Packet Switching에 의한 공중 Computer 통신망 개발 연구 -제1부 : KORNET 개요 및 NMC 개발

(Development of a Packet-Switched Public Computer Communication Network -PART 1 : KORNET Overview and Development of Network Management Center)

殷鍾官*, 李敬根**, 金華鍾*, 柳承文*, 朴淳*, 安秉英***,
李在天*, 李惺洙*, 趙東浩*, 趙熒濟*

(Chong Kwan Un, Kyung-Geun Lee, Hwa Jong Kim, Seung-Moon Ryu, Sun Park, Byung Young Ahn, Jae Chon Lee, Hwang Soo Lee, Dong Ho Cho and Hyung Je Cho)

要 約

본 논문은 packet 교환 방식의 computer 통신망인 KORNET 개발에 관한 4편의 논문중 제1부로서 KORNET의 전체적인 개요와 network management center(NMC)의 개발에 관하여 기술하였다. NMC는 operator dialogue를 통하여 network내 여러 소자의 상태를 감지하고 가입자와 network의 운용을 담당하며 routing의 관리를 담당하는등 network의 중추적 기능을 수행한다. 본 KORNET의 구현에 있어서 통신 protocol은 CCITT의 권고사항을 충실히 따랐으며 NMC의 응용 software인 operator interface, primary/secondary부, session layer와 packet level adaptor는 자체적으로 개발하였고, packet, link 및 physical level protocol은 Data General사에서 개발한 XODIAC X.25를 응용하여 개발하였다.

Abstract

This is the first part of the four-part paper describing the development of a packet-switched computer network named the KORNET. In this paper, we present the overview of the KORNET, and discuss various aspects on the development of the network management center (NMC). The NMC acts as a nerve center of the network, performing such functions as network monitoring, subscriber and network management and routing management using operator dialogues. In the implementation of the NMC, we have developed various application softwares that include operator interface, primary/secondary part, session layer and packet level adaptor. As for packet, link and physical level protocols, we have modified the XODIAC X.25 originally developed by Data General, Inc. All the network protocols we have developed comply completely with the CCITT recommendations.

*正會員, 韓國科學技術院 通信工學研究室
(Communications Research Laboratory, KAIST)
**正會員, 三星半導体通信(株) 派遣
(Samsung Semiconductor & Telecommunications Co.)
***正會員, 韓國데이타通信(株) 派遣
(Data Communications Corp. of Korea)

接受日字 : 1985年 6月 10日

I. 序 論

근래에 computer의 사용이 보편화 됨에 따라 computer들의 효율적인 사용과 data base의 공유 및 data

(※본 논문의 연구는 1982년 6월부터 1984년 12월까지 행한 과학기술처의 기업주도 특정연구 개발사업으로 과학기술처, 한국데이타통신(주), 금성전기(주), 삼성반도체통신(주) 지원으로 이루어졌음.)

의 자유로운 교환을 위한 computer와 computer사이, terminal과 terminal사이, 또는 computer와 terminal 사이의 data통신은 정보화 사회의 구현을 위한 필수적인 기술이 되었다. 최근에는 data 정보의 자유로운 교환뿐만 아니라 여러가지 형태의 정보들 즉, 음성, 전신, telex, facsimile, videotex 등을 data와 함께 하나의 통신망을 통해 통합 전송하는 integrated services digital network(ISDN)을 실현하려는 노력이 구미 각국에서 이루어지고 있다.[1]

Data통신의 기본이라 할 수 있는 packet 교환 방식에 의한 computer 통신망의 개발은 매우 방대하고 어려운 과제이기 때문에 구미 선진국에서도 단지 몇 개국 정도가 상당히 오랜기간 동안 많은 투자를 하여 이의 개발을 위해 힘을 기울여 왔다. 이러한 이유로 몇년전 까지만 해도 국내에서는 data 통신망을 자체 개발한다는 것이 극히 어려운 일로 간주되었었다. 그러나 다행히도 근래 data 통신이 정보산업은 물론 사회 전 분야에 걸쳐 생산성의 증대와 국가 발전에 막대한 파급 효과를 준다는 사실을 인식하게 됨으로써 국내에서도 공중 computer 통신망을 개발하게 되었다.

본 논문에서는 지난 3년 동안 한국과학기술원의 통신공학연구실을 주축으로하여정부 및 기업체들의 참여하에 성공적으로 일차 개발 완료한 공중 computer 통신망(이하 KORNET이라 칭함)의 개발 과정을 기술한다. 본 논문은 내용이 방대한 이유로 4부로 나누어서 제1부에서는 개요와 KORNET의 중추 역할을 하는 NMC에 관하여 기술하고, 제2부에서는 KORNET의 구조와 설계를 다루고 제3부와 4부에서는 packet 교환 기능을 담당하는 network node processor(NNP)의 software 설계 및 구현에 관하여 기술한다.

Network의 개발은 network를 종합 운용하는 NMC의 개발과 data의 교환 기능을 담당하는 NNP의 개발로 크게 나뉘어진다. NMC는 network dialogue 운용을 통하여 network과 통신을 함으로써 그 기능을 수행하게 된다. NMC는 network을 제어하는 기능을 갖고있으며 그 기능은 operator interface, primary/secondary (P/S)부와 상위 및 하위 level communication protocol 처리부의 세 부분으로 나눌 수 있다.

첫째로 operator interface는 사람과 기계 간의 통신을 위한 것으로 CCITT 권고사항 Z.311로부터 Z.341까지에 걸쳐서 정의된 man-machine language(MML)를 근간으로 하여 구현하고 있다. Operator terminal을 통해 입력된 여러가지 명령과 그 수행 조건이 이 operator interface를 통해 분석되며 아울러 그에 따른 응답도 이 module을 통해서 operator에게 제공된다.

