고성능 병렬 처리를 위한 수평적인 프로토콜 처리 구조

김 평 중[†] · 박 치 항[††]

요 약

분산 멀티미디어 응용은 다양한 전송 품질을 요구하지만, 가장 중요한 특성은 높은 성능과 적은 전송 지연이다. 고속 통신망의 전송 능력은 Gbps 수준을 제공하고 있지만 통신 구조상의 처리 부분 병목 현상 때문에 멀티미디어 응용까지 효율적으로 제공되지 못하고 있다. 이러한 문제점을 해결하기 위하여 통신 구조상 처리 부분을 수평 구조화, 병렬화 함으로써 고속화 하였다. 불행하게도 OSI 프로토콜 스택은 계층 단위의 병렬화를 가로막는 순서상 제약 조건을 갖고 있다. 우리 모델은 고정 패킷을 사용함으로써 망 계층에서 부터 표현 계층까지 동시에 수행시킬 수 있었다. 프로토타입 구현에 의하면, 우리 모델은 기존의 OSI 계층 구조 모델에 비해 약 61%의 성능 향상을 보여 주었다.

Horizontal Protocol Processing Architecture for High Performance Parallel Processing

Pyeong Jung Kim[†] · Chee Hang Park[††]

ABSTRACT

In the distributed multimedia application, high throughput and low delay is one of the most important QoS (Quality of Service) requirement. Emerging high speed communication network offers transmission rate above Gbps, but it can not be utilized efficiently by the performance bottelnecks of communication protocols. To overcome the problem, we propose a horizontal processing architcture that processes data as soon as it arrive from the network. Unfortunately, the OSI protocol stack often imposes ordering constraints that prevent concurrent processing of the protocol layers. By using a fixed packet format, the network layer through the presentation layer are processed in parallel. Our prototype shows that the proposed model has performance improvement upto 61% more advantage than the conventional approach.

1. 서 론

광통신 기술의 발전으로 수백 Mbps 부터 수 Gbps의 전송 능력을 갖는 고속 통신망이 등장하고 있다. 예를 들어 FDDI(Fiber Distributed Data Interface)는 100 Mbps에서 동작하고, ATM(Asynchronous Transfer Mode)은 수 Gbps의 전송 능력을 보유하고 있다. 지난 10년 동안 통신망의 전송 능력은 수 Kbps에서 수백 Mbps로 10^5배 증가한 반면 상용 프로세서의 처리 속도는 단지 10^3배 증가 되었다. 이 처럼 고속 통신망의 전송 속도 증가율이 프로토콜을 처리 하는 상용 프로세서의 처리 속도 증가율 보다 상대적으로 훨

† 정 회 원:한국전자통신연구소 멀티미디어연구부 선임연구원
†† 정 회 원:한국전자통신연구소 컴퓨터연구단 단장
논문접수:1996년 4월 16일, 심사완료:1996년 8월 2일

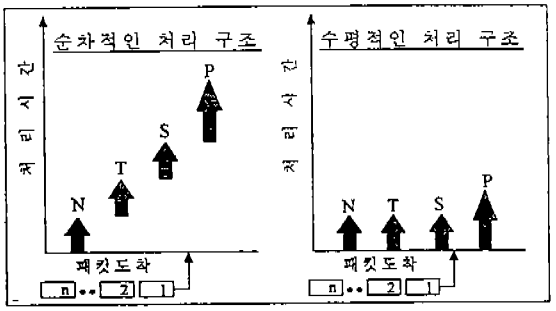
인 크기 때문에 프로토콜 처리에 병목 현상이 발생한다[1].

고속 통신망의 등장은 필연적으로 분산 멀티미디어 응용을 촉진시키고 있다. 분산 멀티미디어 응용 서비스는 다양한 종류의 미디어(음성, 화상, 그래픽, 텍스트)와 다양한 접근 방식(대화식, 실시간, 대량 데이타)의 카테이션 곱 만큼 존재할 수 있고, 다양한 요구 사항(성능, 지연, 지터(jitter), 기타 응용 서비스 종속적인 사항들)을 갖는다[2, 3]. 특히 영상회의 시스템처럼 대화식이고 실시간 접근 방식으로 동영상이나 음성 데이타를 동시에 요구하는 응용 서비스는 적은 지연 시간과 넓은 통신 대역폭을 요구하는 문제점이 발생한다.

이러한 문제점들은 통신 프로토콜 계층 구조상 허리 부분에 해당되는 계층들을 고속화 함으로써 해결할 수 있다. 연구 방향은 기존 프로토콜의 약점을 보완하여 새로운 프로토콜을 설계함으로써 프로토콜 자체의 성능을 높이거나[4, 5], 또는 기존의 프로토콜을 고속으로 처리하는 방법 등으로 진행되고 있다[6, 7, 8]. 후자의 경우는 새로운 프로토콜을 설계하는 것보다 좋은 구현 환경(효율적인 프로세스 스케쥴링, 타이머 메카니즘, 버퍼 관리 등)에서 구현하는 것이 [6, 7, 8] 중요하다고 보는 경우이다. 이러한 방법에는 VLSI(Very Large Scale Integration)를 이용하거나 병렬 처리를 하는 경우가 있다.

