TMN체계의 EMS(Element Management System) 설계 및 구현

정연기[†] · 서승호[††] · 김영탁[†††]

요 약

대규모의 종합 정보 통신망을 운용하고 있는 공중 통신망에서는 ISO/ITU-T의 TMN (Telecommunications Management Network) 체계를 기반으로 한 통신망 관리 기능 구축을 추진하고 있다. TMN은 다양한 기술로 구성된 통신망, 서비스, 장비 등을 관리하기 위해 개발되어 왔다. 본 논문에서는 TMN 체계의 망관리 기능 구현에 핵심적인 EMS (Element Management System)의 기능을 분석하고, 효율적인 서브네트워크 관리 기능 구현을 위한 EMS 구현방안을 제시하며, 구체적인 기능 구현 및 성능 시험을 통하여 이를 검증한다.

본 논문에서 제안하는 EMS는 관리 대상인 NE(Network Element)를 제어하기 위한 EML-Manager모듈 부분과 상위 계층인 NMS(Network Management System)로부터의 요구를 수행하는 NML-Agent 모듈 부분으로 구성되어 있다. 제안된 EMS의 실제 동작을 시험하고 운영자와의 상호작용을 위해 GUI 사용자 인터페이스를 구현했으며, NE-Agent는 본 연구실에서 설계, 구현한 TMN NE-Agent를 사용하였다. 구현된 EMS는 구성관리, 성능관리, 장애관리 등의 기능을 가지며, 시험 결과 정상적으로 동작함을 확인하였다.

앞으로 본 논문에서 제안하는 EMS를 기반으로 NML-Manager기능과 SML-Agent기능을 수행하는 NMS(Network Management System), SML-Manager기능을 수행하는 SMS(Service Management System)의 구현을 통해 SML 계층까지의 전체적인 TMN 체계의 구현이 요구된다.

Design and Implementation of EMS(Element Management System) based on TMN Architecture

Youn-Ky Chung[†], Seung-Ho Seo[††], Young-Tak Kim[†††]

ABSTRACT

We have implemented the telecommunication management functions based on TMN (Telecommunications Management Network) architecture that is standardized by ISO/ITU-T. TMN has been developed to manage heterogeneous networks, services and equipment. In this paper, we first analyze the detailed functions of EMS (Element Management System) that performs core functions in the subnetwork management of TMN architecture. We propose an implementation method of EMS for efficient subnetwork management, and verify its performance and functionality through detailed implementation and various testing.

The proposed EMS consists of EML-Manager module and NML-Agent module. The EML-Manager module controls NEL (Network Element Layer) through the CMIP operations on the managed-objects. The NML-Agent module processes the requests from NMS (Network Management System) in upper layer. We have implemented NMS that performs only the NML-Manager function to test the proposed EMS, and we have used NE-Agent that was designed and implemented by the B-ISDN Lab. of Yeungnam University. The implemented EMS has three management functions: configuration management function,

이 논문은 1998년 한국학술진흥재단의 학술연구비에 의하여 지원되었음.

[†] 경일대학교 컴퓨터공학과
[††] 영남대학교 멀티미디어 통신공학과
[†††] 영남대학교 정보통신공학과

performance management function, and fault management function. We confirmed that the EMS operated normally as we tested the EMS.

As further research works, the implementation of NMS with both NML-Manager function and SML-Agent function is required.

1. 서 론

통신망에 대한 기술이 급속도로 발전하여 점차 초고속화, 광대역화 됨에 따라, 공중 통신망에 있어서 해결해야 할 가장 큰 과제 중의 하나는 ATM/B-ISDN과 같은 새로운 통신망을 효율적으로 운용, 관리하는 기술이다. 통신망을 체계적으로 관리하기 위해 ISO 및 ITU-T에서는 CMIP(Common Management Information Protocol)을 기반으로 한 TMN체계를 제정하고 있다[1-5].

CMIP 기반의 TMN 기능 구조는 SNMP에 비하여 훨씬 복잡하고 다양한 서비스 기능을 포함하고 있으므로, 구현이 용이하지 않아 모든 기능이 구현되기까지는 상당한 시간이 소요될 것으로 예상된다.

TMN체계는 크게 통신망 관리 및 시스템 관리를 위한 Agent기능과, CMIP기반의 Q_3 접속점을 통해 접속되는 운영 시스템(Operation System)으로 구분된다. 운영 시스템은 다시 EML(Element Management Layer), NML(Network Management Layer), SML (Service Management Layer), BML(Business Management Layer)과 같은 4계층으로 이루어지며, 이 계층들은 각각 단위 시스템 및 서브네트워크 관리, 지역 망 및 전국 망 관리, 서비스 관리, 비즈니스 관리의 기능을 담당한다. 그림 1에 TMN 망 관리 계층 구조와 EMS 관련 기능블록을 보였다. 그림에서 음영 처리된 부분은 EMS자체와 EMS에 관련된 기능블록을 표시하고 있다. 별표가 들어간 블록이 본 논문에서 설계하고 구현하는 EMS에 해당되는 부분이다. 이들 4계층 중, 본 논문에서 설계, 구현하려고 하는 EMS와 직접적으로 관련된 계층인 NML, EML, NE계층을 중심으로 좀 더 자세히 알아본다.

NML은 통신망의 종단간(end-to-end) 연결을 취급할 수 있는 네트워크 단위를 관리한다. 따라서 NML계층은 다수의 EML계층 기능모듈(EMS)을 하위에 포함하게 되며, 각 EMS가 관리하는 서브네트워크를 연속적으로 연결하여 하나의 네트워크를 구성하게 된다. NMS의 SML-Agent는 SMS나 상위의