둘째로 NMC의 primary/secondary부의 기능은 요금계산(charging), 운용상태의 측정(measurement), 기술적(technical) 기능으로 대별할 수 있다. 요금계산 기능은 call과 facility 사용에 대한 계산 기록을 모으고 분석하며, 운용상태의 측정 기능은 NNP의 facility를 제어하고 network 및 network 소자들의 상태를 측정하는 것을 말한다. 또한 기술적 기능은 network configuration을 제어하고 network과 network 소자들을 시험하며 이밖에도 이상 상태에 대한 보고를 모아서 기록한다. 이와같은 NMC의 P/S기능 중에서 KORNET에서는 일차적으로 network monitoring, network management, subscriber management 및 routing management와 같은 기본적인 technical 기능만을 구현하고 있다.[2]

세째로 communication protocol 처리부는 NMC와 NNP의 통신에 사용되는 부분이며 이를 통해 NMC의 명령과 NNP의 응답이 처리된다. 이 부분은 상위 protocol인 session layer protocol과 X.25의 packet, link 및 physical level protocol로 구성된다. 본 NMC에서 session layer는 Bell 연구소의 session layer protocol을 근간으로 하여 구현하고 있으며,[3] X.25 protocol은 Data General사의 XODIAC X.25를 이용하여 구현하고 있다.[4]

본 논문에서는 제Ⅰ장의 서론에 이어 제Ⅱ장에서는 KORNET의 전체 시스템 개요를 기술하고, 제Ⅲ장에서는 NMC의 기능 및 protocol을 설명한다. 또한 제Ⅳ장에서는 NMC의 구현 방법을 다루며, 제Ⅴ장에서 결론을 맺는다.

Ⅱ. KORNET의 개요

Packet 교환 방식에 의한 computer 통신망인 KORNET

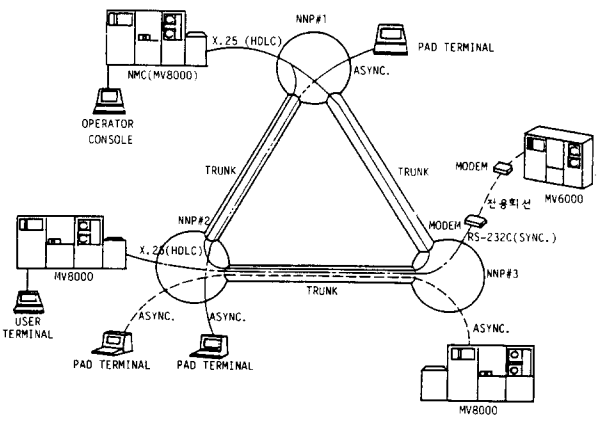

그림 1. KORNET의 3 node 구성도
Fig. 1. KORNET configuration with 3 nodes.

의 기능은 그림 1 에 나타낸 바와 같이 NMC와 NNP에서 행하여지는 역할로 대별된다. NMC에서는 network을 제어하며 network 감지와 운용, 가입자의 운용 및 routing 운용 기능 등을 수행한다. 즉 각 node의 link와 논리 채널 상태를 파악하고 가입자의 network 사용을 관리하며 각 node의 link를 연결 또는 차단한다. 또한 각 node의 routing table을 수집하고 변경한다. 한편 NNP에서는 boot-strapping 및 reconfiguration과 관련된 intrinsic 기능과 가입자가 network를 이용할 때 node가 service해 주는 on-line user 기능 외에 NMC의 network 감지 기능을 도와주는 동작 등을 수행한다. 이밖에도 개발된 NNP는 동기 또는 비동기형 computer와 terminal을 부착시킬 수 있다.

NMC의 hardware는 Data General사의 32bit computer인 MV/8000 시스템을 사용하였고, software는 그림 2 에 나타낸 바와 같이 MV/8000 computer의 PL/1 및 assembly language로 개발되어 있는 XODIAC X.25

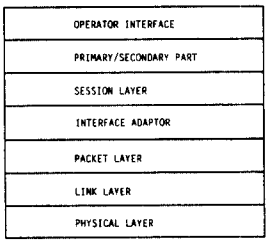

그림 2. NMC의 software 구조
Fig. 2. NMC software structure.

의 link 및 packet level을 수정한 다음에 packet level adaptor, session, primary/secondary와 operator interface 등의 상위 level protocol을 PL/1 language로 자체 개발하였다.

KORNET의 packet 교환 기능을 담당하는 NNP는 16bit microprocessor를 이용한 multi-task/multi-processor 기술과 C language를 사용하여 hardware와 software를 개발하였다.[5,6,7] 또한 NNP hardware는 network concentrator로도 쉽게 변형하여 사용할 수 있도록 modularity와 확장성을 고려하여 설계하였다. NNP 시스템은 그림 3 에 보인 바와 같이 크게 4개의 module로 구성되며 각각 고유의 일을 담당한다. Master central processing module (MCPM)에는 NNP를 종합적으로 운용하는 기능과 NMC로부터 내려오는 각종 명령을 해석하고 수행하기 위한 session 및 P/S 기능이 내장되어 있다. Line processing module-A (LPMA)에는 data circuit equipment (DCE) 또는 data switching exchange (DSE)로 동작하는 X.25 packet 및 link level 기능이 내장되어 있으며, LPMB에는 start/stop (S/S) mode DTE를 보조하기 위한 packet assembly/disassembly (PAD) 기능과 virtual call service를 위한 packet DTE/DCE기능이 내장되어 있다. 이밖에 common memory module (CMM)은 MCPM, LPMA, LPMB 등의 각 module 사이의 통신을 위해서 사용된다.

지금까지 개발한 NNP의 software 전체크기는 약 800 Kbyte에 달한다. MCPM에 230Kbyte, LPMA에 220 Kbyte, LPMB에 120Kbyte씩 내장되어 있으며, 이외에

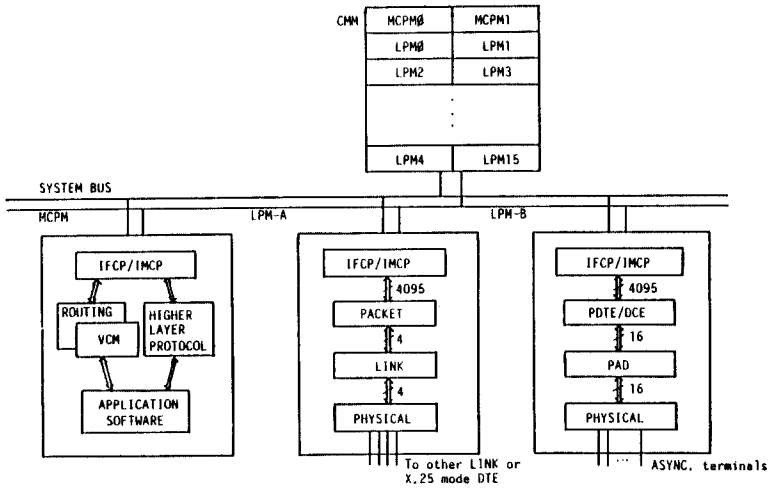

그림 3. NNP의 전체 software 구조
Fig. 3. Overall structure of NNP software.

