전자교환기용 Message Gathering System 개발에 관한 연구

論 文
53D-2-4

A Study on the Development of Message Gathering System for Digital Electronic Switching System

申 承 湜* · 鄭 讚 壽**
(Seung-Sik Shin · Chan-Soo Chung)

Abstract - The digital electronic switching system has some problems, when this system processes OAM(Operation, Administration And Maintenance) messages about the fault, the traffic, the account, and the performance, etc., at the present. One of these problems is that each the module processing these messages in the digital electronic switching system can not process OAM messages out of another digital electronic switching system. Each the module processing these OAM messages, that is, has been programmed in order to process only the OAM message out of the digital electronic switching system mounting itself, not to process the message out of the digital electronic switching system based on another platform.

Therefore, we studied on methods that could process OAM messages out of all digital electronic switching systems and, first of all, developed the MGS(Message Gathering System) to be able to accept and process the fault message of OAM messages out of all digital electronic switching systems in real time.

Key Words : MGS, MCP, MLP, MFP, Configurator

1. 서 론

현재 디지털 전자교환기의 메시지 게더링 시스템(Message Gathering System : 이하 MGS)은 교환기의 기능(장애, 트래픽, 과금, 회선, 성능 등)에 따라 다양한 형태의 운영 · 관리 메시지를 출력하고 있으며[1][2], 이 운영 · 관리 메시지를 출력하도록 프로그램되어 있는 각각의 기능별 프로세서들은 각 교환기 별로 프로그램되어 유지 · 보수되고 있다[3]. 즉, 각 기능별 프로세서들은 자기 자신이 탑재된 교환기의 특성만을 고려하여 프로그램되어 있기 때문에 다른 교환기로부터 나오는 관리 메시지를 처리할 수 없고, 또 이러한 프로세서들로 구성된 MGS는 전체적으로 다른 교환 시스템의 메시지들을 처리할 수가 없다[4][5]. 이에 본 논문에서는 서로 다른 교환기를 동시에 연결하여 각 교환기에서 나오는 메시지들을 동시에 실시간으로 처리할 수 있는 MGS를 개발 · 구현하고자 한다.

본 논문에서 제안하는 MGS 시스템은 전체적으로 4가지 기능 프로세서로 세분화할 수 있다. 즉, 각 교환기의 다양한 출력 메시지를 모집하는 프로세서(Message Collector Processor), 모집한 메시지를 파일로 저장하는 프로세서(Message Logger Processor), 또 이렇게 모집한 메시지를 비교 · 분석하여 정형화된 메시지를 데이터베이스에 저장하는 프로세서(Message Filter Processor), 그리고 이 Message Filter Processor에서 교환기 출력 메시지를 비교하는데 필요한 데이터의 설정과 출력으로 내보낼 정형화된 형태의 메시지를 정의해주는 프로세서(Configurator)로 구분할 수 있다. 이와 같이 본 연구에서 개발할 MGS 시스템의 프로그램 프로세서 전체 공정은 Message Collector Processor, Message Logger Processor, Configurator, 그리고 Message Filter Processor로 구성된다.

* 正 會 員 : 崇實大學 電氣工學科 博士修了
** 正 會 員 : 崇實大學 電氣工學科 敎授 · 工博
接受日字 : 2003年 10月 1日
最終完了 : 2003年 12月 2日

2. Message Gathering System

2.1 Message Gathering System 개발 환경

본 논문에서 MGS시스템을 탑재할 시스템으로 리눅스 서버를 사용하고, 메시지의 관리를 위해 Oracle DB(Oracle 8i 8.1.5 Standard Edition for Linux)를 사용하였으며, MGS에 경보 메시지를 보낼 교환기로서는 범용 IBM PC로 모형화하였다. 또한, MGS 시스템과 여러 기종의 교환기 시스템을 연동하기 위해 Terminal Server를 사용하였다. 이러한 MGS 시스템 개발 환경을 그림 2.1에 나타내었다.

그림 2.1에서 경보 메시지를 보낼 교환기1과 교환기2는 범용 IBM PC로 모형화할 수 있고, 또 각각의 교환기는 RS-232 port를 통해 9600bps의 전송속도로 경보 메시지를 Terminal Server에 보낸다. Terminal Server에서는 이 경보 메시지를 패킷 처리하여 MGS가 탑재되어 있는 Linux Server로 보내고, Linux Server는 이 교환기의 경보 메시지를 개발 PC에 탑재되어 있는 Configurator의 설정 상황과 비교하여 정형화된 메시지를 데이터베이스에 적재한다.

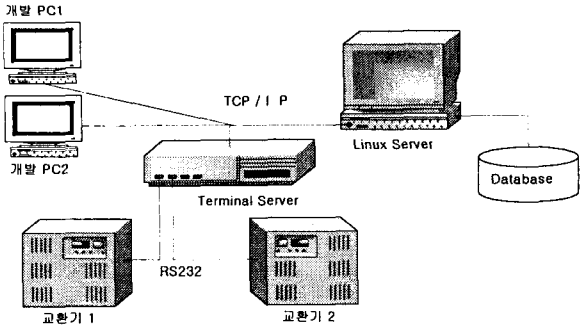


그림 2.1 MGS 시스템 개발 환경
Fig. 2.1 Developing environment for the MGS

2.2 Message Gathering System의 프로세서

본 논문에서 개발하고자하는 MGS의 Processor는 서론에서 언급한 바와 같이 크게 MCP(Message Collector Processor), MLP(Message Logger Processor), Configurator, MFP(Message Filter Processor) 등으로 이루어져 있으며, 본 절에서는 4가지 주요 processor에 대하여 자세히 설명하고자 한다. 먼저, MCP의 주요기능 및 세부기능에 대해서 알아보겠다.

2.2.1 Message Collector Processor

2.2.1.1 주요 기능

각 장비에서 나오는 메시지는 X.25, TCP/IP, RS232/422 등과 같이 다양한 port에서 출력되는 것을 전제로 한다. 이때 MCP는 해당 장비로부터 출력되는 메시지를 서버(예 : UNIX System)의 디바이스 장치(Terminal Server)에 연결시켜 1Byte 단위로 읽어내고, 읽어낸 메시지를 MLP로 전달한다. MCP의 세부 기능은 그림 2.2에 나타낸 Message Collector Processor의 세부 모듈 구성도를 보면서 설명하겠다.