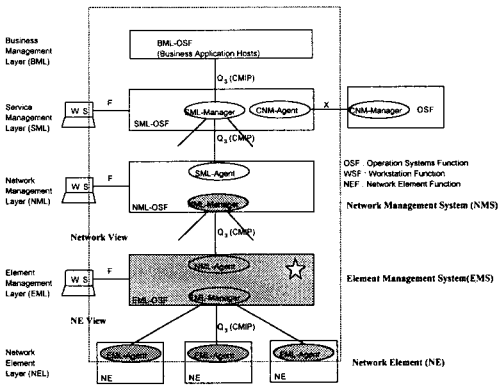

그림 1. TMN 망관리 계층 구조와 EMS관련 기능 블록

NMS로부터 망 단위의 망관리 기능요청을 받아들이고 이를 다시 서브네트워크 단위의 망관리 기능을 자신이 관리 하고 있는 EMS로 내려 보낸다. 망관리 요청을 받은 EMS는 해당 요구를 수행하기 위해 망 관점(Network Level View)의 MO(Managed Object)로 구성된 MIB(Management Information Base) / MIT(Management Information Tree)를 구성해야 한다. 또한 EMS는 자신이 관리하는 서브네트워크 범주 내에서 망관리를 하기 위해 다시 EML-Manager를 통해 NE 속의 EML-Agent로 망 관리 요구를 하게 된다. 이 부분에서 NML과 EML계층간의 망 관점과, EML과 NEL계층간의 망 구성요소 관점(Network Element View)에서 망 관리 기능의 변환이 이루어진다. 또한 EMS나 NMS같이 Manager 역할과 Agent 역할을 모두 수행하는 시스템에서는, 상위 계층에 대해서는 Agent역할을 해야 하고 하위 계층에 대해서는 Manager기능을 제공해야 한다. EMS의 경우 상위의 NMS로는 NML-Agent기능을, 하위의 NE로는 EML-Manager의 기능을 제공해야 한다. 따라서 NML-Agent로 들어온 망 관점의 망관리 명령을 정확히 망 구성요소 관점의 EML-Manager명령으로 변환할 수 있어야 한다.

EML-Manager기능 부분에서는 NML-Agent에서 내려온 명령을 처리하기 위해, CMIP프로토콜을

사용하는 Q_3참조점을 통하여 NE안의 EML-Agent와 망 관리 정보를 주고 받으며, EML-Agent가 관리하는 MO들에 대한 개략적인 정보를 알고 있어야 하며, 추가적인 정보를 탐색해낼 수 있어야 한다.

마지막으로 EML-Agent는 EMS의 EML-Manager로부터 들어온 망관리 요구에 따라 실제 자원을 적절히 조절하고 그에 대한 응답을 해야 하며, 실제 자원에 대한 사건 보고를 EML-Manager로 보내 주어야 한다.

EML-Agent로부터 응답과 사건보고를 받아들인 EML-Manager는 응답과 사건보고를 가공한 후 F, Q_3참조점을 통해 각각 운영자와 상위의 NMS로 보내 주게 된다. 본 논문에서는 위에서 살펴본 TMN체계의 EML-Manager기능과 NML-Agent기능을 수행하는 EMS를 설계하고 구현하였다.

본 논문의 제2장에서는 TMN체계의 계층구조와 본 논문에서 구현할 EMS와 연관된 기능 블록에 대해서 설명한다. 제3장에서는 EMS와 연관된 기능 모듈의 기능을 정의하고 ATM Forum af-nm-0073-000의 시나리오를 따라 VP연결 설정 절차를 보여준다. 제4장에서는 EMS의 설계를, 제5장에서 실제로 구현된 EMS와 실험결과를 보여주며, 제6장에서 결론을 맺는다.

2. TMN Manager의 계층 구조

TMN체계를 더욱 세분화해서 기능 블록으로 나누면 OSF(Operation System Function), NEF(Network Element Function), WSF(WorkStation Function), MF(Mediation Function), QAF(Q Adapter Function) 블록으로 나눌 수 있으며, Manager에 해당하는 OSF 기능 블록은 다시 관리하는 관리 계층(Management layers)에 따라 여러 계층으로 분할될 수 있다.

본 논문에서 설계구현하고자 하는 관리 계층에 해당하는 OSF기능 블록은 EML계층의 EML-OSF이며 이를 EMS라 칭한다.

EMS가 가져야 하는 기능은 크게 NML-Agent기능과 EML-Manager기능으로 분류된다. NML-Agent 기능은 계층 구조상 상부에 위치한 NMS로부터 들어오는 요구를 CMIP을 사용하는 Q_3참조점을 통해 받아들이고, 이것을 자신이 구성하고 있는 MIB/MIT상의 망 관리 객체(Network Managed Object)를 통해 적절한 EML-Manager 명령으로 바꾸어주는 역할을 한다. 이렇게 바뀐 명령을 다시 하위에 위치한 NE속의 EML-Agent로 내려 보내는 기능을EML-Manager기능이라고 한다.

그림 2는 NMS, EMS, NE 간의 관계와 설계 범위를 나타낸다. NE와 EMS간은 망 구성 요소 관점이며, EMS와 NMS간은 망 관점이다. 따라서 EMS는 상위의 NMS로 망 관점의 정보를 제공하기 위해 자신이 관리하는 서브네트워크를 모델링한 망 관리객체 단위의 망 MIB/MIT를 구성할 필요가 있다. 반대로 EMS는 NMS로부터의 망관리 요청을 NE로 다시 요청하기 위하여 NE가 구성하고 있는 NE MIT/MIB에 대한 간략한 정보를 알 필요가 있으며, 실행시에 이를 탐색해낼 수 있어야 한다. 또한 NMS와 EMS간, EMS와 NE간의 망관리 정보 교환을 위해 CMISE/CMIP이 사용되고 있다.

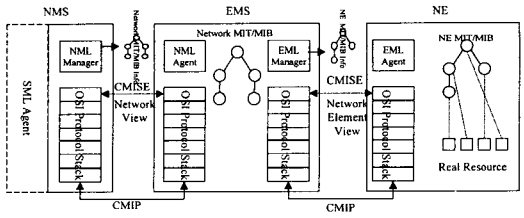

그림 2. NMS, EMS, NE간의 관계와 설계 범위

본 논문에서 구현된 EMS는 상위 계층에 위치한 NMS의 SML-Agent는 향후 과제로 남겨두고, 직접 EMS에 대한 관리를 담당하는 NML-Manager만을 고려하였다. 또한 EMS에 의해 관리되는 EML-Agent는 이미 본 연구실에서 개발이 완료 된 EML-Agent를 이용한다[6-8].

