) [9], which was proposed by Object Management Group (OMG) [10], has been widely adopted for developing distributed information systems in nearly all areas of computing and telecommunications systems [6]. Since CORBA provides an infrastructure for the interoperability of various object-oriented applications in distributed environments, people have started using CORBA for integrating network management systems. TMN realization using CORBA technology is chosen as a good solution by many groups such as JIDM, TINA and TMF [23]. JIDM proposed CORBA/TMN interworking specifications which define CORBA interfaces that can handle CMIP- and SNMP-based devices [4, 5].

However, there remains many more realization efforts on top of the CORBA interfaces. TMN functionality should be realized by using CORBA technology, without depending on low-level protocol gateways. To do that, there should be a facility to enable many different protocol gateways and provide generalized CORBA management interfaces to the upper level. This kind of facility can be formed as a ‘proxy coordinator’. Also, on top of the proxy coordinator, TMN functionality should be analyzed and divided into CORBA services, which are organized harmoniously to provide various TMN services to users.

While the TMN-based network management framework has been gaining public attention, a Web-based network management framework [31, 32, 38], which absorbed rapidly developing World-Wide Web (WWW or Web) [14] technologies. WWW technology such as Sun Microsystem’s Java [12] enables software developers to create sophisticated applications working on the Internet without losing the user-friendly Web interface. Java also provides a portable object-oriented component framework
across multiple platforms.

Since many proprietary network management solutions are too complicated and expensive for general users to obtain and maintain, there has been a strong need for inexpensive, user-friendly network management solutions. JMAPI [31] and WBEM [32] are examples of such efforts. Their goal is to provide a network management solution that is executable on any Web browsers.

CORBA technology can be integrated seamlessly with Web technology. OMG has completed CORBA Interface Definition Language (IDL) to Java mapping in the recent CORBA specification [7]. Many CORBA implementation vendors now support Internet Inter-ORB Protocol (IIOP), which extends the Object Request Broker (ORB) over the Internet [7]. CORBA classes implemented in Java enable Java applications to use CORBA services via IIOP.

In this paper, we integrate a number of different technologies and suggest a TMN framework using CORBA and Web. Any Web browser can gain access to the Web-based TMN network management system which handles any type of management protocols such as SNMP, CMIP, or proprietary protocols via protocol-dependent gateway systems. We have selected the alarm surveillance management service as our implementation example. Since the alarm surveillance management service is one of the most well-defined TMN services in the TMN standards documents, we believe this implementation architecture can be easily extended to support other TMN services as well.

The organization of the paper is as follows. Section 2 surveys related work, summarizes TMN, CORBA and Java technologies, and shows the integration approaches of the different technologies. In Section 3, we propose a TMN integration framework using CORBA and Java technologies. This generalized framework enables TMN management application developers to implement applications easily. Sections 4 and 5 present the design and implementation details of our alarm surveillance system, respectively. Section 6 summarizes our work and discusses possible future work.

2. Related Work

In this section, we survey several technologies used in integrated TMN-based network management systems. They are TMN, CORBA, and Java technologies. We then offer some examples of technology integration. Open Management Group (OMG) [10] and Joint Inter-Domain Management (JIDM) [5] provides two different viewpoints of TMN/CORBA integration. Also examples of integrated frameworks which combine different technologies are presented.
2.1. Telecommunications Management Network

Telecommunications Management Network (TMN) [3] has been developed by International Telecommunication Union (ITU) to provide the solutions for the end-to-end management of networks and services provided by different service providers. TMN is a framework for the integrated management of network elements, networks, and services. Within the general TMN framework, there are three basic architectures which can be considered separately when planning and designing a TMN: physical, information and functional architectures. The physical architecture describes realizable interfaces and examples of physical components that make up the TMN. The functional architecture describes the appropriate distribution of functionality within the TMN to allow for the creation of function blocks from which a TMN of any complexity can be implemented. The definition of function blocks and reference points between function blocks leads to the requirements for the TMN recommended interface specifications. The information architecture described in ITU-T M.3010 [3], based on an object-oriented approach, gives the rationale for the application of Open Systems Interconnection (OSI) [26] systems management principles to the TMN principles.