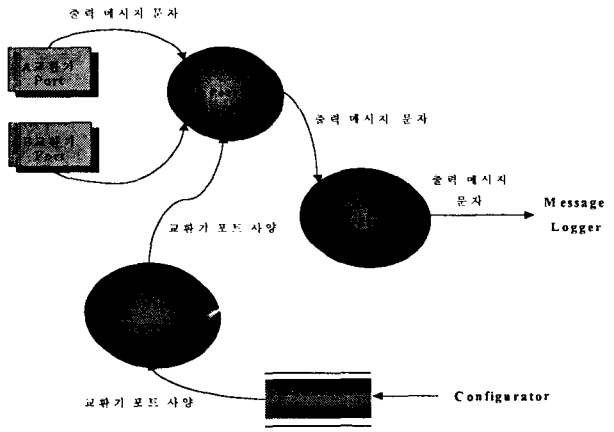


그림 2.2 Message Collector Processor의 세부 모듈 구성도
Fig. 2.2 Detailed diagram of Message Collector Processor

2.2.1.2 각 모듈의 세부 기능

1. Port Detector Module

Port Detector Module은 MCP에서 가장 중요한 모듈로서, 각 장비에서 출력되는 메시지 포트들(X.25, TCP/IP, RS232/422,…)에 상응하여 각 프로토콜에 맞는 응용 프로그램을 수행시키고, 이를 Server 디바이스 장치에 논리적으로 연결시켜 해당 장비와 통신할 수 있도록 하는 모듈이다. 여기서의 응용 프로그램이란 해당 Port와 서버의 디바이스 장치와 통신할 수 있도록 개발되어 있는 프로그램을 말한다. 각 장비에서 출력되는 메시지를 전송하는 라인 등에 따라 응용 프로그램은 달라질 수 있다.

2. Message Log Sender Module

각 장비에서 출력된 해당 메시지를 메시지 로거 프로세서로 전달해주는 모듈이다.

3. Collector Memory Monitor Module

운영자가 교환기 Port 사양으로 입력한 조건을 공유 메모리에서 감지하는 모듈이다.

2.2.2 Message Logger Processor

2.2.2.1 주요 기능

MLP는 MCP로부터 전달받은 1 Byte 데이터를 운영자가 원하는 log 파일로 만들어 MFP로 전달하는 프로세서이다. MLP의 세부 모듈 구성은 그림 2.3에 나타내었다.

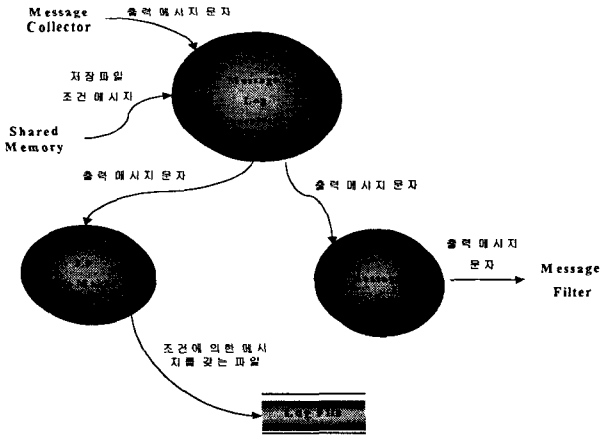


그림 2.3 Message Logger Processor의 세부 모듈 구성도
Fig. 2.3 Detailed diagram of Message Logger Porcessor

2.2.2.2 각 모듈의 세부 기능

1. Message Log Receiver Module

MCP로부터 전달받은 데이터를 1 Byte씩 읽어낸다.

2. Message Filter Sender Module

MCP로부터 전달받은 1 Byte씩의 데이터를 MFP로 전달한다.

3. File Maker Module

File Maker Module은 모든 데이터 Log File을 생성한다. 운영자가 Configurator Processor에서 지정한 시간별, 일별, 월별 등으로 운영자가 지정한 디렉토리에 Log File을 생성한다.

2.2.3 Configurator Processor

2.2.3.1 주요 기능

Configurator Processor는 전자 교환기 및 사설 교환기의 Port 사양, 해당 장비의 여러 가지 메시지 설정, 여러 프로세서들의 기동 및 종료, Log File 디렉토리 및 생성 주기, 미리 관리되고 있는 정형화된 메시지 Class 등의 모든 정보들을 직접 GUI 화면상에서 용이하게 설정할 수 있도록 구현된 프로세서이다. Configurator Processor의 세부 모듈 구성도는 그림 2.4와 같다.

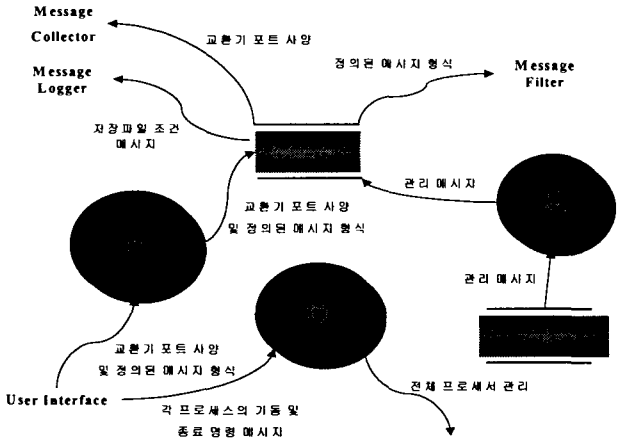


그림 2.4 Configurator Processor의 세부 모듈 구성도
Fig. 2.4 Detailed diagram of Configurator Processor

2.2.3.2 각 모듈의 세부 기능

1. User Machine Interface Module

Configurator Processor상에서 가장 핵심적인 부분인 User Machine Interface Module의 세부 기능을 설명하면 다음과 같다.

① User Machine Interface Module은 X Programming GUI 화면으로 구성되어 있기 때문에, 각각의 다른 장비의 특성들을 GUI 화면에서 직접 설정할 수 있다.

② MLP에서 생성된 Log File을 운영자가 User Machine Interface Module에서 수동 삭제할 수 있을 뿐만 아니라, 생성된 Log File을 주기적으로 자동 삭제시킬 수도 있으며, Log File 생성 주기 역시 시간별, 일별, 월별로 시스템 사양을 고려하여 운영자가 직접 설정할 수도 있다.

③ User Machine Interface Module에 다양하게 정형화된 메시지를 등록할 수 있을 뿐만 아니라 이미 관리되고 있는 데이터베이스 또는 파일 메시지 Class를 삭제할 수도 있다.