본 논문에서는 기존 프로토콜을 고속화하는 방향으로 선택하였고, 각 프로토콜 계층들을 수평 구조(horizontal architecture)에 입각하여 병렬로 처리함으로써 허리 부분을 고속화 하였다. 이러한 방향을 선택한 이유는 다음과 같다. 첫째, 종단 시스템에서 프로토콜의 처리 속도를 증가시키는 측면에서 볼 때 순수한 프로토콜 상태 머신의 처리 비용이 전체 프로토콜 처리 비용의 약 20%에서 30% 정도밖에[9] 되지않기 때문에 새로운 프로토콜 설계 보다는 기존 프로토콜의 구현 환경에 중점을 두었다. 둘째, 통신 구조 상에 병렬성을 가능한 많이 발견하여 최대의 병렬 처리를 [7, 8, 10] 수행 함으로써 처리 능력을 향상시킬 수 있고, 세째, 망으로 부터 패킷을 받자마자 가능한 빨리 허리 부분 전체를 동시에 수행시킬수 있기 때문이다.

(그림 1)에서 순차적인 처리 구조는 망으로부터 패킷이 도착 하였을 경우, 계층 구조 모델의 개념에 따라 각 계층을 순차적으로 처리했을 때 전체적인 수행 시간을 나타내고, 수평적인 처리 구조는 각 계층의 프로토콜을 수평적인 처리 구조 모델에서 병렬 처리했을 때 전체적인 수행 시간을 나타낸다. 여기에서 N은 망 프로토콜을, T는 수송 프로토콜을, S는 세션 프로토콜을, P는 표현 프로토콜을 표시한다.



(그림 1) 순차적인 처리 구조와 수평적인 처리 구조
(Fig. 1) Sequential and horizontal protocol processing structure

본 논문의 구성은 다음과 같다. 2장에서는 분산 멀티미디어 응용 서비스 요구 사항을 추출하고, 3장에서는 다른 프로토콜 처리 시스템의 구조를 비교 분석한다. 4장에서는 통신 구조상의 허리 부분을 고속화하는 수평적인 프로토콜 처리 구조 모델을 제시하고, 프로토타입핑(prototyping)에 대하여 기술한다. 5장에서는 제안 모델에 대한 분석을 하고, 6장에서 결론을 내린다.

2. 분산 멀티미디어 응용의 요구 사항

멀티미디어는 다양한 종류의 미디어들이 단순히 혼재되어 있는 것이 아니라 디지탈 형태로 저장, 검색, 전송 그리고 표현되며 하나의 통합된 환경에서 동시에 다루어진다. 분산 멀티미디어 응용은 멀티미디어 데이터 트래픽 특성을 갖는 망 응용(network application)으로 정의하면, 다양한 종류의 미디어(텍스트, 그래픽, 음성, 이미지 및 동화상 등)와 다양한 접근 방식(대화식, 실시간, 대량 데이터 등)의 카테이션 곱 만큼 존재하고[2], 응용 분야에 따라 무한히 많이 존재한다.

〈표 1〉 트래픽 유형에 따른 전송 품질
〈Table 1〉 Comparison of quality of service with traffic streams

전송품질 / 트래픽 유형	최대 전송지연 (s)	최대 지연지터 (ms)	평균 성능 (Mbps)	허용할 수 있는 비트 오류율	허용할 수 있는 패킷 오류율
음성	0.25	10	0.064	< 10	< 10^{-1}
영상 (TV수준)	0.25	10	100	10^{-2}	10^{-4}
압축된 영상	0.25	1	2~10	10^{-4}	10^{-7}
데이터 (화일전송)	1	-	2~100	0	0
실시간 데이터	0.001~1	-	< 10	0	0
이미지	1	-	2~10	10^{-4}	10^{-7}

분산 멀티미디어 응용의 대표적인 시스템으로서 ETRI 멀티미디어 연구부에서 개발한 영상회의 시스템이 있다[11]. 이 시스템은 영상회의, 영상메시지, 그래픽 공동편집기 등의 서비스를 제공하고 현재 정부기관에 설치 운용되고 있다. 영상회의 시스템은 참가자들간 서로 얼굴을 보면서 대화할 수 있고, 그래픽 공동편집기를 사용하여 필요한 자료를 공유할 수 있다. 이 때 전송되는 트래픽 유형은 동영상, 음성, 그래픽 데이터, 이미지, 그리고 마우스로 지적한 정보(pointing information) 등이 전달된다. 영상 회의인 경우 연속성 있는(continuous) 대량의 동영상과 음성 데이터를 다수의 참가자 사이에 전송해야 한다. 3자 영상회의는 320*240*8bit(256 색)*15(15 프레임/초)*6(3명*2)이므로 약 55Mbps의 영상 데이터와 11.025KHz*8bit*6(3명*2)이므로 약 0.5Mbps의 음성 데이터를 요구한다. 우리는 20자 이상의 영상 회의를 할 경우 이진 트리 형태를 사용한다. 근 노드로서 의장과 발언자가 있고, 영상 회의 중에는 2개의 이진 트리를 갖고 있다. 중간 노드는 자식 노드에게 동영상과 음성 데이터를 전달해준다. 의장은 회의를 호출한 사람이므로 고정되지만, 발언자는 동적으로 변하기 때문에 그 때마다 발언자 트리를 새로 구성한다. 실제 FDDI 망에서 운영해본 결과, 근 노드에서 부터 리프 노드까지의 지연 시간은 2~5초 정도 걸리고, 지터 문제는 멀티미디어 입출력 서버인 MUX에서 큐를 사용하여 해결한다. 이 때 가장 중요했던 특성은 망 대역폭의 고속화와 각 노드의 고속 처리 능력, 그리고 적은 지연 시간이었다.