Figure 1: The Relationship Between the Physical Architecture and Function Blocks in a TMN

Figure 1 shows the relationship between the physical architecture and the function blocks. TMN management services are realized by these function blocks. Among the function blocks, the Operation System Function (OSF) block is the place where most TMN management functions, such as the
monitoring of communication functions of network elements, and the processing of management information for coordination, are actually performed. TMN management application functions are categorized into five functional areas: performance, fault, configuration, account and security management. Some of these TMN functions in ITU-T Recommendation M.3400 [8] can be realized by taking advantage of OSI Systems Management Functions (SMF) and the Common Management Information Service (CMIS) as underlying information-carrying vehicles.

Figure 2: TMN Functional Architecture

Figure 2 shows the TMN functional architecture where operations system functions are conceptually classified into business, service, network, and network element management functions. This hierarchical functional structure is based on the TMN Logical Layered Architecture described in ITU-T M.3010 [3]. Furthermore, these functions can be realized in a distributed manner. In summary, the TMN management functions whose aims are to realize the OSF specified in TMN functional architecture can be distributed and hierarchically layered, and can access the management information that is specified in an object-oriented specification language.

This complicated functional architecture tends to make the realization of TMN functions rather cumbersome and very difficult for ordinary programmers and system developers. In addition, there are other powerful functions such as mediation to be implemented. These complexities may be some of the reasons why a CORBA-based approach for the realization of TMN management services has been attempted in spite of the previous efforts by the Telemanagement Forum (TMF) [23] to speed up the realization process of the TMN standards.
2.2. OMG’s CORBA and TMN Integration

OMG presents an architecture for a CORBA-based telecommunication network management system [22]. The proposed architecture includes an outline implementation alternative to the OSI open interface and the OSI System Management concepts using the CORBA paradigm. An objective in the development of this new implementation architecture is to reuse the knowledge and expertise acquired over the years in the establishment of the ITU-T/OSI standards. Moreover, it also ensures that the management system will provide complete compatibility with proprietary, CMIP, SNMP, and CORBA-based network elements.

2.2.1. CORBA for Managing Systems

OMG has been pursuing CORBA-centric telecommunications management systems supporting gateways to other communication paradigms as necessary. With the perspective focused on operation systems within a TMN, there are a lot of issues to be resolved, such as a framework for the development of operations systems, existing standards-based management, and systems management.

Figure 3 illustrates the OMG’s operations system framework for integration of TMN and CORBA [6]. A Management Application Function (MAF) [3] component is introduced with the same meaning as in TMN. MAF implements a management service. MAFs are not subject to standardization within TMN and are similarly not subject to standardization within the OMG. They can be considered to be specific cases of applications objects within the OMG’s OMA [11]. The current operations system framework includes the systems management facilities defined by the X/Open system management group [46]. It
also includes the proxy devices/services defined by the JIDM group. The space reserved for telecommunications management facilities is currently void and it has requested the telecommunication group to devise suitable services required by the telecommunications management community for insertion in this category. Prime candidates for this category currently include a telecommunications management event service and a telecommunications relationship service.

2.2.2. CORBA for Managed Systems

The OMG’s Systems Management starts from the assumption that CORBA objects will represent managed resources in computing telecommunications systems. As a translation between Guidelines for the Definition of Managed Objects (GDMO) [39] and CORBA Interface Definition Language (IDL) [7] is accomplished by JIDM, CORBA could be a candidate for implementing a new generation of telecommunications management interfaces.

♦ CORBA Managed Objects

In this case, each Managed Object class is defined by one IDL interface. It is necessary to define a commonly agreed managed object class for all CORBA based management systems. A base managed object class is able to include information about the class of a managed object, the identifier for a managed object class instance, the properties of the managed object instance, etc. Then the base managed object class is specialized for each class of managed object in the system.

♦ Managed System with CMIP/CMIS Interface

The JIDM group looked at providing CMIP/CMIS [40, 41] to CORBA gateways in order to facilitate this continued deployment where the distribution of the management part within the managed device is handled using the CORBA mechanism. While in the JIDM activities, the proxy device which presents the CORBA interfaces to network resources may physically reside at the following places: within the network element, between the network element and the operations system, or within the domain of the operations system. However, the location of the proxy device is of little consequence to the managing system. The JIDM GDMO to IDL mapping and the proxy device specifications will facilitate the use of CORBA to develop the management part of network devices. In this environment the proxy device will assume the agent role in the OSI manager/agent context. We will describe the work of JIDM for integrating TMN and CORBA in the following section.
2.3. JIDM’s Specification Translation and Interaction Translation

The JIDM (Joint Inter-Domain Management) task force which is jointly supported by both Network Management Forum and X/Open has been developing specification and interaction translation methods for the interoperability of SNMP, CMIP and CORBA technologies. In order to solve the interoperability problem, the following two main areas need to be addressed between a particular pair of domains: specification translation [4] and interaction translation [5]. Specification translation describes a mechanism for translating between GDMO (MIB definition language used in conjunction with CMIP), SMIv2 (MIB definition language used in conjunction with SNMP) [42], and CORBA’s Interface Definition Language (IDL). They provide translation algorithms to translate the specifications of managed objects used in GDMO/ASN.1 and SMIv2 to those of IDL and vice versa.