개발된 시험용 프로그램이 약 230Kbyte 정도에 이르고 있다.

Ⅲ. NMC의 Protocol과 기능

1. Man Machine Interface의 Protocol과 기능

(1) Man-machine Interface Protocol

Network을 운용하기 위한 모든 동작은 operator에 의한 명령 즉, network command language (NCL) 입력으로 이루어지게 된다.[1]

사람과 기계간의 통신을 담당하는 man-machine interface의 protocol은 CCITT권고사항 Z.311에서 Z.341까지 수록된 man-machine language (MML)를 근거로 한다. 여기서 MML이란 stored program control (SPC) system의 동작, 유지 및 시험에 관련된 입력, 출력 및 제어 조치 등의 기능을 포함하는 언어이다. MML은 가입자가 배우기 쉽고 사용하기 편해야 하며 명령이나 응답의 표현이 알기 쉬워야 한다. 그리고 interactive mode 혹은 menu mode로 동작하거나 직접 입력이 가능하여야 한다.

한편 MML은 명령이나 제어 조치의 error발생으로 시스템 전체의 동작 중지를 야기 시키지 않도록 구성되어야 하며 입출력 장치에 따라 달라지지 않는 일반성을 지녀야 한다.

KORNET에서 구현한 operator interface의 MML은 Bell Telephone Manufacturing (BTM)사의 packet switching network DPS 25에서 구현된 NCL 중에서 technical 기능에 관한 명령과 응답 및 operator service를 위한 command line interpreter (CLI) 명령들이다.

(2) Operator Interface의 기능

NCL의 명령과 입력 변수를 받아들여 명령 및 응답 처리를 하는 operator interface는 terminal manager와 syntax analyzer, execution controller로 구분된다. Operator interface는 NMC의 P/S부와 operator사이에 위치하여 NCL의 해석과 수행을 담당하는 중요한 부분이다.

Operator는 network 운용을 위해 여러가지 NCL 명령을 내린다. 이때 명령은 terminal manager를 통해 인식되며 syntax analyzer에서 분석된다. 만일 옳지 않게 명령이 입력 되었으면 잘못 되었다는 내용을 만들어 error message를 terminal에 내보낸다. 옳게 들어온 명령인 경우에는 변수명을 차례로 내보내어 변수값을 받아들이고 분석을 하게 된다. 명령과 그에 따른 모든 변수가 옳게 입력됨이 확인되면 명령과 변수는 적당한 형태로 묶여져 execution controller로 보내진다. 여기서는 명령의 순서나 전송부의 상태가 제대로 되었는지를 검사하고 모든 것이 정상임이 확인되면 적당한 형태로 formatting하여 P/S부로 전송한다. 한편 이의 내용이 network에서 수행되면 P/S부로부터의 응답이 execution controller를 통해서 terminal manager로 전달되며 terminal manager에서는 그 내용을 번역한 후 operator terminal로 응답에 해당하는 string을 내보낸다.

2. NMC의 Primary/Secondary 기능

Operator의 명령은 operator interface에서 해석되어 수행 형태로 바뀌어 P/S부로 넘어온다. Primary subsystem은 이들 명령의 대부분을 수행한다. 일부 명령은 session을 통해 NNP와 NMC간의 transaction을 setup 시키지만 일반 technical 기능의 명령은 subscriber management, network management, network monitoring 및 routing management 등으로 분류된 다음에 수행된다. 이들 명령 중에는 NMC자체적으로 처리할 수 있는 것도 있고 NNP의 submodule에서 수행할 수 있는 것이 있다. NMC에서 하는 일이란 결국 이들 technical 기능의 명령을 network에 내리고 그 결과를 받아서 처리하는 것이 된다.

(1) Subscriber Management

NMC는 subscriber를 결정하는데 필요한 모든 요소 즉 link, permanent virtual circuit (PVC), subscriber등을 개설하거나 수정 또는 제거시키는 동작에 관련된 제반 조치를 담당한다.[9]

여기에는 많은 변수들이 입력 조건으로 제공되어야 하며 operator는 NMC의 table과 NNP에서 제공되는 응답으로 부터 그 수행 결과를 확인할 수 있게 된다.

(2) Network Management

Network에 있어서 transmission resource의 운용, 상태 변화의 비교검사 등을 위한 NMC의 동작을 network management라 한다. 여기서 transmission resource의 운용이란, PVC나 virtual circuit (VC), link, line의 개설, 취소를 비롯한 traffic에 대한 처리를 말한다.

(3) Network Monitoring

Network monitoring 기능은 operator에게 network 소자의 상태 변화를 알려 주는 것으로 NNP에 의해 검출되어 NMC로 보내진 것과 NMC에 의해 검출되는 것이 있다. Operator는 이러한 event나 alarm에 대해 network에 적절한 조치를 취한다.

(4) Routing Management

Network 내에서 data의 path를 결정하는 routing은 routing table의 작성과 수정, call packet 및 data packet의 routing으로 그 일의 범위를 정할 수 있다.[10]

분산 적응 처리 방식의 routing을 구현한 KORNET에서는 이의 대부분을 NNP에서 수행하며 NMC는 table의 수집과 변경을 담당한다.

3. Session Layer Protocol

Session은 P/S 기능이나 presentation service 같은 상위 layer entity간의 연결로서 다음과 같은 service를 제공한다.

- 상위 layer 사이의 통신을 위한 통화로를 개설하고 제어하며 종결시킨다.
- Data 전송중 end-to-end 제어와 synchronization 의 기능을 제공한다.

