New Naming Service Architecture for Limited Resource System within CORBA-based Network Management

Joon H. Kwon, Moon S. Jeong, and Jong T. Park
School of Electronic and Electrical Engineering, Kyungpook National University
{jhkwon, msjeong}@ain.knu.ac.kr, park@ee.knu.ac.kr

CORBA 기반 망 장비를 위한 새로운 Naming Service

권 준 협　　정 문 상　　박 종 태
경 북 대 학 교　전 자 공 학 과

Abstract

Nowadays, efforts are in progress for the standardization of CORBA-based telecommunication network management framework. To implement a network management system based on the framework completely, CORBA ORB and some of CORBA services should be installed in the element. And then, there must be the naming tree, which corresponds to the containment relationships between components in the network element. If we use conventional OMG naming service to form the naming tree, all MOs, a software fragment that corresponds to each component in a system, should be instantiated. However, the network element is usually a kind of limited resource system, which cannot provide sufficient resources for applications run on it. Hence, instantiating all MOs can cause problems for that kind of system. This paper presents Smart Naming Service architecture as a solution to the problem.

1. Introduction

To date, Telecommunication Management Network (TMN)[1] architecture has been applied for the management of telecommunication network. The initial TMN interface specification for network management was developed using Common Management Information Protocol (CMIP) as the protocol. Since CMIP is not suitable for developing applications in distributed environment, it has been hard to implement the higher layers of TMN architecture [1].

Common Object Request Broker Architecture (CORBA) [2], developed by OMG, is the most powerful technology for building distributed applications. With CORBA and Common Object Service (COS)[3], the higher layer of TMN architecture can be easily implemented and used for more sophisticated telecommunication network management.

A gateway approach was the first step for adoption CORBA into TMN system [4]. This approach has already standardized by the Joint Inter-Domain Management (JIDM) task force [5].

Now, there are several working groups for pure CORBA-based telecommunication network management [6] [7]. One prominent group among all research groups is working group T1M1.5 of American National Standard Institute (ANSI). To construct the complete Network Management System (NMS) based on the framework of T1M1.5, network elements should be equipped with ORB and some of CORBA services. Upon these bases, all CORBA-based MOs that correspond to each physical and logical component in a network element should exist, and network element also should provide naming tree, which corresponds to the containment relationship of the network element.

With conventional OMG naming service [3], all MOs in a network element must be instantiated to form the naming tree of the network element. However, instantiating all MOs are not suitable for network elements, since network elements are limited resource system. [8], which has a limitation in providing system resources to applications run on it. So, there must be an efficient way in using system resources.

This paper introduces new naming service architecture, Smart Naming Service (SNS). SNS does form naming tree, and does not instantiate all MOs. The basic idea of new Naming Service, SNS, is "MO instantiation on demand." To achieve the functionality, SNS does not use MO instance reference. Instead, it uses instantiation information of MO to make a name binding, which maps a name to object instance [3]. SNS instantiate an MO instance when there is a resolve request to the MO.

This paper is organized as follows. In section 2, naming scheme suggested by T1M1.5 is described, and, the problems with conventional OMG naming service are also pointed out in detail. Sections 3 introduces the design requirements for the new Naming Service firstly, and then, designs new Naming Service, i.e. SNS. At the end of section 3, detailed descriptions on newly introduced building blocks and methods are presented. Finally, section 4 ends with a summary.

2. Naming Architecture of T1M1.5 using Conventional OMG Naming Service

T1M1.5 is defining the framework through two documents. The first one covers framework requirements, CORBA COS usage requirements, information modeling guidelines, and Interface Definition Language (IDL) style conventions. The second document specifies a generic network level information model to be used in telecommunications network management based on CORBA. It defined in IDL a set of generic interfaces and constants. These documents have been and are still being revised now. In this section, the naming scheme by T1M1.5 [5] will be is presented. Then, the problems with conventional OMG naming service will also be pointed out

2.1. Naming Scheme Suggested by T1M1.5

Fig.1 gives an example of naming scheme according to the requirements presented by T1M1.5 using conventional OMG naming service. One real component is mapped into a combination of MO instance with a naming context object instance [5]; MO instance for representing its real component, and naming context object instance for forming naming tree according to the containment relationship of real components in the system. The MO instance has name binding [3] in corresponding naming context using a name with the string, "Object", in "ID" field. Naming contexts in the figure are depicted as dashed circle, and actual MO instances are drawn as solid circle..

In the figure, the naming tree of ATM switch 1 is connected to the naming tree of ATM network 1.