④ 운영자는 User Machine Interface Module에 메시지 Parsing rule을 입력시킬 수 있기 때문에, 새로운 메시지 포맷이 나왔을 경우 운영자는 쉽게 원하는 메시지를 수집할 수 있도록 메시지 형태를 등록할 수 있다.

⑤ 새로 설정된 값들은 상호 연동할 수 있도록 공유 메모리를 사용하여 설계되어 있기 때문에, 각각의 기능을 수행하고 있는 MCP, MLP, MFP 등에 쉽게 적용할 수 있다.

⑥ 기존의 Parsing rule을 삭제 및 수정할 수 있다.

2. Define Message Reader Module

MFP가 메시지를 분석할 수 있도록 하기 위하여, Define Message Reader Module은 User Machine Interface Module에서 운영자가 미리 정형화시킨 메시지 Class를 읽어내어 공유 메모리에 저장한다.

3. Process Controller Module

Process Controller Module은 전체 MGS에서 기동되는 모든 프로세서를 장비에 맞게 기동 및 중지시킬 수 있는 역할을 수행한다.

2.2.4 Message Filter Processor

2.2.4.1 주요 기능

MFP는 해당 장비에서 출력되는 다양한 메시지 종류들 중 Configurator Processor에서 이미 등록한 관리 Message Class와 운영자가 등록한 새로운 메시지를 모두 처리한다. MFP의 세부 모듈은 그림 2.5와 같다.

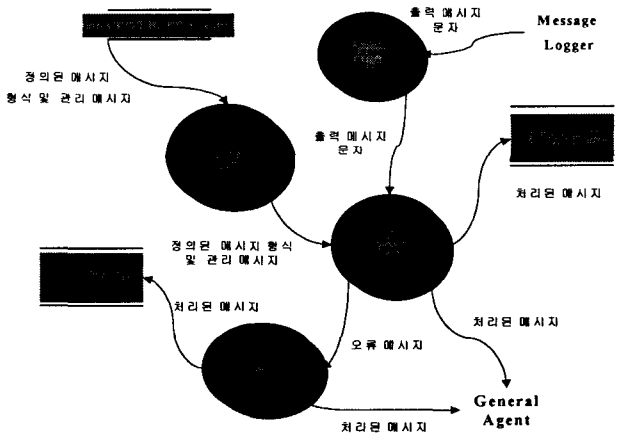


그림 2.5 Message Filter Processor의 세부 모듈 구성도
Fig. 2.5 Detailed diagram of MFP

2.2.4.2 각 모듈의 세부 기능

1. Parser Memory Monitor Module

Parser Memory Monitor Module은 운영자가 실시간으로 Parsing rule과 메시지 형식을 등록하면 이를 즉각 수용할 수 있도록 공유 메모리를 감지하는 모듈이다.

2. Message Parser Receiver Module

Message Parser Receiver Module은 MLP로부터 전달받은

1 Byte 데이터를 읽어내어 Message Parser Module로 전달하는 모듈이다.

. Message Parser Module

Message Parser Module은 MLP로부터 읽어들인 1 Byte 데이터를 읽어들여 문자열 형태로 만든다. 문자열 형태의 데이터 스트림을 생성해내면서 동시에 이 데이터가 운영자에 의해 등록된 메시지인지를 판별한다. 운영자가 등록한 메시지이면 메시지 형식에 맞쳐 메시지를 Parsing하고, 이를 분석된 메시지 Class에 저장한다. 운영자가 등록하지 않은 메시지이면 Message Debugger Module로 보낸다. 운영자가 Configurator Processor에서 등록한 모든 메시지를 운영자가 정한 Parsing rule에 의해서 메시지를 분석한다.

. Message Debugger Module

Message Debugger Module은 Message Parser Module에 의해 오류 메시지로 판별된 메시지를 정형화된 오류 판별법에 의해 알맞게 가공하는 모듈이다.

3. MGS(Message Gathering System) 구현

3.1 메시지의 구조와 분석

교환기에서 출력되는 대부분의 메시지는 그림 3.1에 보인 바와 같이 Head, Body, Tail 형태의 구조를 갖는다[4].

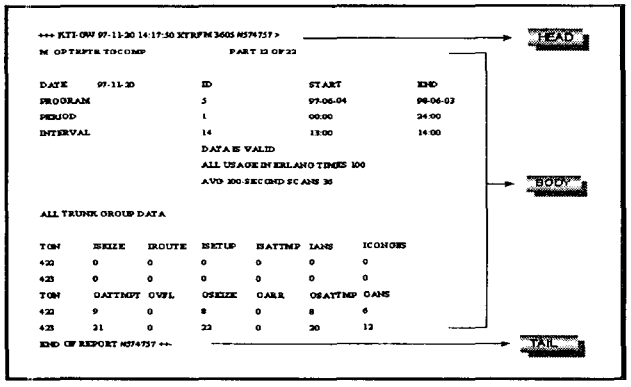


그림 3.1 메시지 구조
Fig. 3.1 Message structure

여기서 메시지의 Head와 Tail은 다른 메시지와 구분되는 특징적인 문자열로 각 메시지의 시작과 끝을 알리는 문자열이며, Body는 Head와 Tail 사이에 측정 결과 등의 메시지 본래 내용을 포함시키는 부분이다[5]. 본 논문에서 사용한 교환기의 출력 메시지는 현재 한국통신 국제망에서 사용하고 있는 AXE-10교환기와 5ESS교환기의 경보메시지이고, 표1과 표2에 AXE-10교환기와 5ESS교환기의 주요 경보메시지를 각각 나타내었다. 또한, 이러한 경보 메시지의 등급은 표3과 같이 세 개의 등급으로 나누어 구분하고 있다.