3. EMS의 기능 설계 및 구현

위에서 설명한 바와 같이EMS 안의NML-Agent와 EML-Manager의 기능설계를 개괄적으로 나타내면 그림 3과 같다. 설계된 EMS는 크게 NML-Agent 기능과 EML-Manager기능을 수행하며, 각각이 독립된 하나의 프로세스로 동작하는 다중 프로세스 형태로 설계되었다. NML-Agent 프로세스와 EML-Manager 프로세스 간의 망관리 데이터의 전송은 FIFO를 이용하고 있다. 따라서 NMS의 NML-Manager로부터

망 관리 요구를 수신하는 것과 NE의 EML-Agent로 망 관리 정보를 전송하는 것이 독립적으로 수행된다. 이렇게 다중 프로세스로 구성된 EMS는 특정 NMS로부터의 요구를 처리하는 동안에도 계속해서 다른 NMS로부터 관리요구를 받아들일 수 있다. 한번에 받아 들일 수 있는 관리명령의 개수는 NML-Agent와 EML-Manager간에 쓰인 버퍼의 크기에 따라 제한 된다. 본 논문에서 제안하는 다중 프로세스가 아닌 하나의 프로세스로 EMS를 구성한다면 특정 NMS로부터의 관리 명령을 완전히 처리하고 해당 NMS로 보고를 되돌리기 전에는 차단(Blocking)상태에 놓이게 되어 다른 요구를 받아 들이지 못하게 될 것이다.

본 논문에서 제안하는 EMS는 그 관리 대상을 ATM 망으로 가정하고 있다. 이미 ATM망이 구성되어 있다면 망관리를 위한 또 다른 망을 구성하는 것 대신 기존의 ATM망을 같이 사용한다면 망관리를 위한 비용을 절감시키는 효과를 가져올 것이다. 따라서 본 논문에서 제안하는 EMS는 NMS, NE와의 망 관리 정보 교환을 위하여 ATM을 기반으로 한 CMIP/ATM을 사용하고 있다.

EMS는 TMN 망관리 계층 구조상 연관된 NMS, NE와의 상호 작용 뿐만이 아니라, F참조점을 통해 운영자와의 상호작용의 기능도 제공하여야 한다. 본 논문에서 제안하는 EML-Manager프로세스는 사용자 인터페이스를 통해 운영자의 GUI와 상호 작용한다. 운영자는 GUI를 통해 EMS가 관리하고 있는 서브네트워크에 대한 정보를 탐색하거나 관리 명령을 내릴 수 있다. 또한 서브네트워크로부터의 응답과 사건보고는 FIFO를 통해 NML-Agent에게로 전달됨과 동시에 사용자 인터페이스를 통해 운영자의 GUI에도 표시된다.

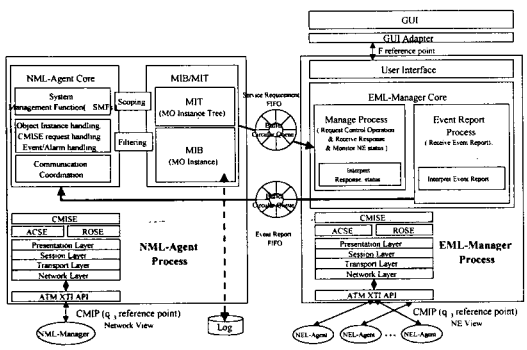

그림 3. 설계된 EMS의 전체 개략도

3.1 NML-Agent 기능

NML Agent는 상위의 NMS로부터 망 관점의 서브네트워크 관리명령을 받아들이고 이를 NE관점의 망관리 명령으로 변환하여 EML-Manager로 전달하는 기능을 한다. 먼저 NML-Agent는 상위의 NMS로부터 관리 명령을 수신하기 위하여 CMIP/ATM을 사용한다. 관리 명령을 수신한 NML-Agent는 자신이 관리하고 있는 서브네트워크를 표현하고 있는 MIB/MIT를 통해 망 관점의 관리명령을 망 구성장치 관점의 관리명령으로 변환하여 EML-Manager를 통하여 NE까지 내려 보낸다. NE에서의 응답과 사건보고는 EML-Manager를 통하여 다시 NML-Agent로 되돌아 오고 NML-Agent는 이를 분석하여 자신이 관리 하고 있는 MIB/MIT를 갱신한 후 다시 NMS로 보고 한다.

NML-Agent는 전체적인 구조 면에서 EML-Agent와 비슷하며, 몇 가지 부분에서 차이가 난다. EML-Agent는 망 구성장치를 모델링한 MO를 유지하여 망 구성장치 관점을 제공하는 반면, NML-Agent는 서브네트워크를 모델링한 MO를 유지하여 망 관점을 제공해야 한다.

망 관점의 MO를 유지하기 위해 EMS는 망 관점의 MO로 구성된 MIB/MIT를 구성하고 관리 하여야 한다. 본 논문에서 제안하는 EMS는 그 관리 대상을 ATM망을 가정하였기 때문에 MIB/MIT구성을 위하여 ATM Forum af-nm-0073.000에 정의된 망 관점에서의 MO가 사용되었다[3-4]. 그림 4는 ATM

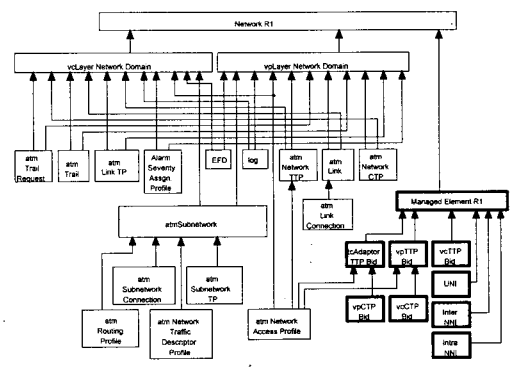

그림 4. 망 관점에서의 CMIP MIB의 포함관계

Forum에서 정의된 망 관점의 CMIP MIB에 대한 포함 관계(Containment)를 나타내고, 그림 5는 망 관점의 CMIP MIB에 대한 상속 관계(Inheritance)를 나타낸다. 그림 5에서 굵은 박스는 ATM Forum의 망 구성장치 관점에서 정의된 객체들을 나타내고 있다. 만약 망 관점만을 제공하고 망 구성장치 관점의 객체를 참조하지 않는 구현이라면 굵은 박스로 표시된 망 구성장치 관점의 객체는 필요 없다.

