A Network Management Framework for Supporting Multiple Management Interfaces

Jae-Oh Lee

Department of Computer Engineering, School of Information Technology
Korea University of Technology and Education, Korea
jolee@kut.ac.kr

Abstract

Java technology is one of prominent developments to deploy the framework of network management which can support a variety of management interfaces for exchanging the management messages in standard and proprietary forms. There are so many evolving and enabling technologies that can be applied into the network management framework and interface. The proposed management framework can be used for the provision of a development environment in which the existing (e.g., CMIP, SNMP, CORBA, Java RMI) and emerging (e.g., XML, SOAP) interfaces might be integrated towards the platform independent approach. This can be useful for the development of network management systems that can interact with legacy and new ones in easy and integrated manners.

1. Introduction

As many Java applications are deployed in the real system, Java based network management is one of the technology parts for integrating the existing and promising management applications and systems. The challenges that TMN faces, and which it must evolve in order to meet, stem largely from the rise of many popular general-purpose technologies suitable for open distributed computing. The two main categories of such technologies are Web-based ones, which include many applications of HTML, CGI, Java Applets and XML, and object-oriented middleware based ones, such as OMG CORBA [5], Sun MicroSystem’s Java RMI [2], and Microsoft’s DCOM [12].

Next generation network services will be assembled on the fly in a plug-and-play fashion, which drastically reduces the time and efforts to develop services. New network technologies are being assembled at a fast and require dedicated network management support. In order to keep the pace with the benefits of these new offerings, network management software tools must provide short learning time requirements. In addition, network management development tools should provide for Rapid Application Development (RAD), so that the benefit of a technology is reaped before the next and more powerful technology replaces it.

There are research efforts to recognize the need for integrated solutions to manage network resources and services in open and global environment [8, 16]. In [8], a layered framework architecture, including element, network management, and visualization service is provided, and high level of information abstraction in network configuration and monitoring is introduced mainly based on the capabilities of the XML. All of these distributed management approaches demonstrate the variety of management interfaces, which are being used, and will be used, for distributed systems management [1]. XML-based interface definition is one of recent works. The ability to express a huge of document types means it has strong potential in the specification of management information and management interfaces [12]. XML tools can play a useful role in the future configuration and deployment of component based management application. Various related works and some applications for the XML technologies in network management are presented in [18]. The improving and revolutionary approaches for Internet management are explained in terms of SNMP framework, XML and Web services within the Internet Engineering Task Force (IETF), Internet Research Task Force (IRTF), and Internet Architecture Board (IAB) [19].

A policy-based management architecture that is extensible and operates in an active and programmable network was proposed in [9]. The Common Open Policy Service (COPS) is a prominent management interface for Policy Based Network Management (PBNM), but it has failed to gain significant market acceptance [19]. The OSS/J design guidelines a set of modeling patterns that can be applied to the J2EE platform for the specification of EJB-based network and service management building blocks such as trouble ticket, service activation, performance management, fault management and so on. The document of OSS/J defines a set of design patterns for the specification of J2EE/EJB/XML interfaces or telecommunications management [10].

This paper is a succession of background technologies presented in [1], which describes the design and implementation of a management framework based on Java development environment. First, the long-term view of integrated network management framework is described in section 2. I assume that managed domains are globally classified into two ones, telecommunications domain for TMN-based approach and IP domain for SNMP-based approach, respectively. The components of management
framework are presented in section 3. In section 4, an integrated network management framework is designed for adopting a variety of existing and emerging management systems by using the Java development kit, which provides the merits of platform independent fashion. Finally, in section 5, I summarize the paper.

2. The Aims of Network Management Framework

In this section, the aims of network management framework are described. The framework is designed for providing the unified development environment, which can be easily plugged into existing and emerging network management systems, which might have capability of Network Management Layer (NML), Service Management Layer (SML), and Business Management Layer (BML). This management framework must be able to satisfy the network and system management requirements of today’s network environments: scalability, extensibility and interoperability.

