

《特別寄稿》

OSI 표준화를 위한 CLNP 개발과 하부망 접속에 관하여

- On CLNP Implementation and Interface with Subnetworks for OSI Standardization -

김병철 · 민상원 · 정 해 · 이희섭 · 변옥환 · 신병철 · 은종관
(한국 과학 기술원 전기 및 전자과)

■ 차 례 ■

I. 서 론	Ⅲ. CLNP 접속 기술
Ⅱ. CLNP의 기능 및 소프트웨어 구조	Ⅳ. 결 론

I. 서 론

정보 통신 사회로 발전하는 현대 사회에서는 정보 처리 분야와 데이터 통신의 결합에 따른 정보의 공유 및 신속한 전달의 필요성이 증가하게 되었고 개인의 요구에 맞는 가공 및 저장이 이루어지게 되었다. 특히 교육, 연구분야에서는 외국과 국내 일부에서 이미 상용 망을 통한 정보 전달이 활발히 이루어지고 있으며 이에 관련된 연구가 많이 진행 중이다. 이 경우 이러한 망들간을 연결할때 겪는 이 기종 컴퓨터 통신 구조간의 호환성 이라는 문제점과 각 group 내의 운용 방식인 Local Area Network(LAN) 규약이 이들을 연결하는 공중망의 규약 및 동작과 서로 다르다는 문제점 때문에 어려움이 따르게 된다[1].

이에 International Standards Organization(ISO) 및 International Consultative Committee for Telegraphy and Telephony(CCITT)에서는 표준화에 대한 개발의 기초로 7 계층의 구조를 갖는 기준 표준 모델을 개발하였으며 국내에서도 Open System Interconnection(OSI) 표준 모델 구현을 위해 GOSIP-K에서 국내 표준화를 추진하였다[2]. 현재 교육 연구망에서 사용하는 통신 규약은 OSI의 표준안이 아닌 TCP/IP로서, 외국에서 사용되는 전산망이 OSI 표준안에 따른 전산망으로 이전하는 경향에 맞추어 국내 교육 연구망도

TCP/IP를 점차 OSI 표준안에 따른 통신 규약으로 이전하여야 할 것이다.

OSI 표준안에 따른 통신 규약중 계층 3인 망 계층은 가장 복잡한 계층으로 TCP/IP의 IP에 해당하는 계층이며 상용망 중 X.25의 Packet Level Protocol(PLP)이 여기에 해당된다. OSI 표준안에서는 망 계층을 2가지 방식으로 정의하고 있는데 하나는 연결성 방식(connection-mode)인 X.25 PLP이고 또 다른 하나는 Connectionless Network Protocol(CLNP)로서 비 연결성 방식(connectionless-mode)이며 ISO 표준안 8473에서 정의하고 있다[3]. 국내 GOSIP-K에서도 망 계층 모델로 CLNP를 정하였고 외국에서도 CLNP로 이전하는 추세이다.

이에 본고에서는 OSI 표준화 환경 구축 및 LAN/WAN 게이트웨이 구현의 기초 연구 단계로서 CLNP를 구현하고 이의 내용과 구조를 기술하였으며 TCP/IP를 제공하는 ethernet LAN 환경및 OSI 표준화에 따른 X.25 공중망과의 접속 연구를 기술하였다.

이 연구의 수행을 위한 CLNP와 하부망 간의 연동 소프트웨어 구성을 위해서는 LAN/WAN 환경 구축이 필요한데 LAN용으로는 EXOS 보드가 사용되었고 X.25 망 규약 구현을 위해서는 Adax card가 사용되었다. 소프트웨어 개발 환경으로는 Microport Unix System V.3.0의 운영 체제가 사용되었고 시험서비스를

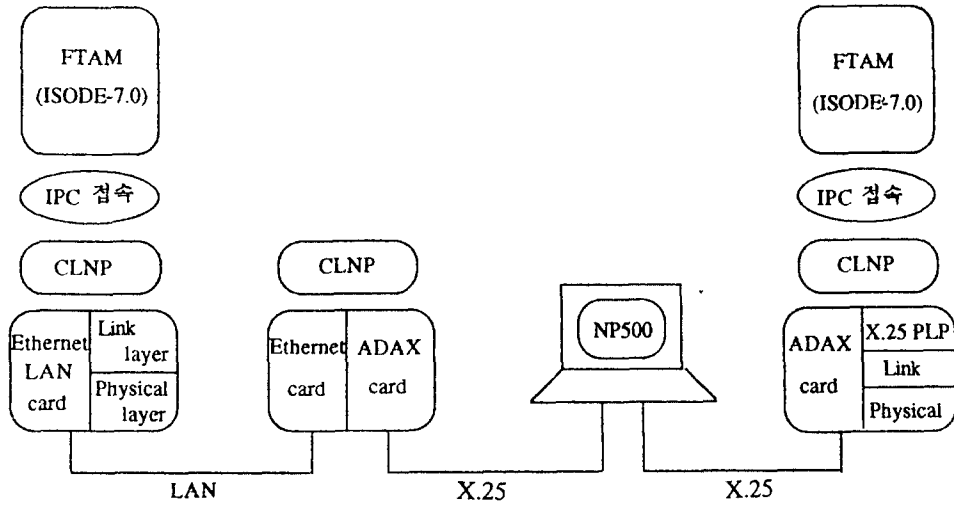


그림 1. 시스템 모델.