상위 layer로부터 session 개설에 대한 명령을 받으면 session layer는 X.25에 call을 명령하고 virtual circuit이 개설되기를 기다린다. X.25로부터 개설에 대한 응답이 오면 예약되어 있는 session의 channel 과 virtual circuit을 연결시킨다. 이렇게 함으로써 session 개설을 위한 준비가 완료되며 이때부터 session message가 교환된다. 한편 session layer는 flow control을 위하여 resynchronization option을 선택할 수 있다. 이 option이 선택되면 송신순열수 m(s) 및 수신순열수 m(r) 이 각각의 session protocol data unit (SPDU) 의 header에 포함된다. 이때 end-to-end의 acknowledge는 sliding window 방법에 의해 행해진다.

4. Lower Layer Protocol

NMC에서 채택하는 lower layer protocol은 packet 교환 방식의 공중 통신망에 대한 규약인 CCITT X.25의 packet, link 및 physical level protocol을 근거로 하였다.

Packet level protocol은 확실하고 효율적인 data 전송을 위하여 다음과 같은 기능을 제공한다.

- Multiplexing - 다중 data stream을 가능케 하는 기능.
- Flow control - 각 data stream에 대하여 송수신측 DTE 사이의 data flow를 조절하는 기능.
- Error control - packet level의 error를 발견해내는 기능
- Reset과 restart - error가 확인되었을 때 packet level에서의 통신로를 reinitialize하는 기능.

위와 같은 기능의 수행을 위해 packet level protocol은 논리 channel 송수신 순열수, data 및 control packet에 대해 정의한다.

또한 link level protocol은 frame의 전송에 필요한 link를 set-up하거나 frame이 올바로 송수신 되도록 하기 위해 요구되는 사항들을 규정한다. NMC의 link level은 packet level과 마찬가지로 CCITT의 X.25권고사항을 기본적으로 구현하되 XODIAC에서 제공하는 X.25를 근간으로 하였다. Link level에서의 link access 과정은 일반적으로 asynchronous response mode (ARM)로 동작하는 link access procedure (LAP) 와 asynchronous balanced mode(ABM)로 동작하는 balanced mode link access procedure (LAPB)가 있다.

LAP에서는 통화가 이루어지는 양국 사이에 master와 slave가 미리 정해져 있어 slave는 master 측의 polling이 있을 때에만 응답할 수 있도록 되어 있다. 한편 LAPB에서는 양국이 서로 대등하므로 full duplex 채널을 통하여 항상 정보의 교환이 가능하다.

현재 대부분의 공중통신망은 LAPB의 규정을 따르고 있으므로 본 KORNET 개발에 있어서도 LAPB의 protocol을 구현했다.

Physical level protocol은 자국의 DTE와 상대 DTE가 전송로를 통해 서로 정보를 교환할 때 각각의 정보를 나타내는 bit이 전송로를 통해 올바로 전달되도록 하기위한 규약이다. physical level은 hardware와 밀접히 관련하여 각 bit의 전압, 속도(baud rate), 전송방향, pin connection 등을 정의한다.

CCITT 권고사항에서는 X.25에 의한 synchronous communication에서 physical connection 방법으로 X.21을 따를 것을 권장하고 있다. 이 X.21에는 synchronous 방식의 terminal과 packet 및 circuit교환 방식의 공중통신망을 연결하기 위해 필요한 사항들이 기술되어 있다. 현재 대부분의 통신장비들을 interface하는 데에는 EIA의 RS-232C가 사용되고 있다. 반면에 X.21은 RS-232C보다 훨씬 빠른 속도로 bit을 전송할 수 있게 할 뿐 아니라 pin connection이 간편하다. 또한 IBM의 SNA와 같은 layer형태를 갖는 다른 공중 data 통신망과의 호환성을 보장한다는 큰 장점을 가지고 있다. 그러나 아직까지 X.21을 만족시키는 LSI chip들의 개발이 늦어지고 있어서 X.21의 사용은 어려운 상태이다. 이 때문에 KORNET 시스템에서는 synchronous line의 physical interface로서 RS-232C에 compatible한 CCITT X.21bis권고사항을 구현하였다.

Ⅳ. NMC의 구현

1. NCL 및 Operator Interface의 구현

Operator가 network으로부터 상태를 파악하고 여러가지 조치를 취해 주기 위해 NCL을 사용하는데 이때 명령과 변수들의 입력은 interactive mode로 operator와 terminal 사이에 이루어진다. 우리가 구현하는 NCL은 BTM의 DPS25 network에서 수행하는 명령들을 선별하여 구현하였으며 수행상의 조건과 동작 내용은 다음과 같다.

- Network operation을 위한 NMC의 입력과 결과에 대한 응답은 NMC내의 한 operator terminal이 담당한다.
- Network 수행 프로그램을 가동시키면 prompt 로서 명령에 대한 sequence counter와 '*)' 이 나타나며 이후에 NCL 명령을 입력시키게 된다.
- Prompt 뒤에는 NCL 명령과 service 명령만을 입력시킨다. Network 명령의 경우 이 명령에 해당하는 변수 이름이 차례로 terminal에 나타나며 이때 operator 는 변수값을 입력시킨다. 만일 변수값이 허용범위를 벗어나게 되면 그 명령과 변수의 입력 후에 error message가 terminal에 나타나며 그 명령은 취소가 된다. 한편 service 명령은 operator가 편리하게 이용할 수 있도록 network 명령의 이용법, 응답기록 등을 구현한다.
- 입력과정에 있어 수정, 삭제 등의 편집기능은 NMC로 사용하는 MV/8000의 모든 편집기능을 이용할 수 있도록 한다.
- Operator의 명령이 지정된 NNP 또는 NMC 자체에서 수행되고 나면 명령에 해당하는 character 형태의 규격화된 응답이 NMC operator에게로 보내진다. 이외에도 operator는 network 소자의 상태 변화나 failure가 탐지 되었을 경우에도 NNP로부터 응답을 받게 된다.