Interaction translation describes a mechanism for dynamically converting between the management protocol in one domain to another without either party necessarily being aware of the conversion. This allows objects in one domain to be represented in the other domain, and it also allows the interaction to be governed by the domain of choice rather than by the domain in which the target object is implemented. For example, an object in the CORBA domain should be able to interact with a GDMO object as if it were in the CORBA domain.

2.4. Examples of Integrated Framework

There are a few example research efforts which provide integrated network management frameworks using CORBA and Java technologies. Carlos Westphall has designed and implemented a CORBA-based alarm surveillance system applied to ATM switch management [18]. Nokia Research Center has developed the Distributed Computing Platform (DCP) to support distributed telecommunications services [36]. CiTR has proposed an integrated TMN service provisioning management environment for distributed VOD services [37].

Westphall’s work is to gain experiences with supporting distributed management in TMN frameworks through CORBA and Java. TMN alarm surveillance service based on ITU-T Q.821, M.3100, and X.721 recommendations [19, 20, 21] was considered. From the recommendations, classes of managed objects were extracted, specified, and implemented. All the Managed Object (MO) classes are specified in CORBA IDL and the alarm surveillance system was implemented using Visibroker [29] and Java. The CORBA alarm surveillance was shown to be a simple system of low cost compared to a system developed in the OSI framework, when implemented in Java. Also, the development with CORBA is simple due to the IIOP protocol and is of low cost due to the use of a development platform.
such as Visibroker that explores Java with all of its facilities, as an object-oriented programming language.

Nokia Research Center’s DCP prototype has been proposed as a standard telecommunication platform to create, manage, and invoke distributed telecommunication services. DCP implements the components of all services as CORBA objects and the objects fall into three categories; CMIP distributed objects, SNMP-based systems, and users. CMIP distributed objects communicate to applications via managed-object IDL interfaces. A CMIP-string package passes protocol data units between CMIP systems encapsulated as objects. CORBA naming and trading services facilitate communication among distributed objects. SNMP-based systems interface to DCP via the CMIP-SNMP gateway. Users access the network from Web browsers, sometimes through CGI gateways, server extensions, and Java or HTTP daemons. To provide object-oriented access to network management services, several management applets such as SNMP MIB browser, GDMO browser, and CMIP PDU sender are implemented. This work emphasized that it was possible to prototype the integration of the Web, Java applets, and CORBA objects with legacy network management system.

CiTR has proposed an integrated service provisioning and management environment with a Java-based user environment, a CORBA-based distributed service management and a TMN-based network management environment. CiTR has focused on the experience of integrating CORBA, Java, and OSI technologies to achieve integrated management. They have designed an integrated management architecture to support automated service ordering and provisioning. The architecture has three basic functional layers: the interface layer supports different type of access to the system and includes Java-based GUI for both operators and customers; the distributed middle layer provides the main service management functionality and CORBA is used as the distributed infrastructure; and finally, the network and element management layer consists of a management platform and a set of network management functions based on OSI-based TMN. The proposed architecture is a scalable, extensible, reliable, and viable solution to telecommunication business process and it has integrated CORBA/TMN and CORBA/Java technologies.

3. TMN Integration Framework using CORBA and Java

Though the TMN standards specify various management functions and services, the detailed management operations for the realization of these TMN management services are not part of the standard documents. Also the ITU-T recommendations do not define any implementation details [23]. Consequently, there have been many research efforts to realize the TMN framework. TMF has
proposed a detailed functional descriptions of TMN management [24] and OMG continues to work on integrating TMN with CORBA technology. JIDM has answered to OMG’s proposal to specify CMIP/CORBA and SNMP/CORBA translations.

However, there remains much more work to complete the full integration. Since integration work has only been done on the gateway level until now, a well-defined generalized architecture from the agent level to the user-interface level is required. Also, as CORBA technology extends to be harmonized with Java technology, Web-based GUI implemented in Java has become a popular user interface for various CORBA-based management systems.

