SNMP 기반 NMS에서 네트웍 관리를 위한 NE MIB의 자동 Import System 설계

강광석, 김영진, 박상대, 정성현, 조종신,
㈜ 현대 시스콤
Phone : +82-31-630-2756, Fax : +82-31-639-6941,
Mail : (sdpurple, forevercho, fpassion, ksk1015, sunghyoun)@hysyscomm.com

Automatic import system of NE MIB for network management at NMS based on SNMP

Kwang-Seok Kang, Young-Jin Kim, Sang-Dae Park, Sung-Hyoun Jung, Jong-Shin Cho
Hyundai SysComm Inc.

Abstract

SNMP is generally used protocol for network management. SNMP need MIB for exchanging management information. But, method for sharing MIB is not automatic. General MIB sharing architecture is done by network management operator by manually. In this paper we describe exchanging MIB automatically using meta-MIB. We will formalize SMI as MIB structure. This meta MIB represent information about MIB of NE Agent. Meta MIB has Information of object in the MIB, Structure and index information for columnar object, meaning of subtype at INTEGER syntax. Also, trap related information is represented too. In this paper, MIB information is represented as instance of meta MIB. This architecture will provide a method for automatic MIB exchanging

1. 서론

망이 점점 고도화 되고 복잡해질수록 그 운용과 유지 또한 복잡성이 증가하였다. 이렇게 망이 복잡해질수록 망 관리가 체계화될 필요성이 제기되었고, 인터넷 망관리 프로토콜로서 SNMP(Simple Network Management Protocol)가 등장하였다.

SNMP 망 관리 구조 모델은 전산망 관리 장비의 망 관리 시스템(NMS : Network Management System) 응용과 전산망 요소(NE : Network Element)의 수행자(Agent) 사이에서 관리 정보 교환에 의해 수행된다.

위와 같은 NMS와 NE Agent간의 관리 정보 항목에 대한 처리 과정을 수행하기 위해, NMS와 NE Agent는 관리 항목을 기술한 SMI(Structure of Management Information)규격의 MIB(Management Information Base) 정의 문서를 오프라인을 통해 공유한다. 그러나, NE Agent와 NMS간에 MIB 공유가 안되면 NMS에서는 해당 NE에 대한 망 관리 기능을 수행하지 못한다는 제약점이 생기게 된다.

본 논문에서는 NMS에서 NE Agent로부터 NE의 MIB을 온라인상으로 제공받아 유연성 있는 망 관리 기능을 수행할 수 있도록 망 관리 기능 개선에 초점을 둔 시스템 설계에 관하여 논한다.

2. SNMP를 이용한 망관리

2.1. SNMP 개요

SNMP는 인터넷에서 망관리를 위하여 표준화된 Protocol로써 Version 3까지 제안되어 있다. SNMP를 이용한 망 관리의 일반적 구조는 하나의 NMS와 하나 이상의 NE Agent로 구성된다. NE Agent는 망 관리의 대상이 되는 다양한 망 요소들에 내재되어 망 요소들이 생성하는 정보에 직접적으로 접근하며, 그 정보를 SNMP에 맞는 형식으로 NMS로 전송하게 된다. 인터넷 망관리 표준의 구성 요소인 SNMP, SMI, MIB은 기본적으로 ASN.1(Abstract Syntax Notation.1)의 형식으로 정의되며 그 Encoding과 Decoding은 ASN.1의 방식인 BER(Basic Encoding Rule)을 따른다.[1]

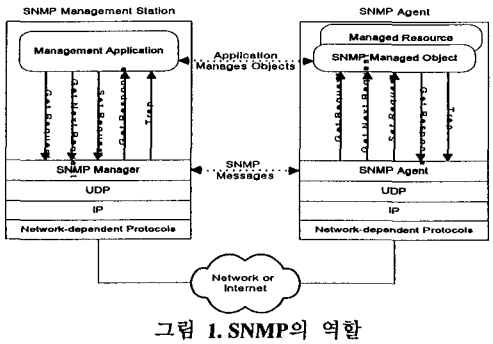

그림 1. SNMP의 역할

SNMP는 Get-Request, GetNext-Request, Set-Request, Response, Trap등의 동작을 지원하며 이러한 동작들을 통하여 Manager-Agent 상호간에 관리정보를 교환한다.