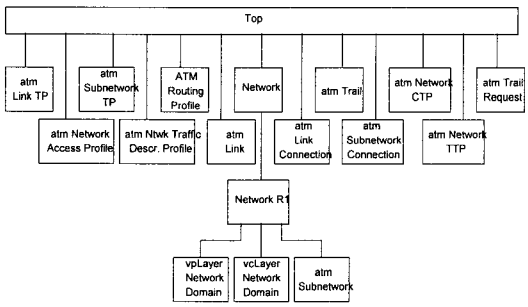

그림 5. 망 관점에서의 CMIP MIB의 상속관계

3.2 EML-Manager기능

EML-Manager의 기능은 NML-Agent의 망관리 요구를 자신이 관리하고 있는 하위의 EML-Agent로 다시 내려 보내는 역할을 하고, EML-Agent로부터의 응답과 사건보고를 종합하여 이를 다시 NML-Agent를 통해 NMS로 올려 보내는 역할을 한다. 또한 EML-Manager는 망 운영자에게 현재 서브네트워크의 정보를 알려주고 관리명령을 받아들이기 위한 사용자 인터페이스를 제공한다.

이를 위해 EML-Manager모듈은 EML-Agent가 관리하고 있는 망 구성장치 관점의 MO들에 대한 개괄적인 정보를 가지고 있어야 한다. 이 정보는 NML-Agent나 운영자로부터 요구된 망관리 요구를 EML-Agent로 전달하는 과정에서, 그리고 EML-Agent가 관리 하고 있는 MIB를 검색할 수 있도록 하는 MIB-Browser등에서 쓰인다. 이를 위해 EML-Agent는 EMS와 연결을 설정하는 즉시 자신이 관리 하고 있는 MO들에 대한 간략 정보를 EML-Manager로 전달 해야 한다. 실행 중에 더 자세한 정보를 요하는 상황이면 MIB-Browser를 통해 더 상세한 정보를 얻어올 수 있다.

3.3 EMS 내부의 NML-Agent프로세스와 EML-Manager 프로세스 간의 망 관리 요구 변환

NMS로부터 NML-Agent로 들어온 망 관점의 망관리 요구는, 망 구성장치 관점의 망관리 요구로 변환되어 EML-Manager로 전달 된다. 이 망 관리 요구 변환은 NML-Agent가 구성하고 있는 망 관점의 MO들에 의해 이루어진다. 이 망 관점의 MO들은 해당 EMS에 의해 관리되고 있는 서브네트워크를 나타내고, 망 관점의 망관리 요구를 받아 하나 이상의 망 구성장치 관점의 명령으로 변환한다. 변환된 망 구성장치 관점의 관리 명령은 EML-Manager를 통하여 실제 망 구성장치까지 전달 된다. 즉, EMS는 NMS로부터 서브네트워크를 바라보는 망 관점의 관리 요구를 받아서, 하나의 망 구성장치 관점의 망관리 요구로 변환하여 NE로 전달 하여야 한다.

예를 들어 NMS로부터 하나의 서브네트워크를 연결하라는 망 관점의 관리명령은 NML-Agent의 서브네트워크 MO로 내려지고, 서브네트워크 MO는 이 관리 명령의 수행을 위하여 하나 혹은 여러 개의 EML-Agent로 망 구성장치 관점의 관리 명령을 전달 한다.

본 논문에서는 NML-Agent와 EML-Manager를 독립된 다중 프로세스 구조로 설계하였으며, NML-Agent로부터의 관리 명령은 환형 큐 버퍼를 사용한 FIFO를 사용하고 있다.

3.4 서브네트워크 연결 관리 시나리오

다음은 ATM Forum af-nm-0073-000의 시나리오에 따른 서브네트워크 관리 시나리오이다[3]. 이 시나리오에서 SubNMS(EMS)는 vcLayerNetworkDomain, vpLayerNetworkDomain그리고 3개의 망 구성장치(NE)를 제어한다. 따라서 망 구성장치 관점과 망 관점의 객체 대부분이 사용된다. 망 관점에서 보면 snTPs가 NE들 안에 있는 CTPs를 직접 가리킬 수 있기 때문에 nwCTPs와 nwTTPs객체 클래스들은 사용되지 않는다. 트레일(trail)들에 대한 관리는 NMS가 수행하므로 SubNMS와 NMS관점에서 특정 트레일 전체에 대한 지원은 필요 없다. 그러나 SubNMS는 SNCs를 설정하고 SNC가 트레일의 끝이라면 TTPs를 연계 시킴으로써 트레일 설정의 일부를 지원한다. SubNMS의 트레일에 대한 지원은

해당 SubNMS가 관리하고 있는 서브네트워크로 제한되며 이때도 해당 서브네트워크 내의 SNC가 트레일의 끝일 때만 트레일에 대한 지원이 필요하다.

다음은 하나의 VP서브네트워크 연결을 설정하는 과정을 나타낸 서브네트워크 연결관리 시나리오이다.

단계1) NMS는 VP서브네트워크 연결의 양쪽 종단점인 snTP1과 snTP3의 식별자와 연결에 관련된 라우팅 정보를 넘겨주며 SubNMS에게 연결설정요구(M-ACTION)를 한다. SubNMS는 NMS의 요구를 서비스하기 위해 필요한 CTPs, profiles과 같은 객체를 자동으로 생성한다.

단계2) SubNMS는 라우팅 정책과 트래픽에 따라 적절한 노드(NEs)들과 링크들을 선택하여 최적의 경로를 결정하고, 선택된 NE들 각각의 atmFabric에 M-ACTION명령을 내림으로써 연결설정을 요구한다.

단계3) NEs는 필요한 CTPs(그리고 필요하다면 TTPs), 대응되는 atmCrossconnections을 사용하거나 생성하여 SubNMS의 요구를 수행한다. 그리고 새로운 개체들이 생성되었다는 것과 연결설정 요구가 수행되었음을 SubNMS로 알린다.

단계4) 수행완료 응답를 받은 SubNMS는 다음을 만든다.

- 필요하다면 NEs들의 CTPs를 가리키는 포인터들에 대응되는 atmSnTP1과 atmSnTP3을 생성.
- NE1.1과 NE1.2간에, NE1.2와 NE1.3간에 atmLinkConnections를 생성. atmLinkConnection 포인터는 NE의 CTPs를 직접 가리킨다.
- atmSnTP1에서 atmSnTP3까지의 atmSubnetworkConnection생성. atmSubnetworkConnection은 서브네트워크에 의해 사용되는 linkConnections와 NEs에 의해 사용되는 atmCrossconnections의 리스트이다.