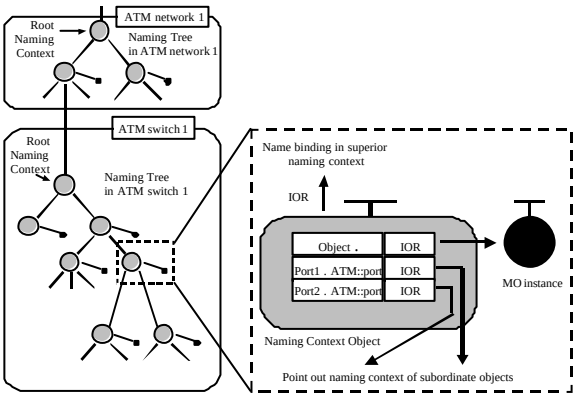

Figure 1. Naming Scheme of T1M1.5

The requirements [5] on using the CORBA naming service to represent the containment relationships among MO instances are followed.

1. Every MO must have one associated name binding representing its position in the containment tree.

2. Each managed object must actually have a corresponding naming context, since a simple name binding cannot identify an object and also containment objects.
3. The "ID" field of a name binding for a naming context representing a managed object will be application-dependent, and it may actually have semantic value beyond uniquely identifying a managed object, for a particular class of objects.
4. The "KIND" field of a name binding for a naming context representing a managed object shall be the scoped class name of the managed object
5. A specially-named binding in each such context will bind the ID value "Object" with a reference to the actual MO.
6. A managed system must provide a local administrative procedure for assigning a CORBA name to its root naming context.

Any naming tree formed in equipment, which participates in the framework, must satisfy the requirements above.

2.2. Resource Shortage Problem

The Interoperable Object Reference (IOR) [2] is a standard reference structure which pointing out an object instance connected to ORB network. OMG naming service uses an object instance IOR to build a name binding for the object. Hence, to form naming tree in network equipment, all MO instances must be instantiated occupying some of system resource for each.

Fig.2 is simplified diagram of the diagram in dashed box in Fig.1. This diagram shows the procedure of resolving MO instance after accessing to the naming context object, which contains the name binding for the actual MO instance.

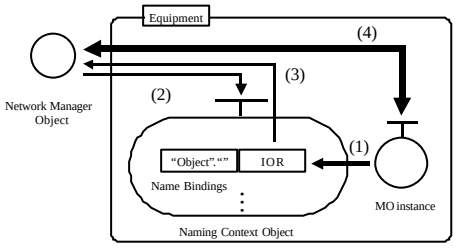

Figure 2. Resolving a MO with OMG Naming Service

The internal resolving procedure with OMG Naming Service is as follows. First of all, (1) to form a naming tree in equipment, every MO instance of the equipment must be created as well as naming contexts corresponding to each MO. (2) When a network manager requires the naming context to return a IOR of actual MO instance, (3) the naming context find the name binding correspond to the name, and then, return IOR of the name binding to the requester. (4) Once network manager have IOR of an MO instance, the manager can access to the instance and perform network management operation.

But, since the network equipment is limited resource system, instantiating all MO is not suitable for that kind of system if the number of MO to be instantiated is remarkably big. With the restricted system resource, the size of MO instance must be reduced. Codes for exception handling or some codes for notification may be omitted for saving the resource. It will make the overall management system defective. Moreover, all MO instances are not used at the same time. This means that the valuable system resource is spent to maintain the instances that are not participating in network management operations. This is Resource Shortage Problem.

3. Smart Naming Service (SNS)

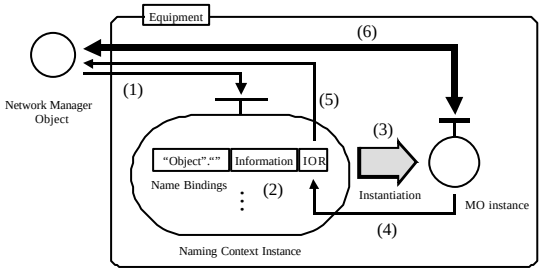

Figure 3. The Basic Architecture of SNS

This section introduces the SNS as a solution to the problem. SNS doesn't change or delete any functionality of OMG Naming Service. It just adds one interface, one data structure, and one new method to the IDL definition of conventional OMG Naming Service.

Fig.3 shows resolving operation of new Naming Service, SNS. The brief procedure of resolve operation is like this; (1) Once a manager object asks a naming context object of SNS to return the IOR of the MO, (2) the naming context object search name binding, (3) and then instantiate the MO when there is not MO instance. (4) After instantiation, the IOR of new instance is held by the naming context for some reason. The reason will be described later in this paper. (5) And then the naming context object returns the IOR of the new instance to the manager object. (6) With the IOR, network manager access to the MO instance and perform network management operations.