2.2. SNMP에 있어서 MIB의 역할

MIB은 SNMP를 사용하여 NE Agent와 NMS 사이에 교환되는 관리 정보를 말한다. 그렇기 때문에 MIB은 NMS와 NE Agent 양단간에 그 의미와 형식이 사전에 공유되어 있어야 한다는 것이 전제된다. MIB은 SMI의 형식 안에서 정의된다. SMI는 MIB을 정의하기 위한 일종의 스키마라고 볼 수 있다. SMI로 정의된 MIB은 관리 객체들의 집합이라고 볼 수 있으며, 각각의 관리 객체는 그 유일한 식별자와 특성을 가진다. 각 관리 객체의 식별자는 전역적으로 유일한 값을 가지며,

관리객체의 특성은 SMI에 정의된 형식을 기준으로 구성되게 된다. 아래의 그림 2는 SNMP를 사용한 일반적인 구조의 MIB 공유구조를 나타낸다. MIB은 NMS와 NE Agent에 각각 제공되어야 하며 그 작업은 통상적으로 운용자에 의해 이루어진다.

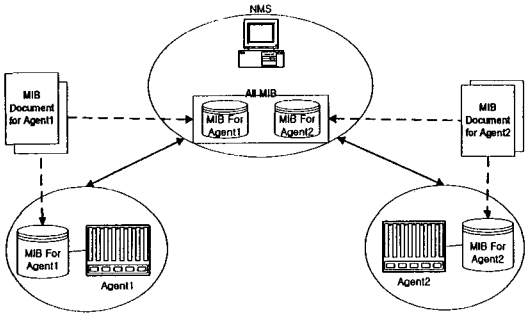

그림 2. 일반적인 MIB 공유 구조

2.3. Meta MIB

NMS와 NE Agent 사이에서 통신하는 형식은 SNMP를 따르지만 교환되는 관리 정보인 MIB은 사전에 상호간 공유가 되어 있어야 한다는 제약사항이 있다. 만약 MIB이 공유되어 있지 않다면 NMS는 NE Agent로부터 어떤 종류의 객체 정보를 가져와야 할지를 알 수가 없으며, 특정정보를 가져오더라도 그 의미를 이해할 수가 없게 된다. 이러한 제약사항을 극복하기 위해서는 NE Agent가 가지고 있는 MIB에 대하여 그 정보를 NMS로 전송하여야 한다. NE-Agent에서 NMS로 전송되어야 하는 정보는 다음과 같다.

- NMS가 NE Agent로 관리 정보를 요청하기 위해 필수적으로 필요한 정보
- NE Agent에서 NMS로 보낸 정보에 대해 NMS가 그 의미 요소들을 분석해 내기 위해 필요한 정보.

이러한 정보들을 MIB의 형태로 구현함으로써 MIB의 Meta정보를 구성할 수 있고 그 Meta 정보를 NMS에서 SNMP로 요청함으로써 사전에 MIB이 공유되지 않더라도, NE Agent가 가지고 있는 MIB과 그 제공 정보에 대하여 알 수 있게 된다. 아래의 그림 3은 Meta MIB을 사용하여 NE Agent에서 NMS로 NE Agent의 MIB 정보를 제공하는 것을 도식화 한 것이다.