단계5) SubNMS가 NMS에게 atmSubnetworkConnection이 생성되었음을 보고하는 것으로 VP 서브네트워크 연결설정을 끝낸다.

그림 6은 VP서브네트워크 연결관리 시나리오의 서브네트워크 관리 체계를 나타낸다. 가장 상단에 NML계층을 담당하는 관리시스템인 NMS1이 위치하고, 중간에 EML계층을 담당하는 관리시스템인 SubNMS.1.1이 위치한다. 여기서 SubNMS는 ITU-T TMN체계의 EMS를 뜻한다. SubNMS 아래에는 ATM NE 3개가 상위의 SubNMS로부터 제어되고 있다.

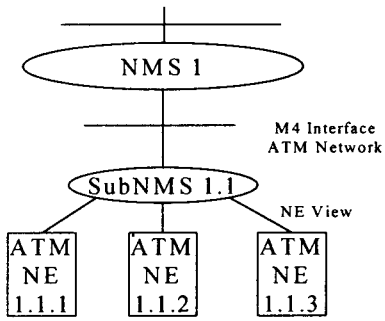

그림 6. 서브네트워크 관리 체계

그림 7은 VP서브네트워크 연결관리 시나리오의 서브네트워크 연결설정 체계와 토폴로지(topology)를 나타낸다. NE들간의 연결은 LinkConnection으로 표시되어 있고 그 종단점인 CTP들이 표시되어 있다. VP서브네트워크 연결이 설정되면 3개 NE의 내부연결은 atmCrossconnections을 이용하고, NE간의 연결은 linkConnections을 사용하여 전체 atmSubnetworkConnection이 구성된다. atmSubnetworkConnection의 양쪽 종단점인 snTP1과 snTP3가 표시되어 있다. 이렇게 ATM NE 3개와 LinkConnection 3개를 합쳐 하나의 Subnetwork를 구성하고 있다.

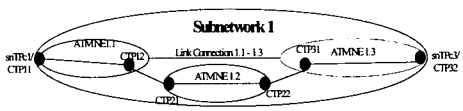

그림 7. 서브네트워크 연결 설정과 토폴로지

3.5 NML-Manager

NMS의 NML-Manager는 EMS에게 Manager역

할을 하는 관리 시스템이다. NMS는 네트워크 전체에 대한 연결설정을 담당하고 전체 연결 설정을 위한 서브네트워크 연결설정을 EMS로 요구한다. 각 서브네트워크 연결설정을 위하여 NML-Manager는 자신이 관리하는 EMS들로 각 서브네트워크의 양쪽 종단점과 관련 라우팅 정보와 함께 SubNMS에게 서브네트워크 연결설정 요구를 한다. NMS는 SMS에게 Agent역할을 하는 SML-Agent기능도 수행한다.

본 논문에서는 EMS의 설계와 구현에 중점을 두었으며, NMS는 SML-Agent기능을 제외한 NML-Manager기능만 갖도록 구현되었다. 따라서 연결을 설정할 두 종단점과 라우팅에 관한 정보는 사용자가 UI(User Interface)를 통해 M-ACTION시 직접 넣어 주어야 한다.

3.6 EML-Agent 와 NE

NE의 EML-Agent는 EMS의 EML-Manager로부터 망관리 요구를 받아들여, 실제 자원에 대해 해당 요구에 맞는 망관리 행위를 수행한다. 앞의 시나리오처럼 EMS가 연결을 설정하라는 요구를 해 오면 EML-Agent는 tcAdaptorTTPBidirectional, vpCTPBidirectional, atmCrossConnection 등의 관련된 MO를 생성하고 해당 요구에 대한 수행 완료 응답을 EMS에게 보낸다.

또한 EML-Agent는 EML-Manager로 사건보고를 해야 한다. 이는 EML-Agent의 M_EVENT_REPORT 기능에 의해 EML-Manager로 전달되며, EFD(Event Forwarding Discriminator) MO에 지정된 조건에 따라 사건보고 여부를 결정한다.

EML-Agent는 본 연구실에서 이미 설계, 구현이 끝난 EML-Agent가 사용되었다[6, 8].

3.7 EMS의 망관리 기능

TMN은 장애관리(Fault Management), 구성관리(Configuration Management), 과금관리(Accounting Management), 성능관리(Performance Management), 보안관리(Security Management)의 5대 기능을 정의하고 있다. 본 논문에서 설계, 구현된 EMS는 그 관리 대상인 NE로 Fore ASX-200 ATM 스위치가 사용되었다. 따라서 5대 망관리 기능의 구현은 ATM 망이 고려되었다.

구성관리에 속하는 연결관리 부분은 CMIS (Common Management Information Service)의 M-ACTION으로 구현되었다. GUI를 통해 운영자로부터 연결에 필요한 정보를 입력 받고, 이를 M-ACTION요구를 통해 EML-Agent로 전송한다. EML-Agent는 요청에 따라 ASX-200 ATM 스위치를 제어하여 VP연결을 설정하고 EMS로 응답을 보낸다. 또한 운영자는 NE의 구성정보를 확인 하기 위해 EMS에 구현된 MO-Browser를 사용할 수 있다.

성능관리는 ASX-200 ATM 스위치의 특정 포트를 통과하고 있는 입/출력 ATM 셀(Cell) 수를 나타내는 것으로 구현되었다. 입/출력 ATM 셀 수를 파악하기 위하여 운영자는 특정 ASX-200 ATM 스위치의 임의의 한 포트에 atmTrafficLoadCurrentData 객체를 생성하고 해당 객체에 셀 계수(Cell Count)명령을 내린다. 셀 계수는 내부적으로 atmTrafficLoadCurrentData 객체에 대해 M-GET을 연속 호출하여 구현되었다. atmTrafficLoadCurrentData로부터 읽어온 입/출력 ATM 셀 정보는 EMS에서 다시 가공되어 그래프로 운영자에게 표시된다.