3.1. Requirements for New Naming Service

Req.1. Even if the SNS is applied to equipment, network manager object should be able to access to an object in the equipment without feeling any changes.

Req.2. Naming context of SNS can make a name binding using information for instantiating an object instead of IOR.

Req.3. SNS should be able to instantiate an object using the information in the object's name binding.

Req.4. The new MO instance and the real component must be synchronized before returning its IOR to the requester. And, furthermore, since the MO instance represents the actual physical component, any changes in MO instance should be reflected to the entity, and vice versa.

Req.5. If the MO instance of interest is already exists, naming context object must return the IOR of existing MO instance. This prevents the multiplication of MO instance.

Req.6. Any instance, which is not used for a long of time, must be deleted for other MO instances, i.e. garbage control should be performed.

Req.7. New naming service should support the way of making name binding in a traditional way, since there exists MO instances that should be instantiated all the time

3.2. Architecture of New Naming Service, SNS

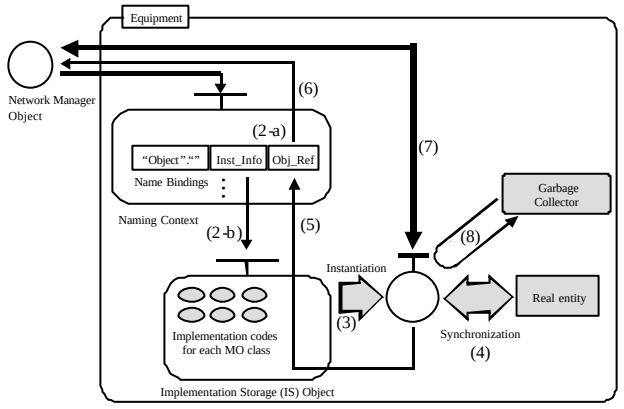

Figure 4. The overall architecture of SNS

Fig.4 shows the overall architecture of SNS satisfying the requirements listed above. The situation here is also same as Fig.2.

(1) When a manager object asks naming context to return the IOR of the MO. (2-a) The naming context first locate the name binding and check the existence of the instance of the MO while checking Obj_Ref field, which contains the IOR of MO instance, in the name binding. If Obj_Ref contains the object's IOR, naming context object returns the IOR. (2-b) If not, that is to say, IOR field of name binding is NULL, the naming context pass the instantiation information in Inst_Info field, which contains a structure which contains the ID of real entity and information related to instantiate the MO, to Implementation Storage (IS). (3) With the Inst_Info, IS instantiate the MO and (4) synchronize the instance and real element identified by Inst_Info. (5) After the synchronization, the IOR of new instance is passed to naming context. The naming context object set the Obj_Ref field of the name binding with the IOR. Setting Obj_Ref field with IOR prevents multiplication of MO instance. (6) And then, after all, the naming context object returns the IOR to the manager object. (7) MO instance perform operations with the IOR. (8) When IS instantiates an MO, it marks time

stamp on the instance. There is garbage collector that deletes instances which is not used for a period of time. Deleting an instance involves setting Obj_Ref field of corresponding name binding to NULL. This enables Garbage Collector (GC) to save system resource while deleting instances not used for a long time.

3.2.1 IDL definition of SNS

Fig.5 shows the IDL definition of SNS. SNS just adds one data structure, one interface, and one method to the IDL definition of OMG naming service.

```
#include "CosNaming.idl"

module SNS
{
  typedef CosNaming::Istring Istring;
  typedef CosNaming::Name Name;
  typedef CosNaming::BindingList BindingList;
  interface BindingIterator : CosNaming::BindingIterator;

  struct Inst_Info{
    string        MO_Interface_Name;                    A
    unsigned long  Real_Resource_ID;
  };

  interface IS{
    Object inst(in Inst_Info order)
      raises(
            NotFound, CannotProceed,
            InvalidName, AlreadyBound
      );
  };

  interface NamingContext : CosNaming::NamingContext{
    void bind_sns(in Name n, in Inst_Info i)
      raises(
            NotFound, CannotProceed,
            InvalidName, AlreadyBound);
  };                                                     B

};
```

Figure 5. IDL definition of SNS

3.2.2 Implementation Storage (IS)

IS is an object, which contains implementations of MOs and is able to create MO instances. With the instantiation information, Inst_Info, IS creates MO instances whenever it needed. The struct, Inst_Info, contains the information; the ID of physical entity in Real_Resource_ID and interface name of MO in MO_Interface_Name. The method, inst(…), receives parameter, `order`, as a parameter of instantiation operation.