Operator interface는 결국 명령인식과 명령처리 그리고 응답표시의 3 단계 작업을 하게 되는데 이는 Ⅲ장에서 언급한 syntax analyzer와 execution controller 및 terminal manager가 각각 나누어 담당하게 된다. Operator interface의 처리과정이 그림 4 에 그려져 있다.

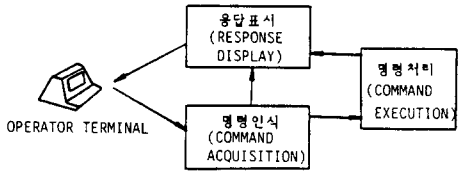

그림 4. Operator interface의 처리과정
Fig. 4. Command processing in NMC operator interface.

한편 KORNET에서 사용되는 주요 명령은 표1 과 같은데 이 NCL들은 operator interface에서 인식되어 network에서 처리되며 그 응답은 operator terminal로 보내어 진다.

2. Primary/Secondary부의 구현

Operator interface의 execution controller로부터

표 1. KORNET에서 사용되는 주요 NCL
Table 1. The NCL used in KORNET.

명령 계열	NCL 명령	내 용
Session 명령	SE-BGN	Session을 개설함
	SE-MOD	Session 구분의 변경
	SE-PVC	PVC session을 개설함
	SE-MSG	Operator session 간의 메시지 교환
	SE-END	Session을 종료함
Subscriber 운용명령	SA-PVC	가입자간 PVC의 개설
	SD-PVC	가입자간 PVC의 제거
	SM-PVC	가입자간 PVC의 변수 변경
Routing 운용명령	RO-TBL	NNP routing table의 수집
	RO-MOD	Routing table의 수정
Network 감지명령	SQ-LNK	Link 상태의 감지
	SQ-PVC	PVC 상태의 감지
	SQ-PVA	PVC 개설의 총량 감지
	SQ-LCH	논리 channel의 상태 감지
Network 운용명령	TC-CLK	Link의 연결
	TC-DLK	Logical link를 끊음
기타 service 명령	HELP	각종 display 제공
	COMMAND	기록 file의 검사
	NCL	NCL 명령 해설

명령과 변수들을 담은 block(BAO+)이 NMC의 P/S부의 queue로 입력되면 명령에 대한 분류가 이루어진다. 이때 subscriber management, network management, network monitoring 및 routing management에 해당되는 process overlay가 각각 불리어 진다.

한편 각각의 process overlay는 communication process monitor의 제어를 받는데 명령에 따라 session 개설에 들어갈 것인지, data 전송 상태로 이끌 것인지, call access만을 할 것인지를 결정한 뒤에 명령과 내용을 상대편과의 규약에 맞추어 session layer로 내려보낸다. 한 BAO에 대하여 응답(BEO++)이 생성되면 P/S부의 기억장치에 명령과 순서번호만 남기고 나머지 사항 즉, 변수 및 응답 내용은 file에 기록한다. 그림5 는 P/S부의 구조를 나타낸다. BEO queue에는 수행error, timeout 등과 명령을 network 에서수행하고 돌아오는 코드화된 정상응답 들이 쌓이는데 비정상응답에 대한 조치는 message를 받은 operator가 취하게 된다.

NMC에서 필요에 따라 주기적 혹은 비주기적으로 발생하는 message는 그림 6 의 flow를 따라 NNP의P/S로 전달된다. NMC의 P/S부에서 message의 각종 변수가 결정되면 NNP와 통신할 하나의 session이 개설된다. Session 개설과정에서는 network의 제어 정보를 지니고 있는 P/S의 data message와는 별도로 약

+BAO : operator command execution request block
++BEO : operator command execution reply block

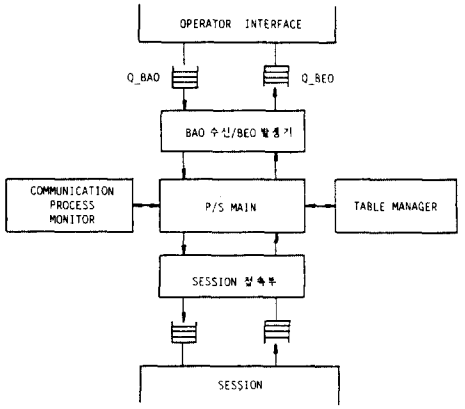

그림 5. Primary/secondary부의 구조
Fig. 5. Structure of primary/secondary part.

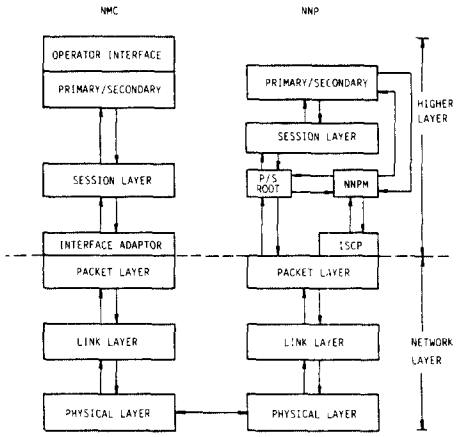

그림 6. NMC-NNP 사이의 protocol 구조
Fig. 6. Internal protocol between NMC and NNP.

속된 내부 message를 이용하여 정보를 교환한다. NNP의 P/S는 전달된 message를 해석하고 그 내용에 따라 적절한 조치를 취하여 NMC의 P/S부로 응답 message를 전송하는데 이때 필요한 정보는 NNP의 각종 facility를 제어, 운용하는 module인 network node processor manager (NNPM)으로 부터의 논리 channel에 관한 정보와 routing에 관한 정보뿐만 아니라 NMC로부터 system setup시 초기화된 변수 등이다. 한편, network 운용에 필요한 각종 변수는 필요에따라 table화 되어있다.

이상의 기능을 요약하면 NNP의 session과 packet의 변환 module인 root는 P/S에 message를 전달하거나 P/S의 message를 NMC로 전송하는 역할을 담당하며 NNP의 P/S는 NNPM으로 하여금 NMC의 명령을 수행하도록 하고 그 결과를 NMC의 P/S부에 보고하는 역할을 한다. 그림 7 은 NMC의 P/S부에서 message가 처음 발생하였을 때 상위 layer간의 message의 흐름을 보여준다.