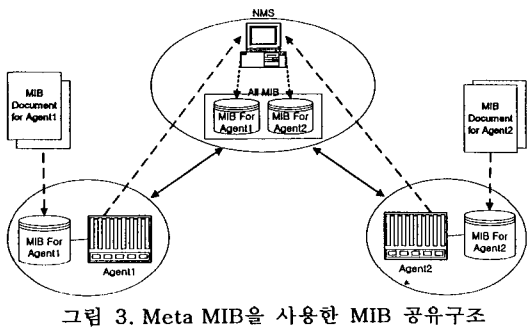

그림 3. Meta MIB을 사용한 MIB 공유구조

3. NE Agent의 MIB 정보를 Meta MIB으로의 표현

3.1. SMI로 표현되는 MIB의 정형화 구조

MIB 정보를 표현하기 위한 Meta MIB으로 구조화 시키기 위해서는 NE Agent MIB을 정의하기 위한 기본 구조인 SMI를 Meta MIB으로 표현할 수 있도록 그 구조를 정형화 시켜야 한다. 이러한 정형화 과정을 위해 SMI의 구조 자체를 일반적인 MIB의 형태로 표현하고 SMI를 정형화한 MIB의 인스턴스로 NE Agent MIB의 정보들을 표현함으로써 Meta MIB을 구성한다. 이를 위해 NE Agent MIB 객체들의 공통 특성들을 표현하기 위한 baseInfoTable을 하나의 테이블로하여 NE Agent MIB 객체들의 공통 특성을 반영하는 Meta MIB 테이블를 구성한다. 그리고 objectInfoTable과 trapInfoTable을 각각 생성함으로써 OBJECT-TYPE과 TRAP-TYPE 객체들을 표현하는 특성들을 반영시킨다. 그리고 테이블의 연관관계와 인덱스를 표현하기 위해 sequenceInfoTable을 생성하고, INTEGER syntax의 의미를 반영하기 위한 syntaxIntegerTable을 생성한다.

3.2. MIB 정보의 Meta MIB으로의 Mapping

SMI구조로 정의된 MIB 정보를 표현하기 위한 Meta MIB의 객체중 MIB 객체가 표현하는 공통 부분에 해당하는 baseInfoTable에 대한 객체를 살펴보면 다음과 같이 매핑 되어 설계된다. baseInfoTable의 baseInfoIndex는 테이블의 인스턴스를 구분하는 인덱스로 사용된다. baseInfoTable의 moduleName 은 NE Agent MIB의 DEFINITOINS과 매핑된다. moduleName으로 NE Agent에서 사용되는 MIB의 종류를 알 수 있다. baseInfoTable의 objectName은 NE Agent MIB에 나타나는 모든 관리 객체들의 이름과 매핑된다. baseInfoTable의 objectType은 NE Agent MIB의 관리 객체들의 타입을 표현하는 것으로 여기에 해당하는 것은 OBJECT IDENTIFIER, OBJECT-TYPE, TRAP-TYPE, NOTIFICATION-TYPE 등이 있다. baseInfoTable의 objectType은 Abstract Syntax로 INTEGER를 사용하여 object-identifier(1), object-type(2), trap-type(3), notification-type(4)으로 매핑한다. baseInfoTable의 objectID는 NE Agent MIB의 관리 객체 OID와 매핑되며, Abstract Syntax로 OBJECT IDENTIFIER를 사용한다. 마지막으로 baseInfoTable의 description은 MIB 관리 객체의 DESCRIPTION 부분과 매핑된다.

3.3. OBJECT-TYPE 객체의 Meta MIB Mapping

NE Agent MIB에서 관리 객체가 OBJECT-TYPE으로 선언되는 객체에 대하여 Meta MIB으로 매핑하는 방법을 기술하면 다음과 같다. Meta MIB안에서 objectInfoTable은 OBJECT-TYPE으로 표현되는 객체에 대한 특성을 반영할 수 있도록 설계되었다. ObjectInfoTable의 objectInfoEntry는 인덱스로 objectInfoIndex와 baseInfoIndex를 사용한다. objectInfoIndex는 테이블의 인스턴스를 구분하는 인덱스이고 baseInfoIndex는 baseInfoTable에서 objectType객체의 값이 object-type인 경우에 해당하는 인덱스의 값을 나타낸다. objectInfoTable의 objectBaseSyntax는 NE Agent MIB의 SYNTAX를 표현한 것으로 ASN.1의 Primitive Type인 INTEGER, OCTET STRING, OBJECT IDENTIFIER와