〈표 1〉에서 음성이나 영상은 실시간 특성이 중요한 반면 비트 오류율은 상대적으로 높고, 이미지나 데이터 파일은 실시간 특성이 적은 반면 비트 오류율이 적어야 함을 보여주고 있다. 데이터 율(data rate) 측면에서 동영상은 음성 보다 많고, 전송 지연(delay) 측면에서 동영상이나 음성이 이미지나 마우스 정보 보다 엄격하다. 이와 같이 멀티미디어 응용에서 서로 다른 트래픽 유형에 따라 〈표 1〉에서와 같이 다양한 전송 품질(quality of service)을 요구한다[3]. 이밖에 다양한 성능(throughput), 다양한 패킷 크기, 신뢰성 있는 전송이나 best-effort delivery(비트 오류, 흐름 제어), 다자간 연결(multipoint connection), 채널의 동기화, 지터(jitter) 등을 요구하고 있다. 그러나, 이러한 전송 품질은 분산 멀티미디어 응용에 따라 천차만별로 다양하기 때문에 모든 요구 사항을 동시에 만족시키기 어렵다. 따라서, Gbps 수준의 망 대역폭을 분산 멀티미디어 응용 서비스에게 제공하기 위하여 통신 구조상 허리 부분을 높은 성능(high throughput)과 적은 전송 지연(low delay)갖도록 제공하는 것이 가장 중요하다.

3. 프로토콜 처리 구조의 비교

병렬 처리 고속화 모델의 대표적인 것은 HOPS(Horizontal Oriented Protocol Structure) 모델[1], ALF(Application Level Framing) 모델[2], 그리고 Choi의 모델

〈표 2〉 병렬 처리 모델의 비교
〈Table 2〉 Comparison of parallel processing model

모델명 / 구 분	Application Level Framing (ALF)	Horizontal Oriented Protocol Structure (HOPS)	Choi's Model
관련계층	4,5,6,7	4,5,6	3,4,5,6
처리구조	응용중심 처리구조	수평 구조	수평 구조
패 킷	ADU	HOPS 패킷	제한된 OSI 패킷
처리모델	MISD	MISD	MISD+Pipeline
처리수준	계층단위+ 패킷단위	기능단위	접속단위+계층 단위
계층개념	있음	없음	있음
망으로부터 표현 수행	가능	가능	가능
특기사항	o ILP o 응용 중심의 프로토콜 설계	수평구조에 의한 병렬화 개념	수평구조에서 계층간 병렬화 처리 구조

[10] 등을 들수 있고, 〈표 2〉에서와 같이 분석하였다.

Z. Haas는 통신 구조의 계층 모델을 수평 구조화한 HOPS 모델을 제안하였다[1]. 이 모델의 장점은 계층 모델에서 계층간의 투명성을 보장하기 위하여 지불되는 비용(예로서 계층간의 통신 부담)을 줄이고, 병렬로 처리 함으써 성능을 향상시킨다. 첫째, 프로토콜 기능들의 수평 구조이기 때문에 하나의 패킷을 처리하는 시간이 프로토콜 기능에 대한 처리 시간의 합이 아니라 가장 늦은 기능의 처리 시간 만큼 걸리며, 둘째, 프로세서를 더 많이 사용함으로써 어느 한계 까지 쉽게 성능을 증가시킬 수 있다. 세째, 수직 구조의 계층 모델에서 발생하는 중복된 프로토콜 기능들을 한번만 수행하게 하고, 버퍼링 같은 계층 간의 통신 부담을 감소시킨다. 네째, 수평 구조 자체가 자연스럽게 병렬 처리 구현 구조로 이끈다. 그러나, 프로토콜 기능 단위를 병렬로 처리하기 때문에 병렬 단위(granuality)가 너무 작아 병렬 처리 오버헤드(예로서 병렬 처리 프로세서 간에 동기화와 통신 부담)가 증가될 수 있다. 또한, 표현 번역 시간은 전체 프로토콜 스택의 90% 이상을 차지하고 있는데도 불구하고 다른 프로토콜 기능과 같은 단위로 병렬 처리 하기 때문에 전체적인 성능 향상에 문제가 있다고 생각한다.

D. Clark는 프로토콜 계층 구조가 응용 서비스에 종속되도록 패킷 구성과 데이타 전송을 수행하는 ALF 모델을 제안하고, 구현 기법으로서 ILP(Integrated Layer Processing)를 제안하였다[2]. 이 모델의 장점은 계층간의 투명성을 그대로 살리면서 프로그램 구현자에게 다양한 구현 방안을 제공할 수 있다는 점이다. 즉, 프로그램 구현자가 OSI 참조 모델을 병렬처리하도록 구현할 수 있도록 한다. 그러나, 각 응용 서비스가 ADU를 구성하기 때문에 통신하고자 하는 기존의 모든 응용 서비스가 변경되어야 한다. 또한 하나의 모델에 대하여 다양한 구현 방식이 존재한다는 점에서 프로토콜 설계와 프로토콜 소프트웨어의 유지 보수, 재사용성 등을 복잡하게 하거나 어렵게하는 문제점이 있다.