표 1 AXE-10 교환기의 주요 경보 메시지
Table 1 Adjustment of main Alarm Message for AXE-10

Ref-ID	메시지 내용
0201	ALI FAULT
0202	COMMAND LOG BLOCKED
0203	RPPAIR FAULT
0204	CPG FAN FAULT
0205	CPT FAULT

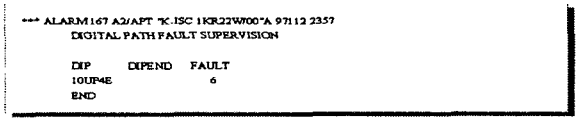
표 2 5ESS 교환기의 주요 경보 메시지
Table 2 Adjustment of main Alarm Message for 5ESS

Ref-ID	메시지 내용
0001	3193*C REPT SM HLSC HORTAGE
0002	3204*C OP OVRLD
0003	0167*C REPT PHASE IN ROGRESS
0004	0237** REPT AUTONOMOUS RESPONSE
0005	0266** REPT DKDIP MESSAGE

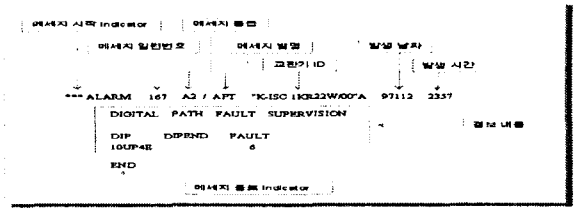
표 3 경보 등급
Table 3 Adjustment of main Alarm Message for AXE-10

Alarm 등급	AXE-10	5ESS	내 용
Critical	A1	*C	Critical Trouble. Immediate Action Required
Major	A2	**	Major Trouble or Power Failure
Minor	A3	*	Minor Trouble Action Required

표 3은 표 1과 표 2에 나타낸 두 교환기의 경보 메시지를 각각의 설정 상황에 따라 세 등급으로 나누어 구분하고 있음을 보여주고 있는 것이다. 즉, 표1과 표2의 경보메시지는 표3의 Alarm 등급에 따라 Critical과 Major, Minor로 나누어 구분하고 있는데, 좀더 자세하게 설명하기 위하여 표 1과 표2의 경보 메시지의 내용이 포함되어 있는 출력메시지를 그림 3.2와 그림 3.3에 분석해 놓았다.

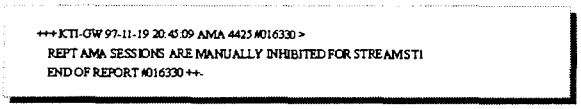


(a) 출력 메시지
(a) Output message

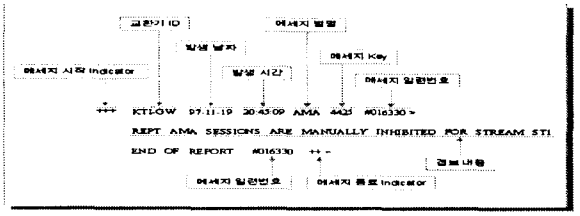


(b) 출력 메시지 분석
(b) Analysis of output message

그림 3.2 AXE-10 교환기의 경보에 대한 출력 메시지와 분석
Fig. 3.2 The alarm output message and analysis of AXE-10



(a) 출력 메시지
(a) Output message



(b) 출력 메시지 분석
(b) Analysis of output message

그림 3.3 5ESS 교환기의 경보에 대한 출력 메시지와 분석
Fig. 3.3 The alarm output message and analysis of 5ESS

즉, 그림 3.2와 그림 3.3은 AXE-10교환기와 5ESS 교환기의 경보메시지(표1, 표2) 중 일부를 나타낸 그림으로서, 그림과 같은 메시지를 분석하면 메시지 시작을 알리는 메시지 시작 지시자(message start indicator)는 ***과 +++로 AXE-10과 5ESS교환기가 각각 사용하고 있는 것을 알 수 있으며, 메시지 종료 지시자로서는 END와 ++-로 사용하고 있는 것을 알 수 있다[6][7]. 메시지를 처리할 때 이렇게 시작을 알리는 지시자와 종료를 알리는 지시자가 틀리기 때문에 각각의 교환기는 같은 기종의 메시지만을 처리하고 다른 메시지들은 무시하게 되는데, 본 논문에서는 이러한 문제점을 해결하기 위하여 메시지 시작 지시자와 종료 지시자를 Main함수의 인자(argument)로 받아 들여 처리한다. 즉, MCP의 Main 함수에서 시작 지시자와 종료 지시자를 입력으로 받아 들여 처리함[8]으로써, 어느 교환기의 메시지라도 수용하여 처리할 수 있는 시스템을 구축할 것이다. 다시 그림 3.2와 그림 3.3에 대해 설명을 하면, 경보 등급(그림에서는 메시지 등급으로 표현)은 그림 3.2에서처럼 AXE-10 교환기는 출력 메시지에서 A2(AXE-10 교환기는 표3에 나타낸 것처럼 경보 등급을 A1, A2, A3로 나누어 구분하고 있음)로 구분하고 있는데 반해, 그림 3.3의 5ESS 교환기 메시지는 AXE-10과 같은 경보등급이 아닌 *C, **, *으로 경보등급을 구분하고 있다[6[7]][10]. 즉, 이렇게 경보 등급 또한 각각의 이기종 교환기가 서로 다른 설정으로 처리되고 있기 때문에 본 논문에서는 이러한 내용도 통합 처리할 수 있는 Message Gathering System을 제안하는 것이다.

3.2 Message Gathering System 모듈의 Simulation

이제, 실제로 모의 실험한 내용에 대해 소개할 것이다. 먼저, MGS의 입력으로 들어갈 교환기의 출력메시지를 random하게 발생시킬 것이고, 발생된 메시지는 MGS 시스템의 MCP에서 읽고, MCP는 MLP로 전송하기 위하여 메시지 큐에 저장할 것이다. 또, MLP는 로그파일을 생성하고 MCP로부터 받은 메시지를 MFP로 전송할 것이다. 물론, 메시지 큐를 이용하는데 마찬가지로 MFP에서도 메시지 큐에 있는 내용을 읽어서 Configurator에서 설정한 "정의된 메시지 형식"과 비교하여, 일치하는 메시지를 DB에 저장한다. 여기서, 정의된 메시지 형식이란 그림 3.2와 그림 3.3에서 설명한 바와 같이 교환기의 시작과 종료 메시지 지시자의 형식을 의미한다. 즉, Configurator에서 설정한 "정의된 메시지" 형식은 사용자나 운영자가 원하는 데이터의 형식으로써, MFP는 이 정의된 메시지 형식과 메시지 큐에서 읽은 메시지 중에서 메시지의 시작과 종료를 나타내는 지시자와 비교하여 일치하면 이 메시지를 DB에 저장하면 된다. 물론, 메시지 큐에 있는 메시지 형식은 MCP의 Main함수에서 인자로 입력받은 시작 메시지와 종료 메시지의 지시자를 의미한다.

이와 같이 수행함으로써, 본 MGS는 이기종 간의 메시지를 통합하여 수용할 수 있으며, 원하는 메시지를 저장하고 출력할 수 있었다.

3.2.1 입력 데이터

본 절의 서두에서 설명한 바와 같이 그림 3-4는 교환기에서 출력되는 메시지를 모형화한 것이다.