Constructor Type인 SEQUENCE, SEQUENCE OF가 해당되며, Abstract Syntax로 INTEGER를 사용하여 integer(1), octet_string(2), object_identifier(3), sequence_of(4), sequence(5)으로 매핑한다[표1]. objectInfoTable의 objectComposedSyntax는 NE Agent MIB의 SYNTAX가 재정의 된 경우를 매핑한다. 예를 들면 RFC1213의 특정 객체의 SYNTAX가 DisplayString 경우 OCTET STRING이 재정의 된 것이다. 이와 같이 재정의된 SYNTAX에 대해 매핑하며 재정의 되지 않은 SYNTAX는 objectBaseSyntax와 동일 하게 매핑한다.[4] objectComposedSyntax는 Abstract Syntax로 DisplayString을 사용한다. objectInfoTable의 objectStatus는 NE Agent MIB 객체의 STATUS 종류를 나타내는 것으로 Abstract Syntax로 INTEGER를 사용하여 mandatory(1), optional(2), obsolete(3)으로 매핑한다. objectInfoTable의 objectAccess는 NE Agent MIB 객체의 ACCESS 종류를 나타내는 것으로 Abstract Syntax로 INTEGER를 사용하여 read-only(1), read-write(2), write-only(3), not-accessible(4)로 매핑한다.

	NE MIB Syntax	Meta MIB 매핑
Primitive Type	INTEGER OCTET STRING OBJECT IDENTIFIER	integer(1) octet_string(2) object_identifier(3)
Constructor Type	SEQUENCE SEQUENCE OF	sequence_of(4) sequence(5)

표 1. SYNTAX의 Meta MIB으로의 매핑

Meta MIB의 syntaxIntegerTable은 NE Agent MIB의 SYNTAX가 INTEGER일때 subtype을 매핑하기 위해 설계되었다. 설계된 객체를 살펴보면 다음과 같다. syntaxIntegerEntry는 인덱스로 syntaxIntegerInfoIndex와 baseInfoIndex를 사용한다. syntaxIntegerInfoIndex는 테이블의 인스턴스를 구분하는 인덱스이고, baseInfoIndex는 baseInfoTable에서 objectType객체의 값이 object-type이면서 objectInfoTable에서 objectBaseSyntax의 값이 integer인 객체의 인덱스 값을 표현한다. syntaxIntegerValue는 NE Agent MIB의 SYNTAX INTEGER의 subtype과 매핑된다. syntaxIntegerValueString은 · NE Agent MIB의 SYNTAX NTEGER의 subtype의 표현 문자열과 일대일 매핑된다.

Meta MIB의 sequenceInfoTable은 NE Agent MIB의 SYNTAX가 SEQUENCE OF인 테이블의 경우를 매핑하기 위해 설계되었다. 인덱스로 sequenceIndex와 baseInfoIndex를 사용한다. sequenceIndex는 인스턴스를 구분하는 인덱스이다. baseInfoIndex는 objectInfoTable에서 objectBaseSyntax의 값이 sequence-of인 baseInfoIndex의 값을 나타낸다.[그림4] sequenceIndexValue는 NE Agent MIB의 테이블 인덱스와 매핑되는 것으로 값은 baseInfoIndex중 NE Agent MIB의 테이블 인덱스를 나타내는 값과 일치한다. sequenceEntryInfo는 NE Agent MIB의 엔트리 인스턴스와 매핑되는 것으로 값은 baseInfoIndex중 NE Agent MIB의 엔트리 인스턴스를 나타내는 값과 일치한다.

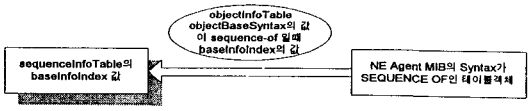

그림 4. sequenceInfoTable의 baseInfoIndex 매핑

3.4. TRAP-TYPE 객체의 Meta MIB Mapping

NE Agent MIB의 Trap 객체에 대한 Meta MIB으로의 매핑 설계 방법을 기술하면 다음과 같다.

Trap 객체 처리를 위해 trapInfoTable을 설계하였다. trapInfoTable에 기술된 인덱스는 trapInfoIndex와 baseInfoIndex이다. trapInfoIndex는 인스턴스를 구분하는 인덱스이고, baseInfoIndex는 baseInfoTable의 객체인 objectType의 값이 trap-type 이거나 notification-type인 경우의 인덱스 값을 나타낸다. trapInfoTable의 trapEnterprise 객체는 NE Agent MIB의 TRAP-TYPE 객체의 ENTERPRISE 값과 매핑되며, TRAP-TYPE이 아니라 NOTIFICATION-TYPE일 경우는 NULL 값으로 매핑한다. trapInfoTable의 trapVariable 객체는 NE Agent MIB에서 TRAP-TYPE 객체의 VARIABLES 값과 매핑되며,. NOTIFICATION-TYPE일 경우는 OBJECTS의 값을 매핑 한다.