3. Session Layer의 구현

Session layer의 software 구조는 그림 8 과 같이 나타낼 수 있다. 4 개의 외부 queue는 변수 table 들과 함께 layer 사이의 정보 교환에 사용된다. 정보를 다른 layer로 보낼 때 함께 보내져야 할 자세한 변수들은 interface table에 의해 전달된다. Data 전송의 요청이나 보고를 하는 경우에 이러한 변수들은 data에 관한 table (T-H-DATA, T-ST-DATA)을 통해서 전달되며 제어를 위한 신호가 교환되는 경우에는 T-H-CONTROL 또는 T-ST-CONTROL의 table을 통해서 전달된다.

상대편으로 보내질 data는 상위 level data buffer

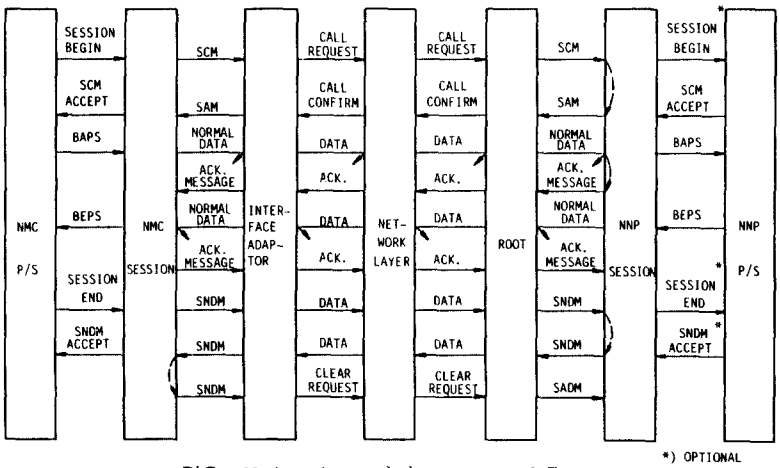

그림 7. Higher layer간의 message 흐름
Fig. 7. Message flow between higher layers.

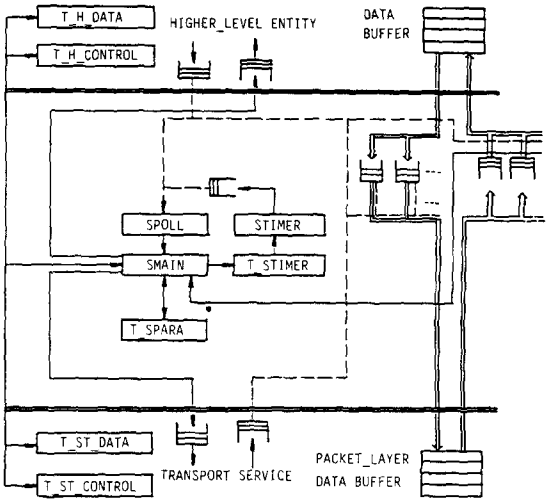

그림 8. Session layer의 software 구조
(= data flow, — information flow, - - - polling sequence)
Fig. 8. Session layer software structure.

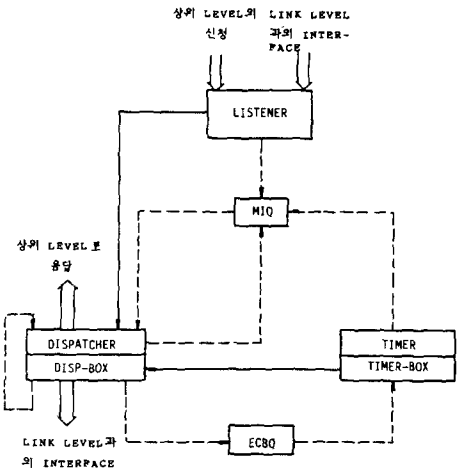

그림 9. Packet level software 구조
Fig. 9. Packet level software structure.

로 부터 읽혀진 후 적당한 길이(SPDU size)로 나뉘어 내부 queue(Q-SD)로 보내진다. Q-SD의 data들은 session protocol에 의해 적당한 session header가 붙여진 후 transport service로 보내져 전송된다. 한편 상대편으로부터 수신된 data는 packet level buffer로 부터 내부 queue(Q-SU)로 전달되고 논리적으로 연결된 일련의 SPDU들이 모두 도착하면 상위 level의 data buffer로 올려 보내진다. 그림 8에서 SPOLL과 SMAIN은 한 *process* 내의 두 *routine*인데 SPOLL은 queue들을 polling하여 사건이 일어났으면 그 사실을 SMAIN에 알려 적당한 처리를 하도록 한다. 새로운 채널의 등록이나 말소, 상태의 변화, 여러가지 변수값과 그 변화 등은 T-SPARA의 table에 기록된다. STIMER는 SMAIN이나 SPOLL과는 별도의 process로서 T-STIMER를 통해 SMAIN으로부터 명령을 받으며 time out 되었을 경우 Q-STIM과 SPOLL을 통하여 SMAIN에 보고한다.

4. NMC의 X.25 구현

(1) Packet Level과 그 Interface의 구현

NMC의 packet level은 CCITT 권고사항 X.25를 따르고 있는 Data General사의 XODIAC X.25를 이용하고 있다. 이 packet level은 MAIN과 LISTENER, DISPATCHER 그리고 TIMER로 구성되어 있는데 그림 9는 이를 나타내고 있다. 여기서 MAIN은 나머지 세 개의 task를 만들고 packet level에서 필요한 environment를 만들어 준다. LISTENER는 상위 level에서 오는 신청 또는 link level에서 오는 신청을 받아들이고 DISPATCHER로 이를 넘겨주는 역할을 하는데 이때에 master input queue(MIQ)를 이용한다. 또한 DISPATCHER는 상위 level, link level 그리고 TIMER에서 보내오는 신청들의 종류를 분석하고 적절한 일을 수행하는 task로서 protocol을 처리하는 기능을 갖고 있는데 DISP-BOX라는 mailbox를 통해 신청을 받고 MIQ로부터 그 내용을 access한다. 또한 DISPATCHER task가 timer를 사용할 때는 event control block(ECB)를 만들어 ECBQ로 보낸 뒤 mailbox인 TIMER-BOX를 이용하여 TIMER task가 이를 처리하도록 한다.