위한 전송 계층용 소프트웨어로는 ISODE 7.0 전송 계층이 사용되었다. ISODE 프로그램은 OSI 표준 환경의 대표적인 개발 도구로서 Berkely Unix, AT & T Unix, HP-UX, Ridge Operating System 등의 다양한 운영 체제에서 실행가능하다[4]. 한편 DTE 방식으로 동작하는 Adax card에 clock을 제공하며 DCE 방식 서비스를 제공하기 위해서 Node Processor 500(NP500)이 이용되었다. 전반적인 시스템 개요 모델은 그림 1과 같다.

본 고에서는 CLNP 연구 및 개발이 2장에, 하부망인 LAN, X.25와의 접속이 3장에 기술되었고 마지막으로 결론을 맺는다.

II. CLNP의 기능 및 소프트웨어 구조

ISO 8348에서는 OSI 망 서비스인 Connection Oriented Network Service(CONS)를 정의하고 있으며 ISO 8348/AD1에서는 Connectionless Network Service(CLNS)를 정의하고 있다. 연결성 방식 전송은 데이터를 전송하기 전에 연결이 설정되어야 하고 전송이 끝나면 연결을 해제하여야 한다. 그러나 비 연결성 방식 전송은 단지 데이터 전송만 존재하며 접속의 연결과 해체에 무관하다. 그러므로 전송하는 패킷은 수신단까지 전달되는 모든 정보를 저장하고 있고, 이 경우 도착한 패킷의 순서가 보낸 순서와 뒤 바뀔 수도 있고 전송도중 분실될 가능성도 있다.

이러한 CLNS를 서비스 사용자에게 제공하기 위하여 ISO 8473에서는 망 계층에서의 CLNP를 정의하고 있다[3]. 아래에서는 CLNP의 기능을 보다 구현적인 측면에서 기술하고 실제 구현된 소프트웨어의 구조에 대하여 간략히 기술하기로 한다.

1. CLNP의 기능

CLNP를 위한 Protocol Data Unit(PDU)의 형식은 크게 데이터 PDU와 오류 PDU와 나뉘며 그 구조는 헤더부와 내용부로 이루어진다[3]. 데이터 PDU의 경우, PDU는 헤더 부분인 프로토콜 제어 정보(Protocol Control Information, PCI)와 내용부인 데이터 영역으로 구성되어 있다. 제어 정보부는 다시 고정부분, 주소영역, 분할부, 선택사항부로 나뉘어진다.

상위 계층에서 내려오는 데이터의 길이(최대 64512 octet)는 하부망에서 전송할 수 있는 최대 데이터 길이보다 클 수 있으므로 Network Service Data Unit(NSDU)를 일정 크기로 분할하여 전송하여야 한다. 분할된 PDU를 유도 PDU라고 하는데, 이것이 표준안의 관점에서 만족되어야 할 조건은 다음과 같다. 각각의 유도 PDU들의 데이터 길이는 0이 아닌 8의 배수로 구성되어야 한다 (단 최후의 유도 PDU는 제외), 즉, 이처럼 사용자 데이터를 분할할때에는 7 octet 이하 크기를 가지는 유도 PDU로 구성하여서는 안된다. 그래서 분할부의 분할 offset 필드의 값은 0이거나 0이 아닌 8의 배수로 할당된다. 일반적으로, 주어진 채널

에서 한 비트에 에러가 발생할 확률이 고정되어 있다면, 어떤 데이터를 n 개의 데이터로 분할할 경우 패킷 헤더부의 오버헤드로 인해 성능의 저하가 발생한다는 사실이 알려져 있다. 그러나 성능의 저하를 가급적 줄이기 위해서는 n 개의 데이터의 길이를 등 간격으로 분할하는 것이 바람직하다[6]. 그러므로 유도 PDU의 제약을 만족하는 동시에 분할된 PDU의 길이를 동화시키는 최적화 알고리즘을 적용하여 분할 기능을 구현하여야 한다.