S. Choi는 OSI 프로토콜 스택에서 표현 연결 마다 표현 문맥(presentation context)이 다르기 때문에 각 연결 단위로 병렬화를 수행한 모델이다[10]. 그리고 망 패킷으로 부터 표현 데이타를 추출하여 부분적으로 표현 번역을 시작함으로써[7, 10] 성능을 향상시켰다. Choi 모델의 장점은 OSI 계층 구조 개념을 수정하지 않고 병렬 처리를 한 점이다. OSI 패킷은 가변 구조 패킷이고, 분할 및 재조립 기능이 망 계층, 수송 계층, 세션 계층에서 발생한다. 망으로 부터 데이타가 도착함과 동시에 표현 번역 기능을 수행시키기 위해서는 망 패킷을 순서적으로 접근하여야 한다. 망 패킷에는 항상 수송 헤더와 세션 헤더가 존재하지 않기 때문에 표현 연결 정보나 헤더 등을 결정하는데 복잡한 메카니즘(연결 정보를 찾기위한 테이블 매핑이나 재조립 과정 등)이 필요하다. 이러한 문제점을 해결함으로써 56% 성능을 향상시켰다. 그러나, 상하위 계층간 순서상 제약 조건을 제거할 수 있다면 계층간 병렬화를 성취할 수 있고, 간단한 메카니즘으로도 구현이 가능할 것이다. 또한, 표현 연결 단위로 병렬 처리를 하기 때문에 연결 단위의 전송 데이타 크기에 따라 성능이 좌우되는 문제점이 있다.

4. 수평적인 프로토콜 처리 구조

분산 멀티미디어 응용에게 높은 성능과 적은 전송 지연을 제공하기 위하여 프로토콜 구조상 허리 부분을 수평 구조화한 모델을 제안한다. 본 논문은 데이타 전송 단계를 중심으로 프로토타입핑 하였고, OSI 프로토콜의 계층 3(망 계층)은 비연결형 프로토콜 타입 1, 계층 4(수송 계층)는 오류 복구 및 멀티플렉싱을 제공하는 클래스 4, 계층 5(세션 계층)와 계층 6(표현 계층)은 커널 기능 단위 등을 포함한다.

4.1 OSI 프로토콜의 수평 구조화

OSI 참조 모델에서 계층 구조의 개념은 계층간의 투명성을 보장하기 때문에 이해나 구현 측면에서 장점을 갖고 있지만, 이 개념의 엄격한 적용은 고속 통신 구현 구조에 부적합하다[2, 5, 12]. 여러 계층에서의 프로토콜 기능 중복, 불필요한 기능 수행, 제어 메시지 오버헤드 그리고 프로토콜 처리의 병렬성 제약 등 때문에 성능을 감소시키고 통신 지연을 증가시킨다. 이러한 문제점을 해결하면서 동시에 가능한 빨리 표현 번역 기능을 수행시키는 모델을 찾기 위하여 우선 계층 구조 모델을 살펴보고, 계층 단위로 수평 구조화하는데 따르는 제약 조건을 분석한다.

OSI 계층 구조 모델에서 (N)-계층에서 (N+1)-계

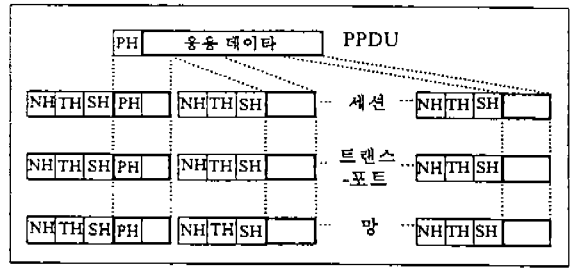
층으로 데이타를 전송하기 위해서는 (N)-계층의 기능이 모두 완료된 후 전송되는 수직 구조(vertical architecture)를 갖는다. 계층간의 정보전달은 상위 계층으로부터 받은 PDU(Protocol Data Unit)에 자신의 PCI (Protocol Control Information)를 첨부하는 계층간 캡슐화(hierarchical encapsulation)를 발생시킨다. 이것은 상하위 인터페이스만 알고도 하나의 프로토콜을 이해하거나 구현할 수 있는 장점을 갖지만, 고속처리를 제약하게 되며[2, 12, 13, 14], 다음과 같이 요약 된다.

- (N)-계층의 프로토콜 기능이 완료되어야 상하위 인접 계층으로 데이타를 전달하므로 순차적인 처리만을 강요한다.
- (N)-PCI를 구성하고 있는 각 필드는 (N)-계층의 프로토콜 기능과 일대일 대응 관계를 갖게 되므로 항상 특정 순서에 따라 처리 되어야만 한다.
- 항상 인터페이스를 통과하므로 계층간 통신 과부하가 발생한다.
- 여러 계층에서 중복된 기능(예, 체크섬)을 수행한다.