XODIAC X.25는 NMC로 사용되는 MV/8000 computer의 operating system인 AOS/VS와 밀접한 관계가 있으며[11] 이를 이용하기 위해서는 interprocess communication(IPC) system call을 사용하여야 한다. 이를 위해 packet level adaptor라는 module을 개발하였는데 이것을 사용하여 session layer가 XODIAC X.25를 이용할 수 있게 하였다.

Adaptor는 그림10에서 보는 바와 같이 ADPT, SEND, AWAIT 및 RECEIVE의 4개의 task로 되어 있는데 이들은 다음과 같이 운용된다. ADPT task는 adaptor module이 처음으로 사용될 때 수행되는 것으로 나머지 task를 구동시킨다. 또 SEND task는 상위 level에서 내려온 명령을 수행하는 task로서 관련된 IPC system call을 사용하며 자신이 수행한 일의 결과를 상위 layer로 보고하도록 되어 있다. 그리고 RECEIVE task는 data packet이나 clear request packet 등을 수신하거나 link level의 이상을 보고하기 위해서 사용된다.

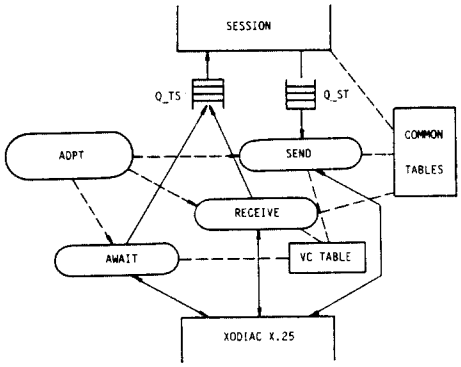

그림10. Packet level adaptor의 software 구조
Fig. 10. Software structure of the packet level adaptor.

끝으로 AWAIT task는 incoming call packet을 수신하고 이를 처리한다.

이밖에도 XODIAC X.25와 KORNET의 NNP에 구현되어 있는 packet level protocol이 잘 맞도록 하기 위해서 timer, 최대 packet 길이 등의 변수값이 조정되고 CCITT X.25 packet layer의 option으로 규정되어 있는 일부분의 기능이 수정되거나 변경되었다.

(2) Link Level의 구현

Link level의 software는 크게 7개의 task로 이루어진다. 이 task들은 link level protocol의 16개 state를 중심으로 상위인 packet level 및 하위인 physical level과의 입출력이 빠르고 확실하게 이루어질 수 있도록 하는데 필요한 것들이다. 이들 task간의 통신은 다른 layer와 마찬가지로 여러 queue와 table로 이루어지게 된다. 그림11은 link level의 software 구조를 보인 것이다.

Link level의 운용상태는 operator에 의해 감시되며 변화시킬 수 있다. Operator에 의한 명령은 host command processing task에 의해 분석되며 그 결과는 transmit task로 전해진다. Link level의 입출력은 transmit와 transmit interrupt task, receive 및 receive interrupt task에 의해 수행된다. 이 중 transmit task와 receive task는 입출력되는 frame의 종류를 분석, 선택하여 읽거나 쓰며 각 frame에 기입되는 여러 상수를 수시로 update 시켜준다. 결국 link level의 state 변환은 이들 두 task에 의해 다루어진다. 한편 transmit interrupt task와 receive interrupt task는 완전히 구성된 frame을 driver에 전달해 주거나 driver로부터 읽혀진 frame을 receiver task로 전달해 주는 일을 맡으며 주로 interrupt로 service되는 routine이다. Link level protocol을 위한 timer는 time out task에 의해 관장되며 driver의 각종 제어신호는 modem control task에 의해 제어된다.

(3) Hardware 및 Physical Interface의 구현

KORNET 개발에 있어 NMC로 사용되는 PDTE는 Data General사의 MV/8000 computer를 채택하였다. NNP와 NMC의 interface를 위해서는 MV/8000 host에 동기식 line을 사용할 수 있도록 하는 interface board가 필요한데 그것이 intelligent synchronous controller

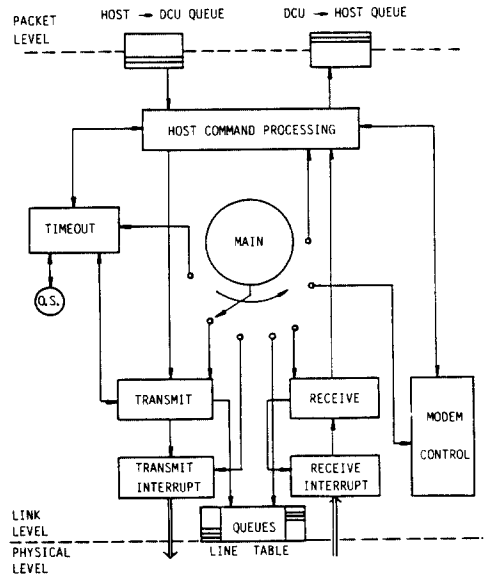

그림11. Link level의 software structure
Fig. 11. Link level software structure.

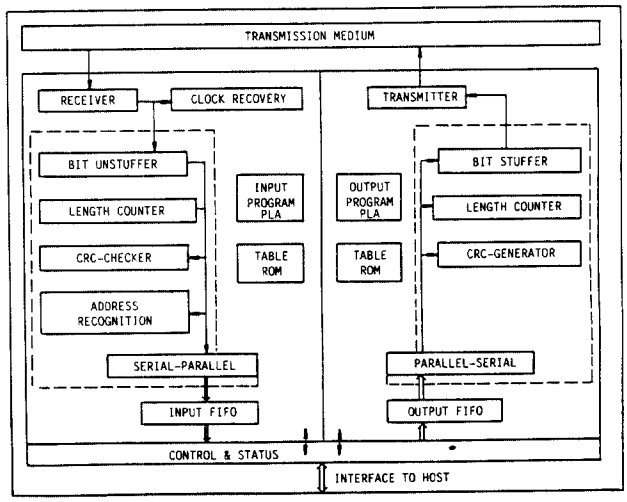

그림12. ISC의 구조
Fig. 12. ISC structure.