분할된 PDU를 받은 수신단에서는 재결합을 수행하며 이를 위해 timing 메커니즘을 포함한다. 다중 계층 통신 시스템을 구현하기 위해서는 통상적으로 server 모델과 activity-thread 모델을 사용하는데, 프로토콜 효율적 측면에서 전자는 하위 계층에 그리고 후자는 상위 계층에 적절한 모델이다[7]. 본 고에서는 CLNP 구현을 위하여 server 모델을 사용하였고 그 중에서도 single process server 모델을 이용하였기 때문에 여러개의 접속을 하나의 프로세스가 동시에 처리한다. 그래서 하부망으로 부터 유입되는 PDU들을 수신하는 버퍼에는 서로 다른 접속을 위한 PDU가 뒤섞여 있게 된다. CLNP는 각 접속에 대한 재결합이나 순서정렬을 위하여 각각의 PDU를 복사해 두거나 하부망으로 유입된 PDU의 포인터와 관련된 정보를 간직해 두는 방법을 써야 한다. 전자의 방법을 사용할 경우에는 처리상의 오버헤드가 심해 지극히 비효율적이다[8]. 그래서 본 고에서는 후자의 방법을 사용하였고 CLNP 내의 재결합을 위한 어떤 한 버퍼에는 하나의 접속에 대한 PDU의 포인터와 관련된 정보를 가지고 있다.

고정부의 영역은 위에서 설명한 것 외에도 존속시간 운영과 에러 발생시 오류 보고를 요구하는 비트 등이 있는데 구현 과정에서 모든 규약을 처리할수 있게 설계하였다. 주소 영역에는 목적지 주소와 발신지 주소를 기록하고 각 주소의 길이를 나타내는 필드가 주어지며 각 주소는 망 계층의 어드레싱 규약인 ISO 8348/Add.2에 준한다[9].

분할부는 고정부에 있는 분할 허용 flag가 '1'로 된 경우에만 존재하며, 초기 PDU가 적절히 조립되도록 유일하게 규정짓는 데이터 확인자, 유도 PDU의 데이터의 상대적인 위치를 기술하는 분할 offset, 그리고 초기 PDU의 헤더부분과 데이터를 포함하는 길이를 나타내는 PDU 전체 길이 필드로 구성되어 있다.

마지막으로 선택사항부에는 PDU 헤더길이를 편의대로 확장하게 하는 패딩 필드, 안전을 유지하기 위한

보안 필드, 서어비스 질을 제공하는 quality-of-service(QoS) 유지 필드, 패킷의 경로를 명시하는 발신지 경로 설정 필드, 중간 시스템이 주어진 패킷의 경로를 기록하는 경로 기록 필드 등이 있다.

경로 설정 과정의 분류에 있어서는, manual configuration에 의해 forwarding table이 얻어지는 정적 경로 설정 방식과 manual configuration과 더불어 진행 메커니즘에 의해 forwarding table이 얻어지는 준정적 경로 설정 방식이 있다[10]. 또한 중심 노드가 forwarding table을 계산하여 각 node에 table을 공급하는 집중 경로설정과, 각 node가 자체적으로 갱신, 결정하는 분산 경로 설정 방식이 있다. 본 고에서의 경로 설정 알고리즘은 manual configuration과 분산 방식에 기반을 두지만 경로 지정 표준안인 ISO 9542가 구현되면 쉽게 접속이 가능하다.

CLNP 프로토콜에 구현된 기능은 크게 송신부와 수신부의 기능으로 나누어질 수 있는데 송신부의 기능은 PDU 진행 기능, PDU 존속시간 제어 기능, PDU 경로 설정 기능, PDU 분할 기능, PDU 폐기 기능, 오류 보고 기능, 패딩 기능이 있고, 수신부의 기능은 PDU 분해 기능, 헤더 형식 분석 기능, 존속시간 제어 기능, PDU 경로 기록 기능, PDU 재결합 및 순서 정렬 기능, PDU 폐기 기능, 헤더 오류 검출 기능, 통화 혼잡 통보 기능이 있다.

2. CLNP의 Software 구조

CLNP의 동작은 세가지 상태의 finite state automation으로 모델링 되며 그 상태는 다음과 같다. 첫째로 automation이 맨처음 존재하는 초기상태, 둘째는 분할된 PDU를 결합하는 과정으로써의 재결합 상태, 셋째는 종결 상태이다[3]. 상태의 천이는 automatic event의 발생으로 야기되는데, 첫째로 망 서비스 경계 지역에서 발생하는 event로서 N_UNITDATA.Req와 N_UNITDATA.Ind이 있고 매개 변수로서는 발신지 주소(NS_SA), 목적지 주소(NS_DA), 발신지의 서비스 품질(NS_QoS), 그리고 발신지의 사용자 데이터(NS_Userdata)가 있다. 둘째, CLNP와 하부망 접속간에 발생하는 event로 SN_UNITDATA.Req와 SN_UNITDATA.Ind이 있고 매개 변수로서 SN_SA, SN_DA, SN_QoS, SN_Userdata가 있다. 마지막으로 CLNP 프로토콜과 지역 환경 간의 접속에서 발생하는 event로서 S_Timer.Req, S_Timer.Cancel, S_Timer.Response가 있으며 매개변수로는 PDU의 Lifetime, Timer의 이름, 그리고 가입자 이름이 있다.