The followings are the aims of the framework to develop the multiple domain-based network management system in an integrated manner such as:

- **Network Management Service Portability: Write Once, Run Anywhere.** Technology development is currently constrained by proprietary management services and interfaces. This increases development cost, time to market, and maintenance requirements. With Java technologies, management services are reusable and deliverable to be components for the management service.

- **Integrated Network Management: Any Network.** By delivering the facility to allow management applications and services to manage a variety of networks such as PSTN, packet (e.g. IP or ATM) and wireless networks, the management framework speeds the deployment of network management applications. As demand for services over IP rises, new economies of scale are possible as well as more efficient management and greater integration with IT. The object model structures supported by management protocols are optimized for interoperability and are suited to efficient and evolving application programming and IT support.

- **Secure Management Network Access: By Anyone.** By enabling management applications residing outside the network to directly access network resources and devices to carry out specific actions or functions, a new environment is created for developers and users. The market opportunity for new services is huge when controlled access is provided to the available functionality and intelligence inside the telecommunications networks through management system.

To satisfy these requirements, secure and distributed interconnections with multiple management applications on the network are essential, among other things. An EAI (Enterprise Application Integration) platform is a collection of technologies – middleware such CORBA or message-oriented middleware, adapters/gateways for protocol conversion, data transformers, transaction managers, and work/process flow systems – that allow diverse applications to talk to each other [10].

3. Adapter and Proxy Approaches for Network Management

Basically, management application is implemented as a combination of the management communication service and the manipulation of managed object instances. The basic management service primitive is composed of as set of management operation type and its related definition of managed object instance(s). Management function and service consist of the Java objects for management communication and information in this implementation. The primitive operations of management application are creation, deletion, list, and modification of management information. For instance, the CMIP-based manager can interact with the agent system to exchange CMIP message after creating CMIP associations between two management entities that might negotiate their contexts. The context parameter contains protocol-related information. The services of agent system are determined from the capability of agent system.

I am going to describe management operation in terms of not only propriety-defined script and Java RMI-based middleware approach for CMIP [1, 7]. These approaches are selected for the feasibility of adoption of the existing management systems. The deployed management system can be realized from the merits of platform independent fashion by using the integrated network management framework as shown in Fig 2.

3.1 Script-based Agent Tester

The script-based agent tester is used to perform the management operation to the agent system by defining the CMIP-like scripts in a simple manner [17]. This system can be operated in a stand-alone style by loading the specified script file and remotely loadable from the help of distributed object technology. Before this one performs the management operations, it can establish an association with the agent system by using INMAssociation interface [7]. This can be viewed as the simple CMIP-based manager system that performs the management operation and receives the event notification in a simple descriptive form.

If the policy-based application module is deployed in this framework, the script is used to perform the management operation on the agent system as a lower level one. The script is based on the CMIP primitive and an example of management script for m-Get operation is written as following [17].
The Java RMI is selected for the feasibility of adopting a distributed manner between Java RMI and CMIP systems. A mediation server to support the message transformation in sources and controlled in different bodies. Thus I make the use of management technologies emerging from different sources and controlled in different bodies. Thus I make the mediation server to support the message transformation in a distributed manner between Java RMI and CMIP systems. The Java RMI is selected for the feasibility of adopting existing distributed object technologies.

### 3.2 Java RMI-based Manager Systems

The general convergence in communication market will mean TMN must support the integration of a wider range of management technologies emerging from different sources and controlled in different bodies. Thus I make the mediation server to support the message transformation in a distributed manner between Java RMI and CMIP systems. The Java RMI is selected for the feasibility of adopting existing distributed object technologies.