(ISC) board이다. 이 ISC board는 고속의 bit전송을 위하여 그림12와 같이 입력 및 출력 부분이 별도로 설계되어 있다. 이들의 신속한 service를 위해서 입출력을 위한 간단한 프로그램과 여러가지 변수들이 PLA, ROM에 저장되어 있으며 이들은 MV/8000 의 system generation시에 수정될 수 있다.

5. NMC의 Integration 및 Test

앞절에서 각기 설명한 software module을 바탕으로 부분적인 test와 integration을 거쳐 NMC를 그림13과 같이 구성하였다. NMC 개발에 있어 test의 단계는 module test, module integration test, loopback test, emulation test 및 실제 NNP와의 total integration test로 나눌 수 있다. PL/1 coding된 각 task들은 MV/8000 computer의 debugging facilty를 이용하여 debugging을 끝냈으며 이때 동작중의 queue와 table을 monitoring 함으로써 test할 수 있었다. Module integration 및 loopback test역시 위의 방법과 아울러 여러가지 입력 sequence에 대한 출력을 살펴봄으로써 확인하였다. Emulation test 단계에서는 protocol tester인 Tektronix사의 TEK834를 이용하였는데 TEK834를 DCE로, MV/8000을 DTE mode로 하여 session개설에 따른 call set up과 data 전송과정 및 clear과정을 test scenario에 따라 수행시켜 그 동작을 확인하였다. 동일한 scenario로서 NNP측도 test한 후 NMC와 NNP간의 total integration을 수행하여 NMC가 network 소자들을 정상적으로 동작시키고 NNP가 제대로 응답하는지의 여부를 test할 수 있었다.

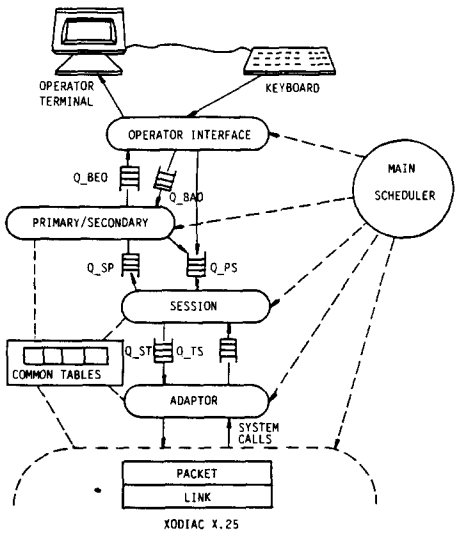

그림13. NMC의 전체 software 구조
Fig.13. NMC overall software structure.

NMC와 NNP의 접속과 아울러 XODIAC에서 제공하는 virtual terminal agent(VTA)와 file transfer agent(FTA) 등 network 응용 utility의 KORNET을 통한 활용도 같은 방법을 통해 이루어졌으며 정상적으로 동작함을 확인하였다.

V. 結 論

KORNET은 그 개발에 있어 network simulation을 통하여 이론적으로 뒷받침하고 있으며[5] 점차로 가입자수를 늘리거나 network의 여러 기능을 개선할 경우에 대비해서 확장에 따른 융통성을 부여하였고 CCITT의 권고사항을 충실히 따라 설계하였다. 특히 NMC와 NNP사이의 통신 protocol은 Bell 연구소의 session layer protocol과 CCITT 권고사항 X.25에 정의된 packet 교환 방식의 공중 data망의 protocol을 따라 구현하였으며 또한 NMC의 man machine interface도 CCITT 권고사항 Z.311부터 Z.341에 정의된 MML과 현재 운용중인 DACOMNET의 NCL을 고려하여 구현하였다.

본 NMC 개발에 있어 operator interface, P/S부, session layer 및 packet level adaptor는 자체적으로 개발하였고 X.25의 packet, link 및 physical level의 기능은 MV/8000 system에서 제공하는 network용 프로그램인 XODIAC을 응용하여 개발하였는데 NMC와 NNP의 접속을 위한 protocol 변환 작업이 통합 단계에서 수행되었다.

통합접속과 시험이 성공적으로 끝난 KORNET에서는 network을 통한 computer간의 통신, terminal간의 통신, computer와 terminal간의 통신이 원활하게 수행되었고, 유용한 응용 software로 virtual terminal agent, file transfer agent 및 data base 운용 system 등이 개발되었다.

參 考 文 獻

[1] G.S. Bhusri, "Considerations for ISDN planning and implementation," *IEEE Commun. Magazine*, vol. 22, no. 1, pp. 18-32, Jan. 1984.

[2] DACOM, *DPS25 Packet Switching System Training-PTC20 System Architecture NMC I-II*, pp. 108-217, 1983.

[3] AT & T, *Operating System Network Communications Protocol Specification.* BX.25 Issue 3, pp. 5-1 to 5-42, Director-Purchased Products, June 1982.

[4] Data General Corp., X.25 Protocol (AOS

and AOS/VS) User's Manual, 1982.

[5] 은종관외, "Packet Switching에 의한 Computer 통신망 개발," 1 차년도 최종보고서 한국과학기술원 통신공학연구실, 1983.

[6] 은종관외, "Packet Switching에 의한 Computer 통신망 개발," 2 차년도 최종보고서, 한국과학기술원 통신공학연구실, 1984.

[7] 은종관외, "Packet Switching에 의한 Computer 통신망 개발," 3 차년도 최종보고서, 한국과학기술원 통신공학연구실, 1985.

[8] BTM Corp., *Packet Switching System DPS 25,* Operating and Maintenance Course, pp. 17-160, 1983.

[9] A. Rybczynsky, "X.25 interface and end-to-end virtual circuit service characteristics," *IEEE Trans. Commun.*, vol. COM-28, pp. 500-510, Apr. 1980.

[10] M. Schwartz and T.E. Stern, "Routing techniques used in computer communication networks," *IEEE Trans. Commun.*, vol. COM-28, pp. 539-552, Apr. 1980.

[11] Data General Corp., *XODIAC Network Management System,* Guide for Managers and Operators, 1981.