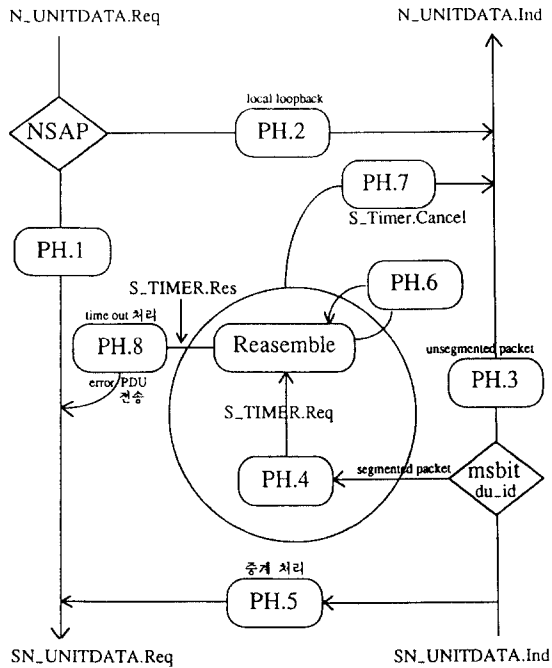


그림 2. CLNP software의 atomic event와 action의 분류.

그런데 하나의 atomic event가 주어진다 하더라도 여러가지 state로 천이할 가능성이 있으므로 그 event 자체만 갖고 천이가 불가능하며, 결국 그와 더불어 관련된 매개 변수 값의 조합으로 유일한 천이가 일어난다. 그래서 주어진 event와 관련 변수에 의해 어떤 행동이 취해지고 천이가 일어나며 그 행위는 지역 변수 값의 변화와 여러가지 기능의 연속적인 수행으로 구성된다. 그림 2에는 그 행동이 여덟가지 phase로 나타나 있고 그 행동을 약기하는 atomic event가 주어지며 부록에 각 phase에 관한 자세한 설명을 나타내었다.

III. CLNP 접속 기술

1. 개요

CLNP의 상위 계층과 하위 계층과의 접속은 CLNP 구현과는 다른 관점으로서 이는 CLNP가 운용되는 환경에 많은 영향을 받는다. 환경 운영 체제가 단일 프로세스이면 CLNP는 subroutine이나 function call처럼 사용하여 상위 계층과 하위 계층과 함께 하나의

프로세스로 이루어지게 된다. 그러나 본 연구에서 목표로 한 교육연구망의 기본 운영 체제는 UNIX 환경으로 다 계층별로 모듈화가 용이하며 계층화의 개념을 유지할 수 있다.

CLNP 및 하부 계층과 상위 계층들을 각각 프로세스로 구성하고 계층간의 primitive는 UNIX의 Interprocess Communication(IPC)을 사용하여 전달한다. IPC는 메시지 큐, 공동 메모리와 semaphore 방법이 있는데 본 고에서는 메시지 큐를 통해서 프로세스간에 데이터를 주고 받았으며 signal 기능도 일부 사용하였다. 시스템에서 제공하는 메시지 큐의 크기는 한정되어 있어서 프로세스간의 데이터 크기가 제한되는데 하부망과 CLNP 사이에서는 데이터 크기가 비교적 작아서 기본적으로 제공되는 메시지 큐를 사용할 수 있지만 상위 계층과 CLNP 사이의 데이터 크기는 ISO 8473에서 권고하는 64512 octet이므로 이를 메시지 큐를 통하여 지원하여 줄 수가 없다. 따라서 두 프로세스 사이의 메시지는 파일로 데이터를 넘겨주도록 한다. 즉, 한 프로세스에서 보내고자 하는 데이터를 파일로 만들고 나서 IPC를 통하여 파일 이름을 넘겨주며, 이를 받은 프로세스는 파일을 통하여 데이터를 받을 수가 있게 된다.

또한, CLNP는 다른 프로토콜을 사용하는 하부 망의 Internet Protocol(IP) 기능을 할 수 있도록 권고하는 프로토콜이다. 가장 많이 사용되는 ethernet과 X.25 망에 대한 IP로 CLNP를 사용하면 그림 1과 같은 형태가 될 것이다.