In this section, we propose an integrated TMN framework using CORBA and Java technologies. Figure 4 describes the functional blocks of the framework. Hierarchically, gateway proxies, proxy coordinator, system management functions, TMN functions, high-level service functions, and java-base applications are stacked as depicted in Figure 4. Every module is implemented as a CORBA component and interconnected to each other via ORB. On the right side of the figure, auxiliary facilities reside: managed object factory, COSS, and WWW server. They are also accessible through the same ORB.

Gateway proxies provide the method for interoperations between CORBA technology and different
network management protocols such as SNMP, CMIP, and other proprietary network management protocols. Protocol-dependent information structure and interaction functionality are converted to CORBA interfaces and vice versa. SNMP’s SMI, CMIP’s GDMO, or any other proprietary information model should be converted in this layer. JIDM mainly contributes detailed specifications to these kinds of translations. Specification and interaction translations are performed in the gateway proxies level.

Proxy coordinator is a module that generalizes protocol-dependent CORBA management interfaces. Accordingly, it can provide generalized CORBA management interfaces to the upper level. This layer also handles the managed object factory to keep managed object class definitions and their IDL definitions.

On top of the proxy coordinator there are systems management functional components implemented in CORBA. These CORBA components represent each general TMN system function. Component technology in CORBA enables these components to be interconnected to each other easily in order to combine sophisticated TMN services.

TMN functions are the five general functions in the TMN framework. They are fault management, configuration management, account management, performance management, and security management. Each management function can be realized easily just by picking up available CORBA components in the systems management functions layer and combining them via the interfaces they provide.

High-level service functions are the specific task-oriented management services for the end-users. Users of a TMN framework can define any kind of service in this layer that can use TMN functions provided from the lower layers. For instance, management of transmissions paths, switched systems, customer services, or alarm surveillance services can be constructed.

Since we suppose Java-based applications are requirements, the GUI applications should be able to run on any Web browser. Java with CORBA interfaces is chosen to be the programming language in this layer. The CORBA extension in Java enables software developers to implement distributed GUI applications such as a TMN management system easily. A Web server is used for downloading Java applications to Web browsers. Some CORBA implementations (e.g., OrbixWeb [34] and Visibroker [29]) provide methods for integration of a Web browser and CORBA interfaces.
4. TMN-based Alarm Surveillance System

Based on the generalized TMN integration framework we have proposed in the previous section, we have developed an example TMN management system which provides an alarm surveillance service.

4.1. Alarm Surveillance System

The alarm surveillance service is one of the subfunctions defined in the TMN fault management standard [8]. It is defined as a set of functions that enables the monitoring of telecommunication network elements for both events and alarm conditions. Since the alarm surveillance service is much more well-defined in TMN standard documents compared to other TMN services, and the service requires nearly all aspects of possible agent functionality, the TMN alarm surveillance service is frequently chosen as an example system in many research activities [4, 5, 18]. To validate our TMN integration framework with CORBA and Web technologies we also chose the TMN alarm surveillance service as our example design and implementation.

The TMN alarm surveillance service operates in the following manner. An alarm information is generated by a network element (NE) due to the detection of an abnormal condition or failure. A TMN manager provides the capability to monitor NE failures in near-real time. When such a failure occurs, an indication is made available by the NE. Based on this indication, the TMN manager determines the nature and severity of the fault. Such alarms should be reported to human users by means of audible or visual warnings at the time of their occurrence, stored for future reference, or both. An alarm may also cause further management actions within the NE that lead to the generation of other fault management data.

In TMN standard documents, functional requirements of a TMN alarm surveillance system are specified [8]. In order to enable the TMN to perform alarm surveillance, NEs must:

- allow monitoring of alarm conditions in a near-real time or scheduled manner.
- allow querying of alarm conditions existing on the NE.
- allow logging and retrieval of historical alarm information.

Also, an alarm surveillance system includes the following function sets:

- Alarm policy function set.
- Network fault event analysis, including correlation and filtering function set
- Alarm status modification function set.
- Alarm reporting function set.
• Alarm summary function set.
• Alarm event criteria function set.
• Alarm indication management function set.
• Log control function set.
• Alarm correlation and filtering function set.
• Failure event detection and reporting function set.

4.2. Design of a TMN-based Alarm Surveillance System

TMN is based on a specific system of operations and management of resources in a distributed system [25]. In such an environment, the information related to the resources and to the services, regarding the management and managed systems needs to be spread-out over the entire network. Thus, the support for distributed processing is an essential requirement.