With `order.MO_Interface_Name`, IS decides which type of MO instance should be created. IS has implementation codes for all MO classes and is able to create MO instance. Once the MO class is decided, IS can easily create MO instance.

With `order.Real_Resource_ID`, IS maps the MO instance onto real entity identified. At instance creation time, IS performs synchronization the instance with the real entity initially. After initial synchronization, IS object return the IOR of the MO instance. Once initial synchronization is performed, any changes in MO instance should be reflected to the entity, and vice versa

3.2.3 Name Bindings in SNS

There's no IDL level definition for name binding structure. This means that the name binding structure is defined at development time. To support the functionality described above, the structure of name binding in the implementation of SNS should contain name, Inst_Info, and IOR.

3.2.4 Naming Context of SNS

Naming context of SNS should be able to make a name binding with Inst_Info instead of IOR of an object. As depicted in Fig.5, the naming context of SNS inherits from that of OMG naming service for back ward compatibility. Just one method is added.

The method, bind_sns(…), receives parameter i instead of IOR. It creates name binding with NULL value in Obj_Ref field. While the original method, bind, will be implemented to make name binding NULL value in Inst_Info field.

In resolve operation, new naming context object locate the name binding corresponds to name received, and check the existence of instance first verifying whether IOR field of the name binding is NULL or not. If there is an instance of MO of interest, i.e. the value in Obj_Ref is not NULL, the naming context object should return the IOR to the requester. Otherwise, it ask IS to create new MO instance giving the parameter, i, directly. After the IOR of newly created MO instance is returned, the naming context should set Obj_Ref field of the name binding before returning IOR to the requester. This prevents multiplication of instance of an MO

3.2.5 Garbage Collector (GC)

Garbage Collector (GC) is one function block in SNS. The word, "garbage", means an MO instance that is not used for some length of time. GC saves system resources deleting the garbage for the other MO instances. At MO instantiation time, IS mark a time-stamp on the new instance. The time-stamp is to be updated every time there's an access from manager object to the MO instance. GC periodically checks the time-stamp of each MO instance while comparing the time in time-stamp with current time. If the difference between the time stamped and current time exceeds the time-length, which is determined by network administrator, GC regards the instance as garbage, and then, delete the instance. GC also set Obj_Ref field in the name binding of the instance to NULL. This prevents MO instance multiplication.

4. Conclusion

OMG naming service is not suitable for network element that is a kind of limited resource system, since all MOs are must be instantiated. To answer the problem, this paper introduced new naming service, SNS, which instantiating MOs when needed.

While saving system resources assigned to MO instances, which is not used, SNS provides more system resources to MO instances being used at that time. Though, new functional object, IS, and new functional block, Garbage Collector, are added to OMG naming service, the overall performance will be improved when the equipment contains a lot of components of same type.

However, SNS has a defect in response time because of the time required in instantiating MO and synchronizing the new instance and real component, while, otherwise, conventional OMG naming service just returns an IOR of an instance. This may be able to be solved providing policies on deciding which instance should be instantiated all the time, and the time limit according to MO type.

5. Reference

[1] ITU-T Recommendation M.3010, *Principles for a Telecommunications management network*, May 1996.

[2] The Object Management Group (OMG), "The Common Object Request Broker: Architecture and Specification," Revision 2.3, June 1999.

[3] The Object Management Group (OMG), "CORBA Services: Common Object Services Specification," Updated version, December 1998.

[4] Jong T. Park, Moon S. Jeong, and Seong B. Kim, "A Platform Architecture for the Integration of CORBA Technology within TMN Framework," IEICE Transactions on Communications, Vol.E82-B, No.11, pp.1770 ~ 1779, November 1999.

[5] The Object Management Group (OMG), "JIDM Interaction Translation," Edition 4.31, OMG TC Document telecom/98-10-10, October 1998.

[6] S. Mazumdar, S. Brady, and D. Levine, "Design of Protocol Independent Management agent to Support SNMP and CMIP Queries," Third International Symposium on Integrated Network Management, San Francisco, USA, 1993.

[7] George Pavlou, "A Novel Approach for Mapping the OSI-SM/TMN Model to ODP/OMG CORBA," International Symposium on Integrated Network Management, Boston, USA, May 1999.

[8] The working group T1M1.5, "CORBA Generic Network and NE Level Information Model," T1M1.5/99-0306: Working Document for Draft Standard ANSI T1.2xx-1999, December, 1999.

[9] The Object Management Group (OMG), "minimumCORBA," Joint Revised Submission, OMG TC Document orbos/98-08-04, August, 1998.

[10] Michi Henning and Steve Vinoski, "Advanced CORBA

Programming with C++,” Addison-Wesley, January 1999.