계층 구조 모델은 수직 구조와 계층간 캡슐화에 따른 순서상의 제약 조건을 발생시킨다[7, 10, 13]. OSI 패킷은 가변 길이 패킷(variable packet)이고, 망 계층, 수송 계층, 그리고 세션 계층에서 분할 및 재조립(segmentation and reassembly)이 발생한다. 이것은 프로토콜 계층 단위로 수평 구조화 하는데 따른 제약 조건이 되고, 다음과 같이 정리될 수 있다[7, 10, 13].

- 분할 및 재조립 제약 조건:(N)-PDU의 크기가 (N−1)-SDU(Service Data Unit)의 크기보다 크면 여러 개의 (N)-PDU로 분할되어 상대방에게 전송되고, (N)-계층 프로토콜은 이들을 재결합하여 상위 계층으로 전달한다. 그 결과 분할된 모든 세그먼트들을 받을 때까지 상위 계층에 전달하지 못함으로써 계층간 순서상 제약 조건이 발생한다.
- PDU 필드의 가변성 제약 조건:각 프로토콜에서 처리하는 PDU의 각 필드는 대부분이 순서, 위치, 크기가 고정되어 있지않기 때문에 항상 패킷의 처음부터 해석, 처리하게 된다.

본 논문은 프로토콜 계층 단위로 수평 구조화 하는데 따른 제약 조건을 없애기 위하여 고정 패킷 형태

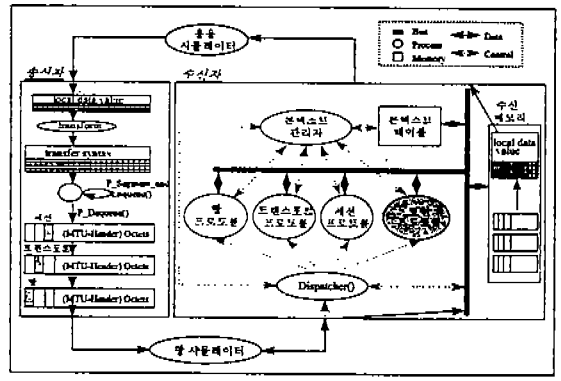


(그림 2) 고정 패킷 형태
(Fig. 2) Fixed packet form

를 채택한다. (그림 2)와 같은 고정 패킷을 사용하기 때문에 망으로 부터 패킷을 받자마자 프로토콜 계층 단위로 병렬화하는데 용이하다.

4.2 제안 모델

본 논문에서는 (그림 3)과 같이 수평 구조화 모델을 제안한다. 이 모델의 송신측은 순서적인 처리 구조로서, 사용자로부터 데이타를 받으면 ASN.1/BER을 사용하여 전송 구문을 생성한다. 분할 및 재조립 기능이 발생되지 않도록 하기 위하여 고정 패킷을 구성할 때 하부 망의 MTU 값을 고려한다. 이를 위하여 표현 연결 설정시에 MTU 값을 알 수 있음을 가정하였다. 고정 패킷 구성시 세션 헤더, 수송 헤더, 그리고 망 프로토콜 헤더 등의 수용할 공간을 미리 생성한다. 생성된 패킷은 큐에 저장하고 필요할 때 하나씩 전송하게 된다. 송신측은 세션 프로토콜, 수송 프로토콜, 그리고 망 프로토콜 등의 순서로 처리되어 전송한다.



(그림 3) 제안 모델
(Fig. 3) Proposed model

수신측은 망으로부터 패킷을 받자마자 병렬로 각 프로토콜들을 수행시킨다. 망 시뮬레이터는 Dispatcher를 부르고, Dispatcher는 그 패킷을 분석하여 각 프로토콜 프로세스에게 헤더와 데이터를 넘겨준 후 병렬로 시작시킨다. 콘텍스트 관리자는 병렬로 수행하는 프로세스들의 동기화 및 통신하는 양측간에 표현/세션/수송 연결 관리를 수행한다.

4.2.1 고정 패킷의 구조

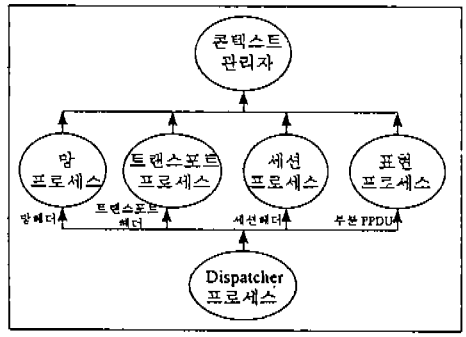
(그림 4)는 데이타 전송 단계의 고정 패킷 구조를 나타낸다. 망 헤더인 N-PCI는 고정 부분(fixed part)과 주소 부분(address part)으로 구성되어 총 31 옥텟으로 구성한다. 수송 헤더인 T-PCI는 정규 형태로 구성하였고, 옵션 형태인 체크섬 필드는 고정 패킷의 끝부분에 포함되었다. 세션 헤더인 S-PCI는 기본 조합(basic concatenation)을 포함하여 구성하고, 표현 헤더인 P-PCI는 simply-encoded-data를 선택하였고, 특히 송신측에서 고정 패킷 구조를 만들기 위하여 임의로 분할하게 된다. 이 때 분할된 각각이 순서적으로 정확히 전송 되었는지를 검사하기 위하여 두개의 필드(P_Seqno, P_EOT)를 포함시켜 구성하였다.