As we have described in Section 2, TMN has functional, physical and information architectures and TMN-based alarm surveillance system should follow the architecture. Figure 5 presents a mapping of our alarm surveillance system to the TMN functional architecture [8, 21].

![Figure 5: Mapping of Alarm Surveillance System Functions to TMN Functional Architecture](image-url)

Basically, the TMN alarm surveillance system consists of three components: user, manager system, and managed system. Between the components, there exist interface functions, which can be mapped to design component as in Figure 5. WorkStation Function (WSF) between a user and the Alarm
Surveillance System is mapped to an Alarm Surveillance User Interface module. Operation System Function (OSF) is mapped to the main Alarm Surveillance Manager System module. Network Element Function (NEF) is mapped to the Alarm Surveillance Agent System module.

Our TMN-based alarm surveillance system has three components: management application, Web-based management server and gateway. Figure 6 presents those three components. At the bottom of Figure 6, there reside various agent systems that we should be managed. These agent systems are not part of our system. They can be any agents running in the existing network environment. Above the agent systems, the main Web-based management server represents the proposed TMN integration framework. The manager system has a number of submodules in it. The gateway modules are responsible for converting protocol-specific operations to CORBA interfaces. The proxy coordinator coordinates the CORBA interfaces from the gateway modules so that it can show generalized, protocol-independent CORBA management interfaces to the upper layers. CORBA management components are created on top of the CORBA management interfaces. The components are interconnected to each other to construct TMN management functions. Management applications written in Java should be downloaded from a Web-based Management Server to users’ Web browsers and run there. A Web server interconnected to ORB is used for this purpose. The Web server also can provide static information such as log data or HTML documents to Web browsers.
5. Implementation

In this section, we describe the implementation details of our Web-based alarm surveillance system. By following the design architecture proposed in Section 4, each module implementation is explained briefly in each subsection. We implemented our alarm surveillance system using CORBA and Java technologies. The implemented alarm surveillance system consists of four modules: agent, gateway, manager, and GUI management application. The implementation architecture of the alarm surveillance system is presented in Figure 7.

Actually, our alarm surveillance system does not include the agent modules. Instead of simulating agent systems, we used real SNMP and CMIP agents running on Unix systems. For our implementation of an alarm surveillance system we chose UCD SNMP 3.5 [43] as the SNMP agent system and OSIMIS 4.0 TMN platform [33] as the CMIP agent system. By using the C++ management interface, we have implemented CORBA/SNMP and CORBA/CMIP gateways which handle each protocol-dependent Unix agents. The OSIMIS Unix agent reports alarm messages in CMIP to the manager system when it detects the specified error conditions of the CORBA/CMIP gateway while the UCD SNMP agent reports SNMP trap PDUs to the CORBA/SNMP gateway. The alarm surveillance management system, which is composed of many CORBA management components, handles the alarm services and produces alarm signs to the human users on Web browsers.
5.1. Web-based Management Application

Web-based management application module is a GUI, which provides interfaces between human users and our management system. Web browser is selected for visual expression because Web technology has many advantages. It is platform independent, easily controlled, easily used, etc. Java satisfies above technology. Figure 8 displays the graphical user interface implemented in Java. This Java applet is downloaded from Web server and establishes IIOP connection to the ORB on the management system. We implemented the Java applet by using Java Development Kit (JDK) 1.1.6 [44] and OrbixWeb 3.0 [45].

When the downloaded Java applet is connected to a manager, it gets agent lists stored by the connected manager. Then the user can perform several management operations on agents. For example, getting MO lists, allowing (or inhibiting) alarm reporting by creating (or deleting) eventForwardingDiscriminator, modifying event criteria, getting/setting any attribute value, allowing/inhibiting logging, and so on. When the Java applet receives alarm notifications from the management system, the status label turns red and an alarm sounds until the human manager checks the status abnormality.

Figure 8 : TMN Alarm Surveillance Java Applet Interface
5.2. CORBA/CMIP Gateway

To enable CORBA to interwork with CMIP/CMIS, it is necessary to map between the relevant object models and to build on this to provide a mechanism to handle protocol conversion on the domain boundaries. We used the JIDM specification to implement “CMIP Gateway”. The gateway must construct two translations.

The first part is referred to as specification translation, and is expressed as a mechanism for translating between GDMO and CORBA’s Interface Definition Language (IDL) which was issued for defining the interactions between objects in the CORBA domain. Translation from GDMO definition to IDL definition is mandatory.