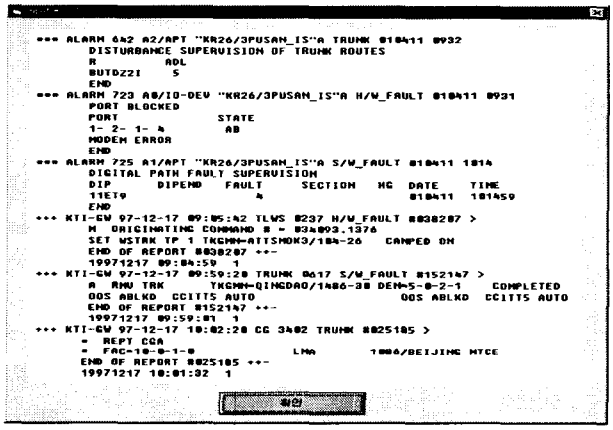


그림 3.4 임의의 입력 메시지
Fig. 3.4 Random input message

즉, 그림 3-4에서 임의의 입력 메시지는 한국통신의 국제관문국에서 사용되는 AXE-10과 5ESS 교환기의 원시 데이터 중 경보에 대한 메시지를 사용하였는데, 이외의 원시 데이터(과금이나 트래픽, 기능 등에 대한 데이터)는 몸체에 해당되는 데이터, 즉 body 부분이 크기 때문에 분석의 용이함을 고려하여 비교적 간단한 형태의 경보 메시지를 사용하였다. 그림 3.4와 같은 메시지는 2장에서 설명한 바와 같이 MCP, MLP, MFP 등의 프로세서에 의해 적절하게 처리될 것이다.

3.2.2 *Message Collector Processor*

서론에서도 언급한 바와 같이 MCP는 교환기로부터 입력받은 데이터를 취합하여 MLP에 전달하는 역할을 한다. 그림 3.5는 MCP를 구성하고 있는 프로그램의 루틴들을 도시하였다.

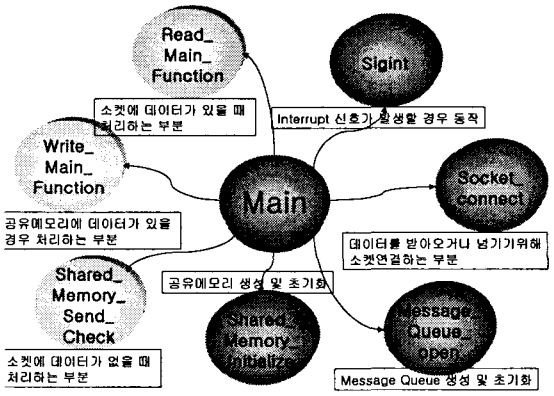


그림 3.5 Message Collector Processor의 프로그램 구성도
Fig. 3.5 Programming diagram of MCP

그림 3.5에서 Main 루틴은 데이터를 받고 전송하기 위해 소켓에 연결하는 함수와 메시지큐 및 공유메모리의 초기화 등을 하는 함수, 인터럽트가 발생했을 경우에 처리하기 위한 함수 등으로 구성되어진다. 이렇게 구성된 루틴들의 순서를 그림 3.6에 나타내었다.

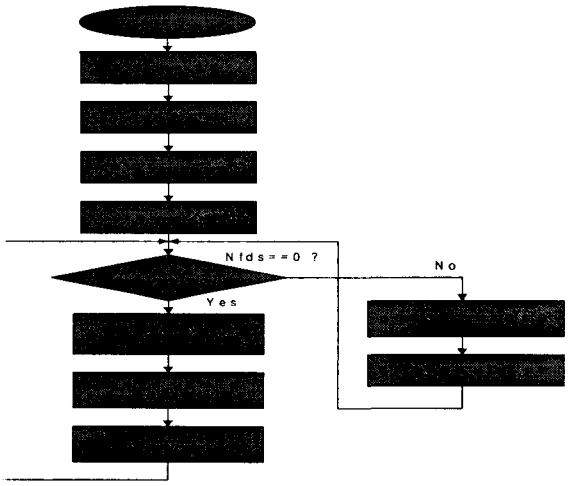


그림 3.6 Message Collector Processor의 순서도
Fig. 3.6 Flow chart of MCP

그림 3.6의 처음부분은 인수를 초기화하는 부분이고 다음은 메시지 큐를 생성하는 부분인데 교환기와 소켓 연결한 후 교환기로부터 입력으로 받은 데이터를 임시로 저장할 곳이다[8]. 다음은 공유메모리를 생성하는 루틴으로서, 공유메모리에는 Configurator에서 설정한 교환기의 Port 사양이 저장된다. 이제 소켓에서 데이터를 읽어오면 된다[9]. 소켓에서 데이터를 읽어올 때는 소켓에 데이터가 있는지 없는지 확인하여 처리해야 할 것이다. 즉, 그림 3.6에서 메시지 큐에 데이터가 있으면(Nfds≠0) 데이터를 읽으면 될 것이고, 없을 경우(Nfds==0)에는 공유메모리에 데이터가 있는지 체크해서 데이터가 있으면 데이터를 읽어오면된다. 참고적으로, Unix 계열의 표준함수인 select()함수를 이용하여 소켓에 데이터가 있는지 확인하였는데, 그림에서 Nfds는 select()함수의 리턴값을 받는 인수다[8]. 만일 Nfds≠0이면 소켓에 데이터가 있는 경우로서 데이터를 읽고, 파일을 만들고, 이 파일에 데이터를 써서 MLP로 보내는 과정을 수행한 후 다시 소켓에 데이터가 있는지 확인하는 구문으로 리턴하며, 만일 Nfds==0이면 소켓에 데이터가 없는 경우로서, 공유메모리에 데이터가 있는지를 체크하는 과정을 거친다. 여기서, 공유메모리에 데이터가 있을 경우에는 공유메모리의 데이터를 MFP로 보내는 작업을 하고, 사용한 공유메모리는 NULL로 셋트시킨 후 소켓에 데이터가 있는지 체크하는 구문으로 리턴한다. 즉 소켓에 데이터가 있으면 그 데이터는 MLP로 보내고, 소켓에 데이터가 없으면 공유 메모리를 체크하여(Configurator에서 설정한 메시지 형식이 있는지 체크하는 과정) 데이터가 있으면 이 데이터를 MFP로 보낸다.

3.2.3 Message Logger Processor

MLP는 MCP로부터 받은 데이터를 파일화하고, 이 데이터를 MFP로 전달하는 역할을 한다.