2. LAN(ethernet) 접속

CLNP 접속을 위하여서는 프로그래밍이 가능한 ethernet 보드를 선택하여야 하고 TCP/IP와 함께 있는 set에서는 링크 레벨의 프로그래밍이 최소한 가능해야 한다. 본 testbed에서 사용되는 Exclan 보드는 링크 레벨의 프로그래밍을 위하여 /etc/rc3.d/S80net의 파일의 netload 명령에 '-l'을 첨가하여야 하고 ethernet 주소를 바꾸기 위한 경우 '-e' 옵션을 통하여 가능하다. 또한 프로그래밍의 헤더 파일은 EXOS/src/include에 존재하는 파일들을 포함하여야 한다.

Ethernet 구동 프로그램은 반드시 슈퍼 사용자가 구동하여야 하고 이는 TCP/IP의 socket과 비슷한 형태로 프로그래밍이 되므로 socket(), send(), receive() 등을 이용하면 링크 레벨의 접근이 가능하다. 이때 send() 함수는 수신단의 수신여부를 확인할 수 없으므로 주의하여야 하고 receive()는 주소 필드를 영

로하여야 하며 메시지를 받는 string은 ethernet 프레임의 크기만큼 충분해야 한다.

3. WAN(X.25망) 접속

CLNP의 하부망으로 X.25망이 사용될 때는, Subnetwork Dependent Convergence Protocol(SNDCP)이 X.25의 연결성 방식 하부망 서비스를 Subnetwork Independent Convergence Protocol(SNICP)에서 요구하는 연결성 방식 서비스로 변환해서 제공해야 한다. 결국 SNDCP의 주된 임무는 X.25 접속 관리를 하면서 SN_UNITDATA primitive의 주소쌍과 그것에 해당하는 Virtual Channel(VC)의 파일 descriptor를 서로 대응시켜 주는 데 있다고 하겠다.

전송 계층과 SNICP 사이의 N_UNITDATA primitive는 SNICP와 SNDCP 사이의 SN_UNITDATA primitive로 바뀌어지는데, 이 때 SN_UNITDATA primitive의 주소들은 CCITT X.121 DTE 주소 형식을 취한다. ISO 8648 Internal Organization of Network Layer (IONL)에서의 Subnetwork Access Protocol(SNAcp)은 바로 X.25 PLP(ISO 8208)가 되며, 이 때 X.25 망은 단순히 CLNP간을 연결해 주는 하부망일 뿐이다. 즉 PLP의 경로선택 기능은 CLNP간의 X.25 패킷 경로선택에만 국한되며 CLNP의 경로선택 기능과는 무관하다. CLNP의 하부 서비스는 적어도 512 octet의 사용자 데이터 길이를 지원해야 하므로, 128 octet을 기본으로 하는 X.25에서는 SNDCP가 분할 및 재 결합하거나 PLP의 Packet Size Negotiation Facility를 이용해서 패킷 크기를 늘려야 한다.

ISO 8473은 X.25 망을 하부망으로 할 때의 Subnetwork Dependent Convergence Functions(SNDCF)에 대해서도 기술하고 있으며, 그 내용을 요약하면 다음과 같다.

(1) 호 설정 및 해제 기능

(2) Protocol Discrimination

— 호 요구 패킷의 사용자 데이터 필드 첫 octet은 ISO 8473 프로토콜 ID인 binary code 10000001로 채워야 한다.

(3) Resolution of Virtual Circuit Collisions

— 2개의 SNDCF가 동시에 서로 접속을 맺으려고 시도할 경우 DTE 주소가 큰 쪽에서 시작한 호만 설정한다.

(4) Virtual Circuit Timer (선택사항)

— 지정해 놓은 시간보다 오랜동안 채널이 idle할 경우 그 접속을 해제시킨다.

(5) Multiple Virtual Circuit (선택사항)

— 한쌍의 망 개체 간에 전송률을 향상시키기 위해서, 또는 채널 별로 다른 QoS의 traffic을 전송하기 위해서 다중 VC를 이용할 수 있다. 이 때는 추가 채널 설정을 위한 호 요구임을 나타내고 아울러 이들을 구별하기 위해 호 요구 패킷의 사용자 데이터 필드 4, 5 octet에 Subnetwork Connection Reference(SNCR) 값을 실어서 보낸다.

한편 비 연결성 방식 망 서비스상의 연결성 방식 전송 서비스에 관한 International Standardized Profile(ISP) TA1111/TA1121에 따르면, ISO 8473에서는 선택으로 기술되어 있는 다음의 기능들이 필수화 되어 있다.