The second part is known as interaction translation and covers the mechanisms to dynamically convert between the protocols in one domain and the protocols within the other without either party necessarily being aware of the conversion. This allows objects in one domain to be represented in the other domain and the interactions can be governed by the domain of choice rather than by the domain in which the target object is implemented. For example, an object in the CORBA domain should be able to interact with a GDMO object as if it were in the CORBA domain, ideally without having to know that the target object is in a different domain.

We implemented both translations [35]. A GDMO-IDL translator for the specification translation is developed. The translator converts GDMO files to IDL definitions files for CORBA objects. We could create management CORBA objects using this IDL. For the interaction translation, we followed JIDM’s specifications [4, 5], but implemented only subset MO classes necessary to support interaction for managing agents within MOs presented.

5.3. CORBA/SNMP Gateway

Basically, the gateway that connects CORBA domain and SNMP domain is implemented by using the same method as CORBA/CMIP gateway. JIDM’s specification defines translation algorithms for converting SNMP SMI to CORBA IDL. Both specification and interaction translations are performed in the CORBA/SNMP gateway and we can get CORBA objects which represent SNMP MIB variables. The translated CORBA objects can be used to implement CORBA objects that manage SNMP agents without knowing the difference of management domains.
5.4. Manager

The alarm surveillance management system must monitor NE failures in near-real time. To satisfy the requirements we used the generalized TMN framework we suggested. On top of the gateway systems there is a proxy coordinator which encapsulates protocol-dependent interfaces. By using this generalized interface CORBA System Management Functions (SMF) components are constructed. We then created alarm surveillance high-level function set following M.3400 ITU-T recommendation. Also in order to provide this functionality to general users, GUI-supporting CORBA components are made. They are responsible for transferring management data to the Java applets running on the Web browsers.

```idl
module ASM { // Alarm Surveillance Module

    interface init_ASM { // initialize ASM
        AgentList getagentlist();
    };

    interface general_ASM { // general ASM functions
        ObjList getagentmolist(in string agentname);
        any getattributelist(in string agentname, moname);
        any getstatus(in string agentname, in string moname,
                        in string instancename, in string instancevalue,
                        in string attr);
        EventHistory gethistory(in short index, in string name);
        void setstatus(in string agentname, in string moname,
                       in string instancename, in string instancevalue,
                       in string value);
        void createobject(in string agentname, in string moname,
                          in string instancename, in string instancevalue);
    };

    interface Event { // event handling functions
        EventReceived getevent();
        void logevent (in string name, in string probableCause,
                                in string alarmSeverity);
    };

};
```

The above IDL definition describes the main functions and arguments of alarm surveillance module operations. The module ASM contains three interfaces: init_ASM, general_ASM and Event. init_ASM interface initializes the ASM module when the TMN manager starts up and the interface is used for obtaining the list of agents that the manager needs to manage. general_ASM interface contains essential operations for the alarm surveillance service. The operations are used for getting agents MO list, getting and setting status of a specified agent, getting history information from the agent, and creating new CORBA object for a given managed object. Event interface establishes an event channel for each gateway module and listens asynchronous alarm messages from agents. It also logs every alarm event for statistical information.
The following describes how the manager works. First, the manager provides interfaces for Java applets to retrieve management information for the agents. When the manager accepts input data from the Java applet, it sends an operational information to the agent through the interfaces that is supported by the gateway implemented in the subset of JIDM’s specifications as mentioned above and returns the results. When the manager detects any alarm conditions of the agents, the manager reports this condition to the Java applet by using Java/CORBA event channel that is established when the Java applet is initialized.

6. Summary and Future Work

In this paper, we have proposed a TMN-based integrated network management framework by using Web and CORBA technologies. We have implemented a TMN alarm surveillance system based on the proposed system framework. As we have pointed out, the Web-based integration has many advantages especially in constructing distributed systems.

CORBA is being used intensively to realize TMN standards over heterogeneous distributed network environments by various groups all over the world. We have used CORBA and Java to realize Web-based TMN system to provide users with user-friendly, easy-to-use, integrated interfaces and developers with the generalized implementation architecture on top of abstraction layer hiding all the protocol-dependent details. The proposed framework can be easily extended so that many other TMN services can be built on it.

In the future, we plan to provide more sophisticated TMN management services developed on our framework to validate the effectiveness of our framework. Also, we plan to enrich generalized interfaces we suggested to support more features that gateways may need.

References