Network Layer Protocol Identifier(NLPI) Length Indicator(LI) Version Lifetime SP MS E/R Type Segment length (2) Chksum (2)	고정 부분 (9)	N-PCI (31)
Destination address length Destination address (10) Source address length Source address (10)	주소 부분 (22)	
DT DST_REF (2) TPDU_NR and EOT		T-PCI (4)
SI LI SI LI		S-PCI (4)
P_Seqno (2) P_EOT		P-PCI(13) (Simply-encoded-data)
[Application0] IMPLICIT Simply-encoded-data length		
Octet string		표현 데이타
Checksum_code(2)		

(그림 4) 고정 패킷 구조
(Fig. 4) Fixed packet structure

4.2.2 수행 방법

우리 모델은 MISD(Multiple Instruction Single Data) 형태이다. 망 시뮬레이터로부터 패킷을 받자마자 Dispatcher는 수신 메모리에 저장시키고 다음과 같은 사항을 결정한다. 첫째, 목적지 망 주소(destination network address)를 사용하여 최종 목적지가 자기 시스템인지 아니면 다른 시스템으로 전송해야(forward) 하는지를 결정한다. 둘째, 패킷이 최종 목적지이면, ⟨Source address, DST_REF⟩를 사용하여 특정 연결 콘텍스트를 찾는다. 수송 연결은 세션 연결 및 표현 연결과 일대일 대응 관계를 갖고 있다.



(그림 5) 수행 방법
(Fig. 5) Processing method

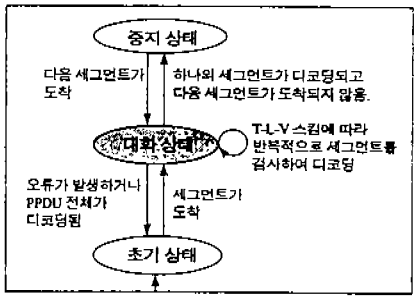
Dispatcher는 (그림 5)와 같이 각 프로토콜 프로세스에게 자기 헤더와 연결 콘텍스트를 전달하고, 병렬로 프로세스들을 시작시킨다. 각 프로토콜은 Dispatcher로부터 연결 콘텍스트와 패킷을 전달 받으면, 패킷 내의 자기 헤더를 사용하여 프로토콜 기능을 수행한다. 망 프로토콜은 수명 관리 기능, 헤더 포맷 분석 기능, 경로 설정 기능 등을 수행하고, 오류가 발생하면 콘텍스트 관리자에게 오류 보고를 한다. 수송 프로토콜은 헤더 분석 기능, 흐름 제어 기능 등을 수행하고, 세션 프로토콜은 헤더 분석 기능과 기본 조합 기능 등을 수행한다. 표현 프로토콜은 표현 문맥(presentation context)의 설정과 관리를 수행하고, 표현 번역을 수행한다. 표현 문맥은 (추상 구문, 전송 구문)의 쌍으로 정의되고, 표현 연결 설정시에 정의된다. 특히, 표현 데이타가 분할된 경우 순서적으로 중복없이 도착되었는지를 결정해야 한다. 이를 위하여 P_Seqno와 P_EOT 필드를 사용하였다.

콘텍스트 관리자는 병렬로 수행되는 각 프로세스

들의 동기화와 통신하는 응용 시스템 간의 연결 관리를 담당한다. 동기화는 병렬로 수행되는 각 프로세스들간의 시작과 종료를 같게하는 것으로서 가장 늦은 프로세스 시간에 의존한다. 우리는 계층간의 병렬화를 추구하고 있기 때문에 각 계층 프로토콜에서 수행된 결과를 판단하고, 관련된 콘텍스트 정보를 관리하여야 한다. 콘텍스트 관리자는 통신하는 양측의 수송 연결, 세션 연결, 그리고 표현 연결 등에 대하여 연결 설정, 해제, 및 관리를 수행하고, 각 프로세스의 프로토콜 상태를 관리한다. 만약 특정 프로세스에서 오류가 발생될 경우 사건(event)을 발생시킴으로써 콘텍스트 관리자에게 보고되고, 그에 따른 적절한 처리를 수행한다.

4.2.3 표현 번역

표현 번역 기능은 ASN.1/BER 부분 디코딩[7, 10, 13]을 사용한다. 망으로부터 받은 데이타는 표현 데이타(PPDU)의 전부를 포함하지 않을 수 있다. 왜냐하면 송신측은 고정 패킷 구조를 만들기 위하여 엔코딩된 데이타(transfer syntax)를 구문과는 전혀 관계없이 MTU 단위로 분할하여 보내기 때문이다. 수신측은 T-L-V 기법으로 구성된 엔코딩된 데이타의 일부만을 갖게 된다. 다시 말하면, Type 필드만 있는 경우, Length 필드가 짤린 경우, 그리고 Value 필드가 모자라는 경우 등이 존재한다. 이러한 경우에 ASN.1 디코딩을 수행하지 않고 다음 패킷이 도착하면 계속 수행하도록 한다[7, 10, 13].