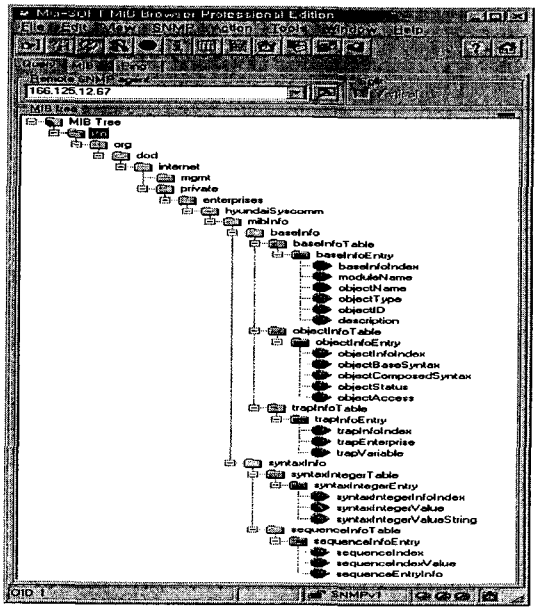

그림 5. 컴파일된 Meta MIB Tree 구조

4. NMS에서 Meta MIB의 필요 정보 추출

NMS는 NE Agent로부터 정형화된 Meta MIB 정보를 SNMP를 통하여 읽어온다. 읽어온 Meta MIB의 결과를 NMS에서는 NE Agent에 대한 MIB을 구성하기 위하여 MIB Data로 재구성하는 작업을 수행한다. 이때 OBJECT-TYPE 객체 정보를 재구성하는 작업과 TRAP-TYPE 객체 정보를 구성하는 작업으로 구분하여 생각할 수 있다.

4.1. OBJECT-TYPE 객체의 정보 추출

OBJECT-TYPE에 대한 정보는 NMS측에서 NE Agent의 MIB 정보나, MIB 구조를 표현하기 위해 필요한 정보들이다. 이러한 정보를 구축하기 위해서 Object Identifier, Object Name, Object Type, Description, Syntax, Access, Status, Scalar Object인지 Columnar Object인지에 대한 구분이 필요하게 된다.

OBJECT-TYPE 객체는 크게 Scalar Object와 Columnar Object로 구분되어진다. Scalar Object는 객체에 대한 인스턴스가 오로지 한 개만 존재하는 객체이며 Columnar Object는 해당 객체에 대하여 한 개 이상의 동적인 인스턴스를 가지는 객체이다.

4.1.1. Scalar Object 정보 추출

baseInfoTable과 objectInfoTable내의 객체들은 NMS측에서 사용할 기본적인 정보를 가지고 있다. baseInfoTable에서 Object Identifier, Object Name, Object Type, Description의 정보를 추출하며 objectInfoTable에서는 Syntax, Status, Access에 대한 정보를 추출한다.

objectInfoTable내의 객체인 objectBaseSyntax의 값이 Primitive Type이고 테이블의 구조가 확정된 경우 테이블 내의 엔트리를 제외한 OBJECT-TYPE 객체에 대해서는 Scalar Object로 판단할 수 있다.

4.1.2. Columnar Object 정보 추출

objectInfoTable의 객체인 objectBaseSyntax가 Constructor Type이면서 그 값이 sequence-of인 경우에 대하여 sequeceInfoTable을 참조한다.

Columnar Object에 대한 정보를 추출하기 위해서는 Scalar Object의 기본적인 정보뿐만 아니라 Columnar Object, 즉 테이블이라는 정보를 나타내는 objectInfoTable의 objectBaseSyntax 객체가 sequence-of인 baseInfoIndex의 값은 테이블에 대한 참조값으로서 필요하다. 또한 위의 baseInfoIndex의 값을 참조로 sequenceInfoTable에서 NE Agent MIB의 테이블 인덱스로 표현하는 객체값은 sequneceIndexValue에서 추출해야 한다.