장애관리는 물리적인 망 장애시 장애 사건보고를 운영자의 GUI에 표시하는 것으로 구현되었다. EMS가 관리하는 ASX-200 ATM 스위치의 임의의 포트에 연결된 케이블을 뽑아 망 장애를 발생시키면 EML-Agent로부터 EMS로 장애 사건보고가 올라가고 EMS는 이를 다시 가공하여 운영자에게 표시한다.

과금관리와 보안관리는 구현에서 제외되었다.

4. EMS의 운영자 인터페이스 기능 설계 및 구현

본 논문에서 설계, 구현된 EMS와 NMS는 현재 관리되고 있는 네트워크에 관한 정보를 가공하여 사용자에게 효과적으로 보여주고, 사용자로부터 망관리 요구를 받아 들이기 위해 GUI(Graphic User Interface)를 사용하였다. 그림 8은 RMI/JNI/FIFO를 사용한 EMS의 Java GUI이다.

본 논문에서는 WebBase GUI를 위해 현재 Web환경에서 가장 주목을 받고 있는 Java를 사용했으며 RMI/JNI/FIFO라는 구조가 사용되었다[9].

① JNI(Java Native Interface) : Native 코드(C/C++)로 작성된 Manager와 Java와의 연결을

위해 사용하였다[9].

② FIFO : 정적 라이브러리(UNIX system에서 .a 파일)를 사용한 Native 코드를 Java JNI를 사용해 연결하기 위해 사용하였다.

③ RMI(Remote Method Invocation) : 현재 Java를 지원하는 브라우저는 보안의 이유 때문에 Native코드의 다운로딩을 허용하지 않는다. 따라서 JNI로 작성된 코드는 Web상에서 local로 다운로드될 수 없다. RMI는 JNI의 시스템 의존적인 Native 코드를 시스템에 독립되게 하여, Web상에서 자유롭게 다운 받을 수 있도록 하기 위해 사용하였다.

그림 8은 Java GUI의 RMI/JNI/FIFO구조를 나타내고 있다. 가장 상단의 Java GUI는 사용자 입력 처리 스레드와 화면 출력처리 스레드로 크게 나뉘어진다. 따라서 사용자가 입력을 하고 있는 동안에도 여전히 응답을 받아들일 수 있다. GUI하부의 RMI/JNI/FIFO구조 역시 각각 독립적으로 동작하므로 셀 계수를 하는 중에도 관리 명령을 내려 보낼 수 있다.

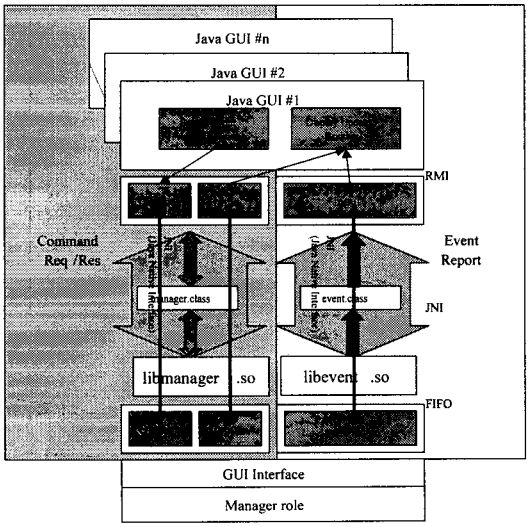

그림 8. RMI/JNI/FIFO를 사용한 EMS의 Java GUI

5. EMS 구현 결과

이상과 같은 설계를 바탕으로 실제 구현된 EMS의 수행 모습을 그림 9에 나타내었다. EMS Java GUI상

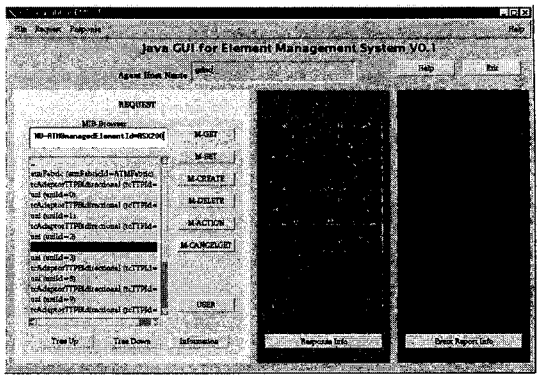

그림 9. 동작중인 EMS의 Java GUI

단에는 Java GUI와 연결된 EML-Agent가 동작하고 있는 호스트 이름이 표시된다. 좌측에는 EML-Agent에서 관리 하고 있는 MIT/MIB상태를 트리 계층구조로 표시하는 MIB-Browser가 위치한다. MIB-Browser 바로 오른쪽에는 CMIS 서비스 요구를 내릴 수 있는 버튼이 위치하고 있다. 중앙에는 EML-Agent에서 EMS로 보내는, 응답을 표시하는 리스트가 있고, 가장 우측에는 사건보고를 표시하는 리스트가 위치하고 있다.

5.1 CMIS 서비스

그림 10은 EMS의 CMIS서비스 중 M-GET을 요청하는 대화상자를 예로 나타내고 있다. 대화상자에는 M-GET서비스를 호출 하기 위한 Object class, Object Instance, Scope, Filter, Attribute 파라메터를 받아들이는 텍스트 필드가 있다. GUI에서 운영자가 입력한 각 파라메터는 RMI/JNI/FIFO를 통해 EML-Manager로 전달되고 이는 m-get()함수의 호출을 통해 실제 M-GET서비스가 제공된다.

EMS는 자신이 관리하고 있는 네트워크의 EML-Agent가 관리하는 MO에 대한 개괄적인 정보를 알고 있으며, 이를 바탕으로 CMIS서비스를 요청할 때 동적으로 변화된 CMIS요청 대화상자를 표시한다. 그림10에서 대화상자 하단에 선택된 MO가 가지는 속성의 종류와 수에 따른 Java Panel이 표시되어 있다. 이는 MO가 바뀌면 다시 해당 MO에 맞게 변경된다.