![Fig. 1. Management Proxy Server with COR and CMIP](image)

The CMIP Over RMI (COR) interface can be used to manage the element and network resources of distributed environments (i.e., Java RMI, RMI-IIOP, CORBA) and the SNMP-based system with the capability of XML-based technologies. Also, those applied distributed environments can be useful to access the SNMP-based agent by adding the adapter module or system. In order to deploy the integrated framework, I have developed CMIP and SNMP interfaces written in Java language [1, 7]. In general, these management interfaces are used to manage the element and network resources of telecommunications and IP network domains respectively. This framework is capable of serving as system boundary interface for both existing and evolving management.

### 4. An Integrated Framework of Network Management

This integrated framework of network management has some capabilities for managing the CMIP-based system through distributed environments (i.e., Java RMI, RMI-IIOP, and CORBA) and the SNMP-based system with the capability of XML-based technologies. Also, those applied distributed environments can be useful to access the SNMP-based agent by adding the adapter module or system. In order to deploy the integrated framework, I have developed CMIP and SNMP interfaces written in Java language [1, 7]. In general, these management interfaces are used to manage the element and network resources of telecommunications and IP network domains respectively. This framework is capable of serving as system boundary interface for both existing and evolving management.

<table>
<thead>
<tr>
<th>Table 1. An Example of Script for CMIP Get Primitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetArgument</td>
</tr>
<tr>
<td>{baseManagedObjectClass globalForm MOCmanagedElementR1, --</td>
</tr>
<tr>
<td>0.0.13.3100.0.3.27</td>
</tr>
<tr>
<td>baseManagedObjectInstance distinguishedName</td>
</tr>
<tr>
<td>{--attributeType 2.9.3.2.7.5,</td>
</tr>
<tr>
<td>attributeValue</td>
</tr>
<tr>
<td>SystemTitle oid 1.1.1.1.1</td>
</tr>
<tr>
<td>},</td>
</tr>
<tr>
<td>-- field &quot;accessControl&quot; is not assigned --synchronization 0, scope level 0, -- baseObject 0,</td>
</tr>
<tr>
<td>firstLevel:1, wholeSubTree:2, filter and {}, --attributeIdList</td>
</tr>
<tr>
<td>{ globalForm 2.9.3.2.7.5,</td>
</tr>
<tr>
<td>globalForm ATsystemTitle,</td>
</tr>
<tr>
<td>globalForm ATadministrativeState</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

COR_DeleteCMIPAssociation, and COR_MakeCMIPAssociation. Also the eligible end systems can be divided into Java User Applet with a scale of Element Management Layer (EML) and BML & SML Management system, which can be interconnected with using distributed environment. The CMIP message queue represents an asynchronous and loosely coupled messaging channel between management system and agent system. The message scheduler is used to control the flow of CMIP messages by adjusting the processing priorities of various messages in a CMIP queue, depending on their types (i.e., alarm event report, normal CMIP response, and others). The Q-Adapter system can be used to absorb the differences of management operations between CMIP-based manager system and SNMP-based agent system.

I have encountered the performance problem of the client with GUI capability and MPS when the event reports (e.g., performance and fault related data) are abruptly sent from the agent systems. The major performance bottleneck of Java development environment using distributed architectures such as Java RMI, CORBA and Java RMI/IIOP can be occurred in the GUI of client-side. Especially, performance problem might be serious when the notifications of event are emitted from agent systems to the client with GUI facility through MPS. The following ones might be useful to lessen the performance bottleneck.

- The application filtering mechanism to apply when the queue size of CMIP message queue in MPS exceeds the maximum queue size.
- The limitation of the number of client’s users, which are concurrently registered in MPS.
- The powerful Event Forward Discriminator (EFD) mechanism in the TMN model caused by the management operation(s).
- The change of granularity of performance related reports. The granularity can be changed either by the operator manually or by MPS automatically.