NE Agent MIB내의 테이블 엔트리 객체들을 추출하기 위해, sequenceIndexValue를 추출하기 위해 사용하였던 baseInfoIndex의 값을 참조하여 sequenceEntryInfo값을 추출한다. 이렇게 추출된 sequenceIndexValue와 sequenceEntryInfo의 값은 baseInfoTable에서 baseInfoIndex가 가르키는 객체가 된다.

4.2. TRAP-TYPE 객체의 정보 추출

Meta MIB으로 매핑된 Trap 정보에서 Trap 객체를 재구성하기 위해서는 Trap Version, Enterprise Object Identifier, Trap Variable에 대한 정보가 필요하다. 이러한 정보는 Meta MIB의 baseInfoTable과 trapInfoTable에서 추출한다.

baseInfoTable 객체 중에서 objectType의 값이 trap-type이거나 notification-type인 경우, 그에 해당하는 baseInfoIndex값을 참조하여 trapInfoTable의 trapEnterprise에서 Enterprise Object Identifier를 추출하고 trapVariable의 값을 추출한다. trapVariable에서 추출된 값은 objectType이 notification-type일 경우 NE Agent MIB에서 정의된 "OBJECTS" 값이 된다.

4.3. MIB Data의 구조

MIB Data 구조는 Meta MIB에서 얻어낸 MIB에 대한 정보를 가지고 Object Identifier에 따라서 순차 정렬을 하여 사용하게 된다. 순차적으로 정렬하게 되면 NE Agent에 대한 MIB 정보의 구조화가 이루어지며 NMS측에서 SNMP Operation을 할 경우 존재하지 않는 Object Identifier에 대한 SNMP Operation을 미연에 방지할 수 있게 된다. NMS에서 구성할 MIB Data의 구조는 아래 표과 같다.

종 류	구 분		
	Scalar Obj	Columnar Obj	Trap Obj
OID	OID	OID	OID
Obj Name	Object Name	Object Name	Object Name
Type	Type	Type	Type
Syntax	Syntax	Syntax	Null
Status	Status	Status	Status
Access	Access	Access	Access
Description	Description	Description	Description
Columnar	No	Columnar	No
Index	Null	Index	Null
Enterprise	Null	Null	Enterprise
Variable	Null	Null	Variable

표 2. MIB 정보를 위한 MIB Data 구성

5. 결론

본 논문에서 NE Agent MIB을 Meta MIB으로 구현하였고 이러한 Meta MIB을 NMS에서 Import함으로써, NE Agent에 대한 MIB을 재구성 할 수 있도록 하였다. 제안한 Meta MIB을 사용할 경우 오프라인에서 NE Agent와 NMS간에 MIB을 공유 시켜야 한다는 제약을 극복할 수 있다. 이렇게 Meta MIB을 사용하는 것을 기존의 방식과 비교하면, 기존의 방식에 비해 정확하고 자세한 MIB 정보를 NMS에서 알 수는 없게 되지만, NMS는 온라인에서 NE Agent의 MIB을 Import 함으로써 NE Agent와 NMS간 MIB의 공유를 자동화 할 수 있다.

본 논문에서는 SMIv1까지에 대하여 구현하였으며, Range를 가지는 경우의 Syntax들에 대한 표현과, 추상적인 타입들에 대한 재정의, 추후 SMIv2까지도 그 확장에 대하여 연구가 지속적으로 필요하다.

참고문헌(References)

[1] "SNMP, SNMPv2, and CMIP The Practical Guide to Network-Management Standards", William Stallings,

[2] RFC 1155:"Structure and Identification of Management Information for TCP/IP-based Internets", M. Rose, K. McCloghrie, May. 1990

[3] RFC 1902:"Structure of Management Information for Version 2 of the Simple Network Management Protocol " J. Case, K.McCloghrie, M. Rose, S. Waldbusser, January, 1996.

[4] RFC 1213:"Management Information Base for Network Management of TCP/IP-based internets: MIB-II" K. McCloghrie, M. Rose, March, 1991

[5] "SNMP, SNMPv2 and RMON Practical Network Management 2nd Edition", William Stallings, January, 1997

[6] "Data and Computer Communications 5th Edition" William Stallings, 1997