(그림 6) ASN.1/BER 부분 디코딩을 위한 상태도
(Fig. 6) State diagram for ASN.1/BER partial decoding

(그림 6)은 부분 데이타를 처리하기 위한 상태도이다. 초기 상태(initial state)는 표현 계층이 첫번째 세그먼트를 기다리는 상태이고, 세그먼트가 도착되면 대화 상태(conversion state)로 천이 된다. 대화 상태에서 ASN.1/BER 디코더는 T-L-V 스킴을 반복적으로 검사하여 디코딩 한다. 오류가 발생하거나 PPDU 전체가 디코딩 되었을 경우 초기 상태로 천이되고, 다음 세그먼트가 아직 도착되지 않았을 경우 중지 상태(suspending state)로 천이된다.

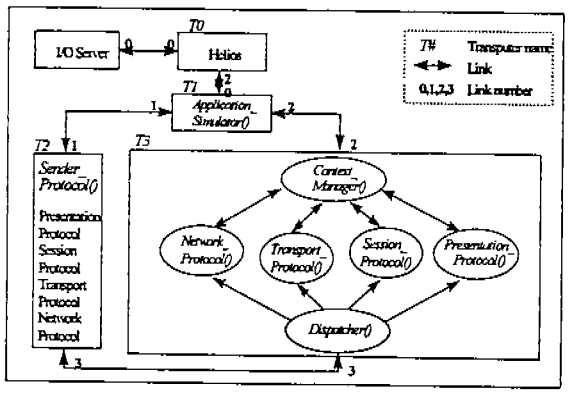
4.3 프로토타입 구현

4.3.1 구현 환경

구현 환경은 Parsytec Computer사의 Transputer를 사용하였다[16]. 구현 언어는 병렬 프로그램 언어인 Par.C[17]을 사용하였고, 구현 시스템은 MultiCluster-2(이후에는 MC-2로 부름) 시스템이다. MC-2는 1개의 NCU(Network Configuration Unit)와 16개의 Transputer로 구성되어 있다. 호스트 시스템은 PC를 사용하였고, PC에는 Helios 병렬 운영 체계[18]가 운영되고 있다.

각 프로세서는 Transputer 망을 구성하기 위하여 resource map을 작성하여 helios에서 컴파일 하면 자동적으로 Transputer 망을 구성한다. 각 프로세서 타입으로 helios와 native 타입이 있는데 우리는 native 타입으로서 구현하였다.

Par.C System은 C 언어에 병렬 처리를 위하여 몇가

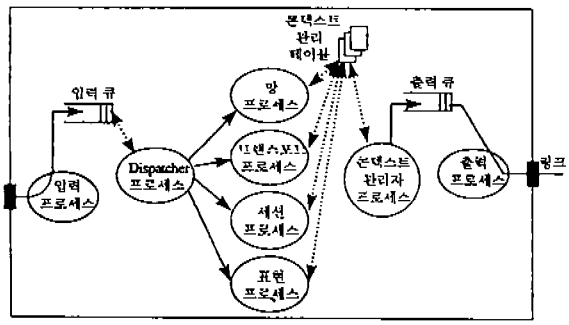


(그림 7) 구현 구조
(Fig. 7) Implementation Structure

지 병렬 기능을 추가하였다. "PAR"는 병렬 프로세스를 시작하고(invocation), "CHANNEL"은 병렬 프로세스 사이에 통신 수단을 제공하고, "SEND/RECEIVE"는 Transputer 사이의 통신을 제공하고, "SELECT"는 정의된 여러 사건 중에서 첫번째 사건을 기다린다. 병렬 프로세스 사이에 통신은 동기화되고 버퍼없는 rendezvous 메카니즘을 이용한다.

4.3.2 구현 구조

(그림 7)은 3개의 Transputer를 이용한 구현 구조를 보여준다. 여기서 Tn은 Transputer 망에서 노드 번호를 의미한다. I/O Server는 helios 운영 체계에서 동작되는 프로그램으로서 Transputer 망을 부팅하거나 재시작 시키고, 모든 Transputer와의 입출력을 수행한다. I/O Server와 연결되어 있는 T0를 Root Transputer라 부르고, T0는 Helios 모드로 동작되고 나머지 Transputer는 Native 모드로 동작된다. T1는 응용 시뮬레이터로서 멀티미디어 응용 데이터를 생성하여 T2에게 보내는 프로세스와 T3로 부터 데이터를 받는 프로세스로 구성된다. T2는 송신측 프로토콜을 동작시키는데, 표현 프로토콜, 세션 프로토콜, 수송 프로토콜, 그리고 망 프로토콜 등을 순서적으로 수행한 후 T3로 전송하는 프로세스로 구성된다. T3는 수신측 프로토콜을 동작시키는데, Transputer가 공유 메모리를 갖고 있지않고, Transputer의 process creation time과 context switching time은 1 microsec. 보다 작기[16] 때문에 하나의 Transputer(T3)에서 모든 프로토콜 프로세스들을 병렬로 수행시켰다.