4. An Integrated Framework of Network Management

This integrated framework of network management has some capabilities for managing the CMIP-based system through distributed environments (i.e., Java RMI, RMI-IIOP, and CORBA) and the SNMP-based system with the capability of XML-based technologies. Also, those applied distributed environments can be useful to access the SNMP-based agent by adding the adapter module or system. In order to deploy the integrated framework, I have developed CMIP and SNMP interfaces written in Java language [1, 7]. In general, these management interfaces are used to manage the element and network resources of telecommunications and IP network domains respectively. This framework is capable of serving as system boundary interface for both existing and evolving management.
systems, which have the capabilities of NML, SML, or BML through their own specific management interfaces. These might be defined in a standard or proprietary manner. Candidate interfaces for underpinning them, in current I consider, might be an open standard set of TMF’s layer management APIs [16] and OSS/J’s Java technology-based APIs [10]. A data-independent approach is also advocated in the TMF in its specification of the network management layer operation. A data-independent layer of function sets that operate on a shared data repository realizes network management processes. The data repository includes data areas such as planning policies, topologies, network configurations, physical inventory, usage data, problem records, and performance measurements. In OSS/J, client/server portability is enhanced by the use of RMI/IIOP, XML/JMS for communication between the client side (i.e., XML/JMS client, EJB-Java client) and the server side (i.e., Message Driven Bean, Stateless Session Bean). The script-based or SNMP-based translation approach can be easily accomplished by the XML technology. From the perspectives of PBNM, SNMP is an existing management protocol to achieve the distribution of policies to devices and the configuration of them. In the case of routers and networking policies, the preferred protocol to use is SNMP [14].

Fig. 2 indicates an integrated network management framework which is composed of management application, MPS, and XML-based system. Java Plug-in software enables web page authors to direct Java applets or JavaBeans components on their intranet web pages to run using Java Runtime Environment (JRE), instead of the browser's default runtime. Web pages are modified to support the plug-in enable users to run Java-based applets in Internet Explorer or Navigator. The Java Plug-in applet in this architecture can perform the management operations via MPS in direct or in indirect through the management system with the capabilities of BML and NML. The basic functions of MPS are divided into the maintenance of ACSE services, the translation of normal management operation and notification, client management, the initial configuration of system, the maintenance of agent registration and the collection of system information. A distributed client application can be easily built and enable network management center operators to be located anywhere (e.g., at home) instead of in a corporate network management center setting with the help of security policy (e.g. signed applet). While the Java Plug-in facilities smooth and secure execution within the confines of the web browsers, Java Web Start provides a way to securely execute programs outside the browser with the help of the Java Network Launching Protocol (JNLP). Not only should the network be open for those wishing to create new services, it also needs to be accessible to every authorized management system in order to allow service customization and “instant” provisioning of services.

The management applications or systems with the capabilities of BML, SML, and NML can perform various management activities by the support of management framework. For instance, policy-based QoS control allows network operators to configure their network devices easily. It provides a high-level view of the network devices and allows the automated translation of business policies to suitable information for configuring network devices [15]. A management system with the capability of EML might be also easily embodied with this management framework in a platform independent manner.

Web services depend on the ability of parties to communicate with each other even if they are using different information systems. I also take the Java XML technology [11] to provide the concept of distributed management objects. The component technologies such as Java Servlets, JavaServer Page, and JavaServer Pages Standard Tag Library are used in developing the presentation layer of a Web application. Web services use XML as an interface to define management functions that are loosely coupled and dynamically bound. By using Simple Object Access Protocol (SOAP), a specification for lightweight Remote Procedure Calls (RPC) using XML documents and management applications can be combined independently of specific platforms. Java API for XML
Processing (JAXP) processes XML documents using various parsers. Java Architecture for XML Binding (JAXB) processes XML documents using schema-derived JavaBeans component classes. The following one is used for representing the XML schema definition for SNMP messages, which can be applied for the JAXB-SNMP translator.