- (1) aborting an outgoing VC attempt
- (2) rejecting an incoming VC
- (3) originating clearing of an established VC
- (4) response to indication of clearing
- (5) window rotation on updated P(R) values
- (6) sending & receiving M-packet, Q-packet

이 밖에도 ISO 8473에서는 PVC와 SVC중 하나를 선택할 수 있는데 반해 ISP에서는 SVC를 의무화 하고 있다. 이번 구현에서는 X.25 접속을 위해서 Adax card를 이용하였는데 그에 속한 소프트웨어를 이용하는 입장에서는 1, 2번 기능을 구현할 수가 없으며 따라서 이와 관련된 VC 충돌 해결 기능도 구현할 수 없게 된다.

그림 3에 SNDCP부 계층을 구성하고 있는 프로세스들과 그들간의 데이터 흐름을 도시하였다. 이처럼 많은 프로세스가 필요한 이유는 Adax 소프트웨어의 제약 때문이지만, 처리할 데이터가 없는 프로세스는 sleep 상태로 존재하므로 1개의 프로세스가 계속 polling 하면서 active 상태로 존재하는 것보다 light traffic에서는 오히려 효율적이다.

이들 각 프로세스의 역할은 다음과 같다.

- (1) sndcp : SNDCP부 계층의 중심 프로세스로서 접속 목록 관리를 함과 동시에 발신 호에 의한 채널로의 패킷 송신, 수신 호에 의한 접속 관련 패킷의 xlogin.alt로 전달기능을 갖는다. 오직 1개의 프로세스만이 존재하며 접속목록 관리를 위해서 xlogin.alt, xread 프로세스들과 signal, 메시지 큐를 이용해 정보를 공유한다.
- (2) xlogin.alt : Adax 소프트웨어는 수신 호당 1개씩의 xlogin.alt 프로세스를 생성시킨다. 따라서 수신 호

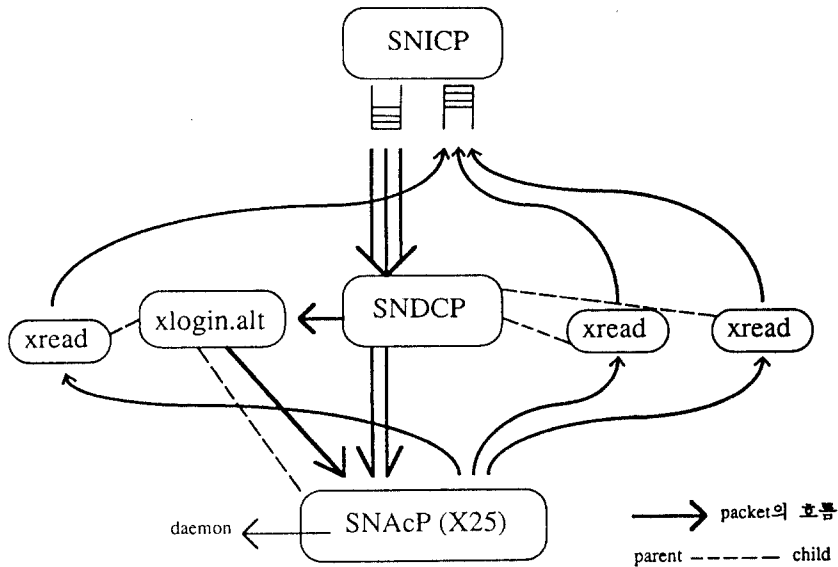


그림 3. SNDCP의 구성 process들과 Packet의 흐름

에 의한 접속 수 만큼의 프로세스가 존재하게 되며, 수신 호에 의한 채널로 패킷을 송신하는 임무를 갖는다. 생성시에는 접속 관련 정보를 접속 관리용 메세지 큐에 담고 snhcp 프로세스에게 signal을 띄움으로써 snhcp가 접속 목록을 관리할 수 있게 해준다.

- (3) xread : Adax 소프트웨어는 X.25 패킷 수신을 위한 system call을 제공하는데 이것을 이용하면 실제로 패킷을 수신할 때까지 멈춰있게 된다. 따라서 패킷 수신을 위해서는 각 접속 당 1개씩의 전담 프로세스를 두어야 함으로 결국 xread 프로세스는 설정된 접속 수만큼 존재하게 된다. 또한 접속이 끊어지면 xread 프로세스는 이 사실을 snhcp 프로세스에 알림으로써(역시 signal과 메세지 큐 이용) snhcp가 접속 목록 관리를 할 수 있게 해준다.

수신 호에 의한 채널로의 패킷 송신시에 xlogin.alt가 SNICP로부터 직접 데이터를 받지 않고 snhcp 프로세스를 거쳐 받는 이유는, SNICP와 SNDCP간의 단일화된 접속을 제공하기 위함이다. 본 연구를 수행하기 위한 X.25 망으로는 편의상 NP500이라는 Node Processor 1대가 이용되었다.