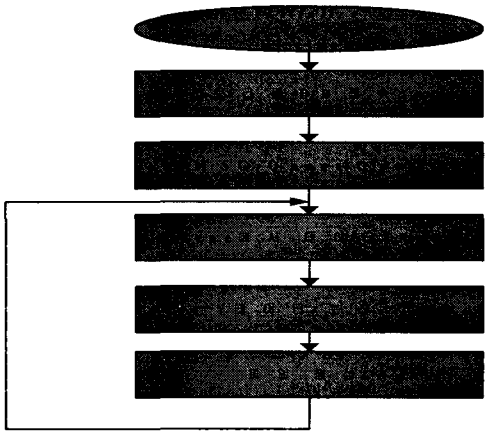


그림 3.7 Message Logger Processor의 순서도
Fig. 3.7 Flow chart of MLP

그림 3.7은 MLP의 순서도이다.
순서도에 대한 설명을 단계적으로 하면 다음과 같다.

1. Initialize & Declare 부
 · MCP와 마찬가지로 변수와 인수 등을 선언한다.
2. 오라클 DB 접속
 · Database와 연결을 한다.
3. MSG Queue 생성
 · MCP에서 설명한 것처럼 메시지 큐에 있는 데이터를 갖고 오기 위하여 메시지 큐를 생성하고 초기화한 후, 데이터를 갖고 올 기술자(identifier)를 선언한다.
4. Queue에서 데이터 Read
 · 메시지 큐에 있는 데이터를 읽어온다.
5. 데이터 분리
 · 메시지 큐에서 읽은 데이터를 또 다른 큐에 저장하고, 원래의 메시지 큐를 NULL로 셋트시킨다.
6. 파일 생성
 · 파일을 생성하여, 이 파일에 저장된 데이터를 저장한 후 단계4로 리턴한다.

지금의 MLP에서는 DB에 연결하여 MCP에서 보낸 데이터를 이용하여 로그 파일을 생성하였다. 또한, MFP에서는

MLP에서 저장해 놓은 메시지 큐의 내용을 읽어서 Parsing rule에 따라 데이터를 분리할 것이다. 참고적으로, 여기서 파일화된 log파일에는 교환기의 메시지가 내용으로 들어 있다.

3.2.4 Message Filter Processor

MFP는 MLP로부터 전달 받은 데이터 중에서 Configurator에서 정의한 메시지 형식과 일치하는 데이터를 DB에 저장하고, 출력하는 역할을 한다. MFP의 순서도는 그림 3.8에 나타내었다.

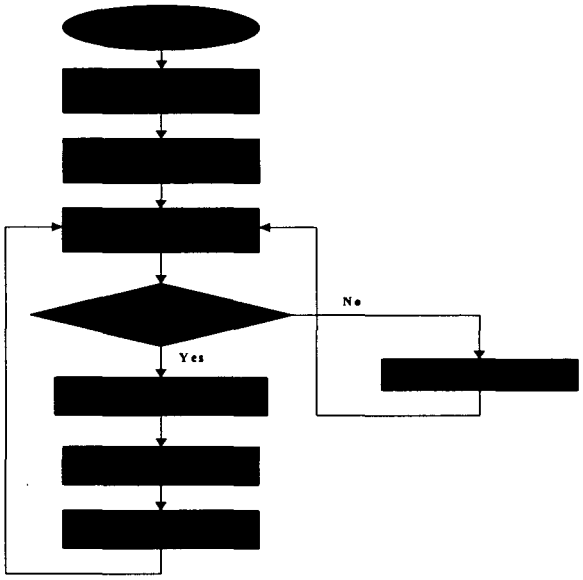


그림 3.8 Message Filter Processor의 순서도
Fig. 3.8 Flow chart of MFP

그림 3.8에 나타낸 MFP에 대해서 설명하면 다음과 같다.

1. Initialize & Declare 부
 · 변수와 인수 등을 선언한다.
2. 오라클 DB 접속
 · MLP에서와 마찬가지로 DB에 연결한다.
3. MSG Queue 생성
 · Message Queue를 생성하고, 초기화한다.
4. Queue에서 데이터 Read
 · 메시지 큐에 있는 데이터를 저장한다.
5. 데이터 비교
 · 메시지 큐에 있는 데이터 중에 정형화된 메시지만을 분리하여 또 다른 메시지 큐에 저장한다.
 · 메시지 큐에 저장되어 있는 데이터와 Configurator에서 설정한 정형화 된 메시지를 비교한다.
6. 데이터 저장
 · 메시지 큐의 데이터와 정형화된 메시지와 같으면 수행하는 루틴으로써, 데이터를 읽어서 다른 메시지 큐에 저장한다.
7. 데이터 Parsing
 · 데이터를 Parsing rule에 의해 분리한다.
8. Parsing한 데이터를 DB에 저장
 · 분리한 데이터를 파일로 생성하고 DB에 적재한다.
9. 메시지 큐의 데이터 삭제
 · 단계6에서 정형화된 메시지와 메시지 큐에 있는 데이터가 다를 경우 수행하는 루틴이다. 이 경우는 운영자가 원하는 경보 메시지가 아니기 때문에 이 데이터를 삭제하기 위한 작업을 수행한다. 메시지 큐에 데이터를 파일로 만들고 데이터를 삭제한다.

3.2.5 Configurator Processor

Configurator Processor는 MCP와 MFP로 교환기의 Port 사양 및 정의된 메시지 형식 등을 전달하는 프로세서이다.

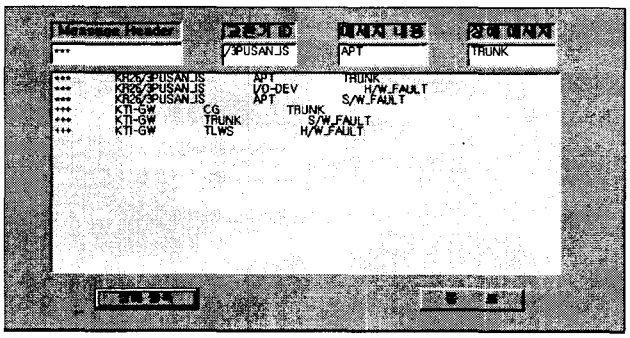


그림 3.9 Configurator Processor의 경보 메시지 등록
Fig. 3.9 Alarm message registration at Configurator

그림 3.9는 Configurator Processor의 경보메시지 등록화면이다. 그림 3.2와 그림 3.3에서 언급한 바와 같이 각 교환기의 데이터 형식 중 Message Header와 교환기 ID 그리고 메시지 내용, 장애 메시지를 등록한다. 여기서 등록된 경보 데이터를 갖는 메시지를 MFP에서는 구분하여 DB에 적재할 것이다. 즉, Configurator Processor에서 등록한 데이터를 제안한 MGS는 수많은 원시메시지 중에서 찾아내어 DB에 저장하고 출력한다.