Table 2. XML Schema Definition of SNMP

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="xmSNMPv3MessageType" type="XmlSNMPv3MessageType"/>
  <xs:element name="xmSNMPMessageType" type="XmlSNMPMessageType"/>
  <xs:complexType name="XmlSNMPV3PDU" type="XmlSNMPV3PDU"/>
  <xs:complexType name="XmlSNMPV3PDU" type="XmlSNMPV3PDU"/>
  <xs:choice>
    <xs:complexType name="XmlSNMPV3PDU">
      <xs:element name="get-request" type="XmlSNMPv3PDU"/>
      <xs:element name="get-next-request" type="XmlSNMPv3PDU"/>
      <xs:element name="get-bulk-request" type="XmlSNMPv3PDU"/>
      <xs:element name="inform-request" type="XmlSNMPv3PDU"/>
      <xs:element name="response" type="XmlSNMPv3PDU"/>
      <xs:element name="set-request" type="XmlSNMPv3PDU"/>
      <xs:element name="inform-request" type="XmlSNMPv3PDU"/>
      <xs:element name="trap" type="XmlSNMPTrapPDU"/>
      <xs:element name="snmpV2-trap" type="XmlSNMPv2TrapPDU"/>
      <xs:element name="report" type="XmlSNMPv2TrapPDU"/>
    </xs:complexType>
  </xs:choice>
</xs:schema>
```

JAXB provides a fast and convenient way to bind an XML schema to a representation in Java code, making it easy for Java developers to incorporate XML data and processing functions in Java application without having to know much about XML itself. The JAXB technology provides methods for unmarshaling an XML instance document into a content tree of Java objects, and then marshaling the content tree back into an XML document. Unmarshaling optionally involves validation of the source XML document before generating the content tree. If management application modifies the content tree, it can also use the validate operation to validate the changes before marshaling the content back to an XML document. JAX-RPC and SOAP with Attachments API for Java (SAAJ) are both based on SOAP. A JAX-RPC runtime system converts the client’s remote method call into a SOAP message and sends it to the server as an HTTP request. On the server side, the JAX-RPC runtime system receives the request, translates the SOAP message into a method call, and invokes it. The root element of this XML message is an <Envelope> element. The SOAP-ENV represents the name space for SOAP as defined in the W3C specification. The <Envelope> element consists of a <Header> element optionally and a <Body> element. The SOAP message has the <Body> element for the instance of SNMP message and <Header> element for the SNMP parameters (e.g., the address of agent, UDP or TCP port, time-out, user, context and so on). After the Web service has processed the request, the runtime system goes through a similar set of steps to return the result to the client. In addition, <Sign> element, which is the child of the <Body> element, might be provided for the authentication of security. Java API for XML Registries (JAXR) provides a standard way to access business registries and share information [11].

A Web service can make itself available to potential clients by describing itself in a Web Service Description Language (WSDL) document. With JAX-RPC, a client written in a language other than the Java programming language can access a Web service developed and deployed on the Java platform. Conversely, a client written in the Java programming language can communicate with a service that was developed and deployed using some other platform. A Lightweight Directory Access Protocol (LDAP) can be used as a kind of data repository for storing a variety of information such as network topology, policy rules, scripts, configuration information and so on.

5. Conclusions

Telecommunications services and managements requirements in terms of reusability for their quick provision and interoperability to alleviate service interactions management and distribution must be supplied. Because Java provides the capabilities of simplicity, object-oriented concept, reliability, portability and multi-threading, the Java-based distributed object computing has become increasingly popular as more complex products are written using a multi-tier architecture [1]. Therefore, I have made an integrated network management framework for the management systems, which want to inherit the merits of Java technologies in platform independent manner. This proposed framework could give us a fact that the integration of network management between the control elements of traditional and emerging networks is a powerful tool for rapidly deploying and offering new management services in a platform independent fashion and in a steady manner.

In the future, I will realize the liaison with other legacy management systems and enhance the functionality of the management framework by integrating existing and new Java technologies.
References