IV. 결 론

본 고에서는 서로 다른 하부망 간의 연동을 통한 OSI 표준화 환경 구축을 연구하여 교육 연구망의 OSI 표준화 서비스를 구현하는 목적을 가진다. 이를 위해 TCP/IP를 제공하는 ethernet LAN 환경과 OSI 표준화에 따른 X.25 공중망과의 연동 문제가 제기되며 이에 이면에 수행된 CLNP 구현과 하부망과의 접속 연구는 앞으로의 LAN/WAN 게이트 웨이 구현뿐 아니라 OSI 환경에서의 다양한 응용 서비스 개발에도 도움이 될 것이다.

본 연구는 표준 망 계층으로 CLNP가 구현되었고 X.25 규약 구현을 위해 Adax card가 사용되어 여기서 제공되는 system call과 CLNP 사이의 접속이 연구되었고 프로세스간 메시지의 교환을 위해 메세지 큐의 연구가 이루어졌으며 표준화된 OSI 응용 계층 서비스를 위해 ISODE 7.0에서 제공하는 전송 계층 서비스가 CLNP 위에 설치되었다.

이면에 연구된 각 하부망에서 수행되는 패킷 송·수신 절차는 특성이 다른 하부망이 여러개 연결된 경우, 접속 부분에서의 게이트 웨이 기능 및 CLNP 경로 설정에 관한 기능의 구현이 추가되어야 하며 이를 위한 방법 연구가 진행되어야 할 것이다. 한편 이러한 OSI

로의 이전은 Workstation 컴퓨터 환경에서 다중 채널 기능을 이용하는 방향으로 계속 진행되어 국내 교육 연구망에 OSI 표준 응용 서비스가 OSI 환경에서 설치, 운영되어야 할 것이다.

참 고 문 헌

1. H.J.J.H Schepers, et al., "LAN/WAN interworking in the OSI environment," *Computer Networks and ISDN Systems* V.23, pp.253-266, 1992.
2. OSI, "ISO 7498 : Information Processing Systems- Open System Interconnection -Basic Reference Model," 1984.
3. OSI, "ISO 8473 : Protocol for Providing the Connectionless-mode Network Service (Internetwork Protocol)," 1988.
4. M. T. Rose, "The ISO Development Environment: User's Manual, Volume 2: Underlying Services, Performance Systems International, Inc., 1991.
5. 은종관 외, "전송 계층 인터페이스 기술 개발에 관한 연구," 제 2차년도 최종보고서, 1991년 6월, 한국과학기술원.
6. 정해, 은종관, "X.25와 SDLC간 Protocol Converter의 성능 분석에 관한 연구," 한국통신학회 논문지 제6권 제9호, pp. 828-837, 1991년 9월.
7. L. Svobodava, "Implementing OSI Systems," *IEEE J. Select. Areas Commun.*, Vol. 7, No. 7, Sept. 1989.
8. D. D. Clark, D. L. Tennenhouse, "Architectural Considerations for a New Generation of Protocols," *Acm.*, 089791-405-8, pp. 200-208, 1990.
9. OSI, "ISO 8348/Add.2: Addendum 2: Network Layer Addressing," 1988.
10. M. Sylvest, "Routing Protocols for Interconnecting LANs," *Telecommunications*, pp. 95-98, Sept. 1990.
11. F. M. Berg, etc, "Networking of Networks: Interworking according to OSI," *IEEE J. Select. Areas Commun.*, vol. 7, No. 7, Sept., 1989.
12. OSI, "ISO/IEC 10608-5 : Information processing systems-Connection-mode Transport Service over Connectionless-mode Network Service, Part 5: TA 1111/ TA 1121 profiles including subnetwork-dependent requirements for X.25 packet-switched data networks using virtual calls"

부 록

(1) Phase 1

- 천이 방향 : 초기 상태 → 종결 상태.
- event : N_UNITDATA.Req.
- 매개 변수 : 목적지 주소가 지역 주소가 아님.
- 목적 : 지역 전송 개체가 상대 전송 개체에 데이터 송신.
- 행동 : 규약에 따라 패킷 헤더를 조립하여 PDU를 하부 계층으로 보냄.

(2) Phase 2

- 천이 방향 : 초기 상태 → 종결 상태.
- event : N_UNITDATA.Req.
- 매개 변수 : 목적지 주소의 값이 지역 주소와 일치.
- 목적 : 지역 전송 개체가 자신에게 데이터 전송 (loopback).
- 행동 : 패킷 헤더 조립없이 직접 N_UNITDATA.Ind를 함.

(3) Phase 3

- 천이 방향 : 초기 상태 → 종결 상태.
- event : SN_UNITDATA.Ind.
- 매개 변수 : 수신 PDU의 주소가 지역 주소와 일치, 분할 offset 값과 more segment 비트의 값이 '0'.
- 목적 : 지역 전송개체가 하부 망으로 부터 데이터 수신.
- 행동 : 패킷 헤더를 제거하여 상위 계층으로 데이터 보냄.