M-GET에 대한 결과는 메인 화면 중앙에 있는 응답 리스트에 표시되며 운영자의 조작으로 이를 다시 확인할 수 있다. 응답에 대한 자세한 정보보기 대화상자는 그림 11에 표시되어 있다. 운영자는 응답리스

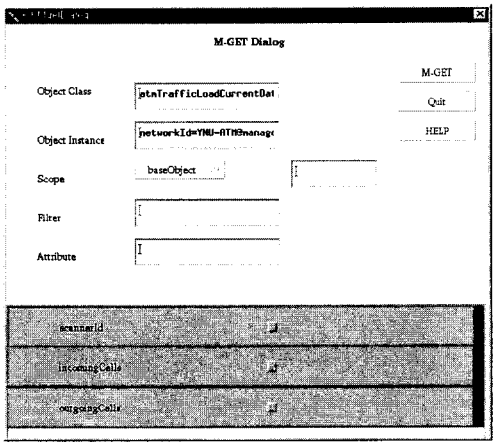

그림 10. CMIS서비스 중 M-GET 대화상자

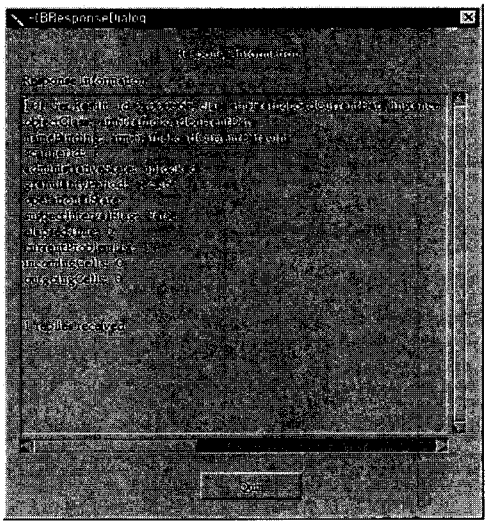

그림 11. EML-Agent의 응답을 표시하는 대화상자

트에서 해당 응답을 선택하고 정보보기 버튼을 통해 더욱 자세한 응답 정보를 살펴볼 수 있다. M-SET, M-CREATE, M-DELETE, M-ACTION, M-CANCEL-GET등의 CMISE서비스도 같은 방법으로 구현되었다.

5.2 연결관리

NE로 연결 설정을 전달 하기 위해 NE의 atmFabric 객체로 CMIS 서비스 중 M-ACTION을 호출 한다. MO-Browser에서 atmFabric객체를 선택하고 M-ACTION 버튼을 선택하면 그림 12와 같은 M-ACTION대화상자가 나타난다. atmFabric객체는 connect, disconnect 두개의 Action이 있고 그림 12 하단에 표시되어있다.

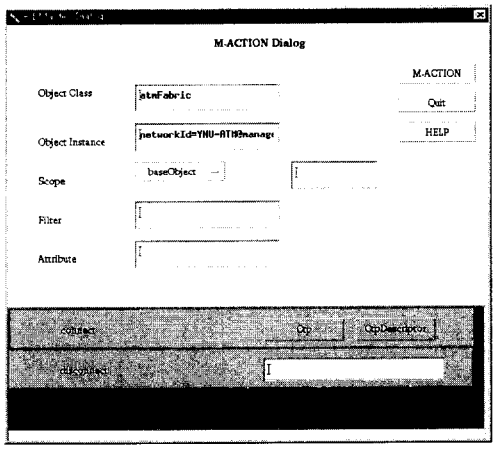

그림 12. 연결 설정을 위한 M-ACTION 대화상자

여기서 connect를 선택하면 그림 13과 같은 대화상자가 나타난다. 운영자는 From Termination과 To Termination에 연결의 양쪽 종단점을 입력하고 Administrative State의 상태를 조정한 후 Set버튼을 선택하면 EML-Manager를 통하여 EML-Agent로 연결 설정 요구가 전달된다. 연결 설정 후 EML-Agent의 보고는 Java GUI의 응답 리스트에 표시되고 운영자의 요구에 따라 더욱 자세한 정보를 제공하는 대화상자가 표시된다.

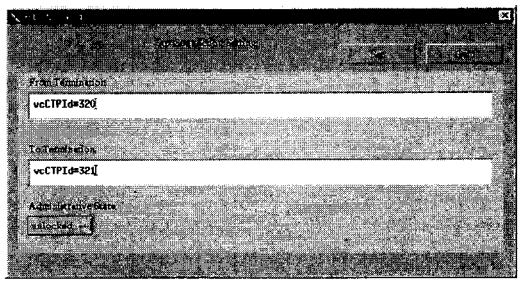

그림 13. atmFabric의 Connect action 호출 대화상자

5.3 성능관리

EMS의 성능관리 기능으로 Fore ASX-200 ATM 스위치의 특정 포트를 통과하고 있는 ATM 셀 수를 모니터링 하고 있는 모습을 그림 14에 나타내었다. 입력 셀 수와 출력 셀 수가 시각적인 그래프 형태로 가공되어 제공되므로 운영자는 망 자원의 성능정보를 보다 직관적으로 파악할 수 있다. 그림 14는 화상통신을 하고 있는 노드가 접속되어 있는 특정 포트에

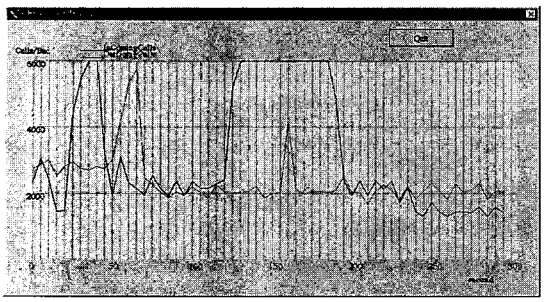

그림 14. ATM 스위치가 특정 포트를 통과하고 있는 셀 수를 모니터링하는 화면

대한 ATM 셀 수이다. 그래프의 초반과 중반부에는 입/출력 화상의 전송이 많은 상태였기 때문에 입/출력 셀 모두 급격한 변화를 보이고 있다. 하지만 후반부에 들어 화상의 움직임을 멈추었을 때 입/출력 셀 수가 모두 급격히 줄어 들고 있음을 나타내고 있다. 그래프에서 세로축은 초당 해당 포트를 지나가고 있는 ATM 셀 수이고 가로 축은 초 단위 시간을 표시하고 있다. 망 운영자는 이러한 성능 측정치로 해당 ATM 스위치에 걸린 부하를 쉽게 파악하고 이를 고려하여 망관리를 수행할 수 있다.