이상으로 본 절에서는 제안한 MGS의 프로세서 구현에 대해 설명을 하였다. 참고적으로 본 연구에서 사용된 데이터베이스는 Oracle DB(Oracle 8i 8.1.5 Standard Edition for Linux)이고, Language는 Proc*를 사용하였다.

3.3 제안한 MGS 시스템의 주요기능

여러 교환기에서 입력되는 메시지를 처리하기 위해 Terminal Server를 사용하여 메시지를 수집하였으며, 교환기 장애 메시지는 한국통신 국제망에서 사용하고 있는 원시 데이터를 사용하였다. 이에, 우리는 2장에서 기술한 프로세서를 이용하여 한국통신 국제망에서 사용하는 AXE-10과 5ESS교환기의 장애 메시지를 시험 데이터로 삼아 다음과 기능을 갖는 MGS 시스템을 구현할 수 있었다.

1. 여러 전자 교환기의 출력 메시지를 동시에 시간으로 처리할 수 있는 기능
2. 교환시스템의 버전업 및 메시지 변경 사항을 즉시 대처할 수 있는 기능
3. 여러 교환기의 다양한 출력 메시지를 용이하게 처리할

수 있는 기능
4 처리하고자 하는 메시지를 등록할 수 있는 기능
5 운영자가 parsing rule을 프로그램에 직접 적용할 수 있는 기능
6 정해진 parsing rule에 의한 메시지 자동 분석 기능
7 오류 메시지 재생성 기능
8 오류 메시지 처리 기능
9 운영자와 모듈간의 인터페이스 기능(Configurator)
10. 원하는 Log File의 사전 선택 기능
11. Log File의 수동 및 자동 삭제 기능

4. Message Gathering System의 결과

제안한 MGS 시스템은 4개의 Processor, 즉 MCP, MLP, MFP 및 Configurator로 구성된다.

MCP는 Terminal Server에 연결되어 Character 단위로 Message를 읽어들이며, 그 내용을 MLP로 전달하는 작업을 한다. 또한 여러 개의 Port 및 Device에서 출력되는 메시지들을 Message Queue로 전달하여, 교환기 하나의 Data뿐만 아니라 여러 개의 교환기 Data 처리도 가능하게 해준다. 그러므로 본 MCP는 교환기 수만큼의 Message Queue를 가지고 있어야 한다. 이에 대한 설정치는 Cofigurator를 이용해 운영자가 직접 System에 맞게 입력할 수 있다.

MLP는 MCP에서 받은 Data를 파일화하며, Configurator의 새로운 요구사항에 따라 새롭게 파일화할 수도 있다. 아울러 MFP로 Data를 전달한다.

MFP는 Configurator와 유기적으로 관계하여 동작하며, 필요한 장애 및 성능 Data를 분리, 가공할 수 있도록 해주는 주요 Processor이다. 즉 Message Logger로부터 읽어들인 Data를 String 형태로 만들고, 이 Data가 Configurator에서 등록한 Message인지 판별하며, 정형화된 형식에 맞게 Message를 분리한다.

위에 있는 세 가지 프로세서는 Configurator와 유기적인 관계를 가지고 있기 때문에, 데이터에 관계된 모든 요소를 운영자가 입력하여 사용할 수 있으며, MGS System이 기동되는 중에도 Configurator의 변경된 내용을 그 즉시 반영될 수 있도록 설계하였다. 이는 공유메모리를 사용하여 변수를 처리하는 방식으로 프로그램을 설계하였기 때문이다. 이 외에도 부수적으로 처리할 수 있는 모듈로는 메시지의 결함을 복원하는 Message Maker 모듈, 전체 Daemon Processor의 기동 상태를 감시하는 Monitoring 모듈, 그리고 Debugging 모듈 등이 있다.

이와 같이 수행한 결과, 그림 3.9와 같이 Configurator에서 등록한 메시지를 MFP는 교환기로부터 들어온 메시지 중에서 등록된 메시지를 구별하여 데이터베이스에 적재하고, 파일화 하였다.

그림 4.1은 MFP가 MLP로부터 들어온 파일을 Configurator의 조건에 맞게 파일화한 그림이고, 그림 4.2는 Configurator의 조건에 맞는 메시지를 MFP가 데이터베이스에 적재한 그림이다.

즉, 서로 다른 교환기의 메시지를 취합하여 운영자가 지정한 교환기의 특정메시지를 파일화 하고 데이터베이스에 적재한 그림으로서 운영자나 그 밖의 업무 담당 사용자는 하나의

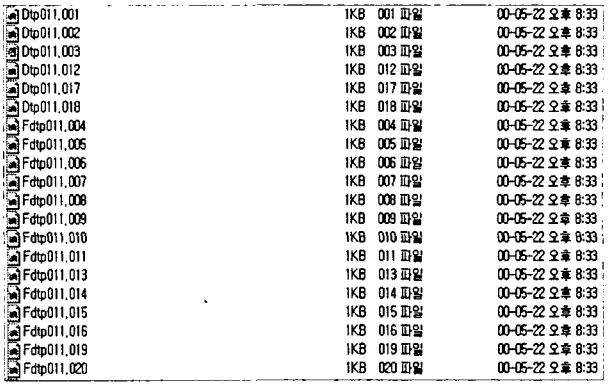


그림 4.1 Message Filter Processor의 출력1
Fig. 4.1 The output1 of Message Filter Processor

```
20:30:58 INSERT : <SESS>  KTI-GW,        97-12-17, 10:02:20, CG,     Minor,    TRUNK
20:30:58 INSERT : <SESS>  KTI-GW,        97-12-17, 09:05:42, TLWS,   Critical, H/W_FAULT
20:30:58 INSERT : <SESS>  KTI-GW,        97-12-17, 09:59:20, TRUNK,  Major,    S/W_FAULT
20:30:58 INSERT : <AXE10> "KR26/3PUSAN_IS", 010411,  0932,  APT,     Minor,    TRUNK
20:30:58 INSERT : <AXE10> "KR26/3PUSAN_IS", 010411,  0932,  IO-DEV,  Critical, H/W_FAULT
20:30:58 INSERT : <AXE10> "KR26/3PUSAN_IS", 010411,  1014,  APT,     Major,    S/W_FAULT
```