(4) Phase 4

- 천이 방향 : 초기 상태 → 재결합 상태.
- event : SN_UNITDATA.Ind.
- 매개 변수 : 수신 PDU의 목적지 주소가 지역 주소와 일치, 분할 offset 값이 '0'보다 크거나 more segment 비트 값이 '1'.
- 목적 : 하부망으로 부터 수신한 PDU의 재결합을 시작함.
- 행동 : 재결합을 위한 자원을 할당하고 S_Timer.Req event를 발생 시킴.

(5) Phase 5

- 천이 방향 : 초기 상태 → 종결 상태.
- event : SN_UNITDATA.Ind.

- 매개 변수: 하부망으로 부터 수신한 PDU의 목적지가 자신의 주소가 아님.
- 목적: 자신이 아닌 서로 다른 두 전송 개체 간의 데이터 송수신 과정에서 중계역할을 함.
- 행동: 존석시간을 갱신하여 하부망으로 PDU를 보냄.

(6) Phase 6

- 천이 방향: 재결합 상태 → 재결합 상태.
- event : SN_UNITDATA.Ind.
- 매개 변수: 수신한 PDU의 목적지 주소가 자신의 주소와 일치, PDU확인 번호가 일치, 발신자 주소가 일치.
- 목적: 분할된 PDU를 계속적으로 결합함.
- 행동: 분할 offset 필드의 값을 이용하여 데이터를 결합.

(7) Phase 7

- 천이 방향: 재결합 상태 → 종결 상태.
- event : SN_UNITDATA.Ind.
- 매개 변수: data_unit_complete.
- 목적: 분할된 데이터를 완전히 결합한 후 지역 전송개체에 보냄.
- 행동: S_Timer.cancel, N_UNITDATA.Ind.

(8) Phase 8

- 천이 방향: 재결합 상태 → 종결 상태.
- event : S_Timer.Response.
- 매개 변수: Timer 값 만료.
- 목적: 재결합 과정에서 시간이 만료된 상태방 전송 개체에 알림.
- 행동: 재결합 자원 해지, 오류 PDU 보냄.

※ 본 연구는 시스템 공학 연구소 교육 연구망 팀의 지원으로 이루어졌음.

김 병 철

- 1965년 10월 19일생
- 1988년 2월: 서울대학교 전자공학과 졸업
- 1990년 2월: 한국과학기술원 전기및 전자공학과 석사 졸업
- 1994년 5월: 한국과학기술원 전기및 전자공학과 박사과정 재학중

민 상 원

- 1965년 2월 15일생
- 1988년 2월: 광운대학교 전자통신공학과 졸업
- 1990년 2월: 한국과학기술원 전기및 전자공학과 석사 졸업
- 1994년 5월: 한국과학기술원 전기및 전자공학과 박사과정 재학중

정 해

- 1962년 1월 1일생
- 1987년 2월: 한양대학교 전자통신공학과 졸업
- 1991년 2월: 한국과학기술원 전기및 전자공학과 석사 졸업
- 1994년 5월: 한국과학기술원 전기및 전자공학과 박사과정 재학중

이 희 섭

- 1966년 4월 6일생
- 1989년 2월: 서울대학교 전자공학과 졸업
- 1991년 2월: 한국과학기술원 전기및 전자공학과 석사 졸업
- 1994년 5월: 한국과학기술원 전기및 전자공학과 박사과정 재학중

변 옥 환

- 1953년 8월 28일생
- 1979년 2월 : 한국항공대학교 통신공학과 졸업
- 1985년 2월 : 인하대학교 대학원 전자공학과 석사 졸업
- 1993년 8월 : 경희대학교 대학원 전자공학과 박사 졸업
- 1978년 9월 ~ 현재 : KIST 시스템 공학 연구소 교육 연구망 실장
- 1983년 12월 ~ 1984년 12월 : 미국 OMS Computer corp. 연구원
- 주관심분야 : Network Management, Network Security 등임.

은 종 관

- 한국과학기술원 전기 및 전자공학과 교수
- 제18권 1호 참조

신 병 철

- 1952년 11월 2일생
- 1975년 2월 : 서울대학교 전기공학과 졸업
- 1977년 2월 : 한국과학기술원 전기및 전자공학과 석사 졸업
- 1984년 8월 : 한국과학기술원 전기및 전자공학과 박사 졸업
- 1977년 2월 ~ 1980년 2월 : 한국전자통신연구소 연구원
- 1987년 3월 ~ 1988년 2월 : SRI International에서 Post-Doc. 연수
- 1984년 9월 ~ 현재 : 한국과학기술원 전기및 전자공학과 부교수
- 주관심분야 : Multimedia 통신, High Speed Network, Wireless