5.4 장애관리

장애관리에 대한 동작을 실험하기 위해, EMS가 관리하고 있는 망에 물리적인 장애가 발생했을 때를 가정하여 Fore ASX-200 ATM 스위치의 특정 포트에 연결된 케이블을 뽑는 것으로 시험하였다. 이에 대한 실험 결과가 그림 9에 나타나 있다. 즉 GUI의 왼쪽에 있는 사건 보고 리스트에 3개의 장애에 관련된 사건보고가 표시되어 있다. 장애발생시 EML-Agent는 stateChange GDMO통지 타입(GDMO Notification Types)으로 Fore ASX-200 ATM 스위치의 asxSw-LinkDown, asxHostLinkDown, asxLinkDown을 보고한다.

사건 보고는 M-EVENT-REPORT를 통하여 서비스되며 이는 다른 CMIS서비스와는 달리 EMS의 요구 없이 EML-Agent가 능동적으로 자신에게 발생한 사건을 보고한다. 따라서 EMS에는 사건보고를 위한 대화상자가 존재하지 않는다. EML-Agent로부터의 사건 보고는 Java GUI의 오른편 리스트에 표시된다.

6. 결 론

효율적인 망관리를 위해 TMN체계는 여러 계층으로 나뉘어져 있으며, 각 계층별로 해당 계층에 가장 적합한 망관리 시스템이 구성될 수 있다. 이들 계층 구조 중에서 본 논문에서는 TMN EMS의 설계 및 구현방안에 관해 연구하였다. 계층 구조 중 EMS는 NMS와 NE사이에 위치하여, NMS로부터 망 관점의 망관리 요구를 받아들여 이를 다시 자신이 관리하는 하나 혹은 다수의 EML-Agent로 보내는 역할을 수행한다.

EMS의 설계 및 구현을 위해 NML-Agent기능 모듈과 EML-Manager기능 모듈을 설계하였으며, NML-Agent와 EML-Manager간에는 FIFO를 통해 망 관리 정보를 전달하는 다중프로세스 구조로 설계하였다. 이를 실제로 동작시키기 위해 EML-Agent는 이미 본 연구실에서 설계, 구현한 EML-Agent가 사용되었다. 또한 EMS의 망 관점의 MO정의는 ATM Forum af-nm-0073에서 정의된 GDMO가 사용되었다.

쉬운 사용자 인터페이스를 위해 Java를 사용한 GUI가 사용되었으며, C/C++ 모듈과 Java모듈간의 연동을 위해 RMI/JNI/FIFO구조가 사용되었다.

TMN의 망 관리 기능 중 구성관리, 성능관리의 기능을 구현함으로써, ATM 스위치의 특정 포트에서 셀 수를 모니터링할 수 있었고, 장애 관리기능을 구현함으로써 물리 망에서 발생되는 장애가 보고됨을 확인하였다.

앞으로 본 논문에서 제안한 EMS를 기반으로 전체 기능을 제공하는 NMS의 설계 및 구현이 요구되며, 이를 이용하여 SML 계층까지, 전체 TMN체계를 설계, 구현하는 것이 남은 과제이다.

참 고 문 헌

[1] ITU-T Rec. M.3010, "Principles for Telecommunications Management Network", Oct. 1992.

[2] ITU-T Rec. X.711, "Common Management Information Protocol Specification for CCITT Application", 1991.

[3] The ATM Forum Technical Committee af-nm-

0073-000, "M4 Network View CMIP MIB Specification Version 1.0", Jan. 1997.
[4] The ATM Forum Technical Committee af-nm-0058.000, "M4 Network-View Interface Requirements, and Logical MIB".
[5] The ATM Forum Technical Committee af-nm-0020.000, "M4 Interface Requirements and Logical MIB", Oct. 1994.
[6] 김성우, 김영탁, "성능 향상을 위한 Multi-thread구조의 GDMO Agent Platform 설계 및 구현", 정보과학회 '98 봄 학술 발표논문집 p255-227, Apr. 1998.
[7] 문애경, 김태우, 조행래, 김영탁, "GDMO Agent에서의 인덱싱 검색 기법을 이용한 MIB/MIT설계 및 구현", KNOM'98 p205-227, Apr. 1998.
[8] ATM/B-ISDN 통신망 관리를 위한 TMN 체계의 GDMO Agent Platform 개발 연차 보고서, ㈜토미스, 1997.
[9] Java Native Interface specification, Sun Microsystems Inc, May. 1997.
[10] 서승호, 김성우, 김영탁, "ATM통신망 관리를 위한 TMN Element Management System(EMS)의 설계 및 구현", 한국통신학회 하계종합학술발표회 논문집 p849-852, July, 1998.

정 연 기

1982년 2월 영남대학교 전자공학과 졸업(공학사)
1984년 2월 영남대학교대학원 전자공학과 졸업(공학석사)
1996년 2월 영남대학교대학원 전자공학과 졸업(공학박사)
1985년 3월~1990년 2월 가톨릭상지대학 전산정보처리과 조교수
1990년 3월~현재 경일대학교 공과대학 컴퓨터공학과 부교수
1998년 1월~1998년 12월 호주 뉴캐슬대학교 전기 및 컴퓨터공학과 교환교수
관심분야 : 멀티미디어 통신, ATM/B-ISDN 기반의 초고속 정보 통신망, TMN/TINA 체계의 통신망 운용관리

서 승 호

1998년 2월 경일대학교 공과대학 컴퓨터공학과 졸업(공학사)
1998년 3월~현재 영남대학교 공과대학 대학원 멀티미디어통신공학과 재학 중
관심분야 : TMN/TINA체계의 통신망 운용관리, ATM/B-ISDN기반의 초고속 정보통신망

김 영 탁

1984년 2월 영남대학교 공과대학 전자공학과 졸업(공학사)
1986년 2월 한국과학 기술원 (KAIST) 전기 및 전자공학과 졸업(공학석사).
1990년 2월 한국과학 기술원 (KAIST) 전기 및 전자공학과 졸업(공학박사)
1990년 3월~1994년 8월 한국통신 통신망연구소 전송망 구조 연구실장
1994년 9월~현재 영남대학교 공과대학 정보통신공학과 부교수
관심분야 : ATM/B-ISDN기반의 초고속 정보통신망, GII(Global Information Infrastructure), 차세대 인터넷(NGI), TMN/TINA체계의 통신망 운용관리