그림 4.2 Message Filter Processor의 출력2
Fig. 4.2 The output2 of Message Filter Processor

MGS에서 여러 교환기에서 출력되는 메시지를 확인할 수 있는 편리성이 있다. 부연하여 설명하면, 그림 4.1은 Configurator 조건에 의해 MFP에서 생성된 경보메시지, file명 Dtp011.xxx(파일명이 대문자 D로 시작)와 Configurator 조건과는 관계없는 경보메시지, file명 Fdtp011.xxx(파일명이 대문자 F로 시작)을 함께 보인 그림이며, 이 파일 중 Dtp011.xxx 파일만이 운영자가 망관리를 위하여 등록한 메시지이므로 TMN base에서 사용될 것이다. 즉, 교환기에서 MGS로 20개의 데이터를 입력으로 주입하였는데 그 중 Configurator에서 등록한 메시지 6개를 구분하여 파일명 Dtp011.xxx로 저장하였고, 그 외의 데이터는 Fdtp011.xxx의 형식으로 저장하였다. 그리고, 이중에서 운영자가 필요로 했던 데이터는 Dtp011.xxx 형식의 파일이므로 이 파일들이 TMN base에서 사용될 수 있다.

5. 결 론

본 논문에서는 교환기(AXE-10, 5ESS)에서 출력되는 장애 메시지를 취합하기 위하여, MGS에 대한 아직 확정되지 않은 ITU-T 권고[11]~[18]를 대신하여 임의의 메시지를 그림 3.9처럼 Configurator에서 장애 등록하였으며, 그 결과를 DB에 저장하고 파일화 하였다. 그리고 본 논문의 공신도를 높이기 위해 모든 교환기 등의 H/W와 공인 검증이 앞으로 남은 상태이며, 그렇게 하기 위하여 본 논문의 연구 방향을 일반적이고 융통성 있게 설계하였다.

메시지 처리 성능은 운영자의 요구에 의해 정의되는 메시지 Format을 각 장비의 전송 라인의 특성(X.25 : 10M Byte, TCP/IP : 10M Byte, RS422/232 : 9600 Byte)에 따라 최대한 처리할 수 있도록 설계하였다. 단, DBMS의 성능에 따라 약

간의 변동은 있을 수 있다.

메시지 처리 신뢰도는 운영자의 요구에 의해 정의된 메시지 형식을 각 장비의 전송 라인의 특성에 따라 최대한 처리하였으며, 오류 메시지 복원은 전송 라인 상에서 받아내는 모든 메시지를 전송 받아, 전송 받은 메시지의 오류를 판별하여 복원하였다.

또한, 본 논문의 MGS 시스템은 TMN을 기반으로 하는 망관리 시스템에서 메시지를 취합하는 Gateway 모듈로도 적용이 가능하도록 설계되었기 때문에, 교환기를 보유하고 있는 산업체에서는 본 시스템을 적극 활용할 수 있다.

앞으로 통신분야에서는 교환기가 차지하는 비율이 점점 커질 것이며, 또한 전송방식에서도 R2방식에서 C7방식으로 빠르게 변하고 있다. 그러므로 본 연구 논문 내용을 안정성 있는 제품으로 개발 완료하고, C7 link를 위한 DB와의 Interface까지 고려한다면, 교환기와 회선관리 시스템과의 통합을 이룰 수 있다.

본 연구에서의 앞으로 남은 과제는 교환기의 원시 메시지를 장애 메시지에만 국한하여 모의 실험하였는데, 이 외의 메시지에도 적용하고, 계속적인 연구·개발을 하여 TMN Platform 등의 추가 응용분야에서 기술 종주국인 미국보다 앞선 기술력을 확보해야 할 것이다.

감사의 글

본 연구는 2000년도 중소기업청 산학연 콘소시엄 사업과 숭실대학교 교내연구비 지원을 받아 이루어진 연구로서, 관계부처에 감사드립니다.

참 고 문 헌

[1] 김현우 외 1명, "전자 교환 시스템과 프로세서", 전자교환기술, 제 1 권, 제 1 호, pp. 34-44, 1986.6.

[2] 이헌, "Fault Tolerant Computing System", 전자교환기술, 제 1 권, 제 1 호, pp. 46-61, 1985.8.

[3] 이용균, "교환기 소프트웨어", 전자교환기술, 제2권, 제1호, pp.45-49, 1986.6.

[4] 한국통신 국제통신망 운영국, "국제망 관리 시스템 시설보강 운영자 지침서" 한국통신기술주식회사, 1997.12.

[5] 한국통신 국제통신망 운영국, "국제망 관리 시스템 시설보강 개발보고서" 한국통신기술주식회사, 1997.12.

[6] 한국통신 국제통신망 운영국 "5ESS Error Message", 1997. 12.

[7] 한국통신 국제통신망 운영국 "AXE-10 Error Message", 2001. 2.

[8] Stevens, "Unix Network Programming", Prentice Hall PTR, 1998.

[9] Keith Haviland, Dina Gray, Ben Salama, "Unix System Programming", Addison Wesley Longman Inc., 1999.

[10] Seungsik Shin and Chansoo Chung, "Development of Message Gathering System for digital electronic switching systems", ICCAS, SA07-4, October, 16~19, 2002

[11] ITU-T. Rec. X.711, "X.711 관리 정보 프로토콜 규격 - CMIP".

[12] ITU-T. Rec. X.721, "관리 정보의 정의".

[13] ITU-T. Rec. X.722, "관리 객체 정의의 지침".

[14] ITU-T. Rec. X.723, "관리객체기능".

[15] ITU-T. Rec. X.724, "상태관리기능".

[16] ITU-T. Rec. X.725, "경보보고기능".

[17] ITU-T. Rec. X.726, "사건보고관리기능".

[18] ITU-T. Rec. X.727, "저장제어기능".

저 자 소 개

신 승 식(申 承 湜)

1969년 1월 17일 생. 1993년 2월 호서대 제어계측공학과 졸업. 1997년 2월 숭실대학교 대학원 졸업. 1999년~현재 동대학원 박사수료

Tel : 02-901-7701, Fax : 02-901-6884

E-mail : threeess@mail.induk.ac.kr

정 찬 수(鄭 讚 壽)

1949년 8월 10일 생. 1972년 2월 서울대 전기공학과 졸업. 1980년 2월 동대학원 졸업(공학석사). 1987년 2월 동대학원 졸업(공학박사)

Tel : 02-820-0645

Fax : 02-817-7961

E-mail : chung@ee.ssu.ac.kr