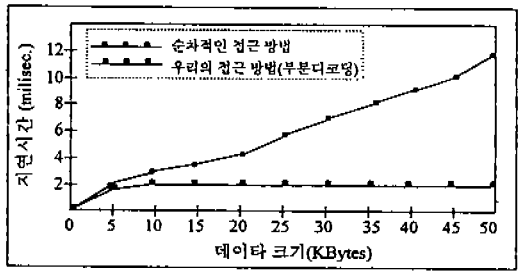


(그림 8) T3의 내부 구현 구조
(Fig. 8) Internal Implementation Structure in a Transputer 3

(그림 8)은 (그림 7)의 T3를 확대한 것으로서, Dispatcher 프로세스, 병렬로 수행하는 4개의 프로토콜 프로세스, 이들간의 동기화와 연결 관리를 수행하는 콘텍스트 관리자 프로세스, 입출력 프로세스 등으로 구성된다. Transputer간의 인터페이스를 위하여 입력 큐와 입력 프로세스, 출력 큐와 출력 프로세스 등이 존재하고, 병렬로 수행되는 각 프로토콜 프로세스들을 동기화하는 콘텍스트 관리자 프로세스 등이 있다. 각 프로토콜 프로세스들은 "PAR"를 이용하여 병렬로 수행된다.

5. 분 석

성능 분석은 시스템에서 제공하는 절대 타이머인 clock()을 사용하였다. 즉, 시스템이 실행될 때 타이머가 일정하게 증가되는데 1초에 15620 tick을 갖는다. 다시 말하면, 1 tick은 64 us를 의미한다. PDU 처리시에 체크섬은 메모리를 계속 접근하므로 상당한 시간이 소요된다. 45 옥텟으로 구성된 망 계층 헤더의 체크섬을 수행한 결과 평균 0.512 ms가 소요 되었다[10]. 만약 MTU 크기가 4354 옥텟으로 구성된 고정 패킷 구조일 경우 체크섬 시간은 약 50ms가 소요된다. 따라서 체크섬은 망으로부터 데이타를 받으면 체크섬 전용 하드웨어에서 실행됨을 가정한다.



(그림 9) ASN.1/BER 디코딩을 시작하기 전의 지연 시간
(Fig. 9) Time delay before starting the ASN.1/BER decoding

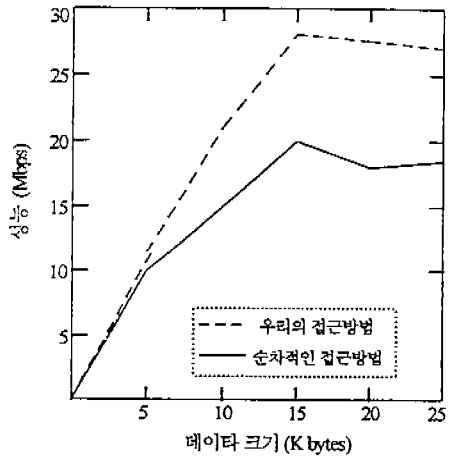
(그림 9)는 순차적인 접근 방법과 우리의 접근 방법에서 ASN.1/BER 디코딩을 시작하기 위한 지연 시간을 보여준다. 다시 말하면, 데이터 패킷을 받은 시간과 ASN.1/BER 디코딩을 시작한 시간과의 차이를 보

여준다. 우리 모델은 전체 PPDU를 받지 않아도 바로 ASN.1/BER 디코딩을 수행할 수 있다는 점이다. 따라서 데이타 크기가 크면 클수록 시간상의 장점을 얻는다.

(그림 10)은 데이타 크기에 따른 성능을 보여준다. 성능 비교는 ASN.1/BER 디코딩을 사용한 것으로서 순차적인 접근 방법과 우리의 접근 방법을 비교해 보면 약 61% 성능 향상을 보여준다. 그 이유는 다음과 같다. 첫째, 순차적인 접근 방법은 프로토콜 계층 간 순서상 제약 때문에 망 프로토콜을 수행한 후 수송 프로토콜을, 수송 프로토콜을 수행한 후 세션 프로토콜을, 세션 프로토콜을 수행한 후 표현 프로토콜을 순차적으로 수행한다. 둘째, 순차적인 접근 방법은 세션 프로토콜과 수송 프로토콜에서 분할 및 재조립 문제를 발생시킨다. 수신측에서 보면, 분할된 PDU를 모두 받아 상위 계층으로 전달해야 하는데 무한히 기다릴 수 없기 때문에 재조립 타이머를 요구한다. 셋째, 표현 번역 시간이 전체 프로토콜 처리 시간의 약 90% 이상을 차지하는[1, 2]데도 불구하고 세션 PDU의 모든 세그먼트를 받은 후에 표현 번역을 수행시키는 문제점이 있다. 우리의 모델은 고정 패킷을 사용하여 수평적인 처리 구조로 수행시켰고, 망으로부터 패킷을 받자마자 전체 PPDU를 받지 않은 상태에서 부분 ASN.1/BER 디코딩을 수행할 수 있었기 때문이다.

또한, (그림 10)에서 데이터 크기가 15KBytes 까지 데이터가 크면 클수록 비례하여 성능이 좋아지다가 15KByte를 넘어서면 성능이 더 이상 증가되지 않는다. 성능은 데이터 크기가 15KByte일 때 28Mbps와 20Mbps, 20KBytes일 때 27Mbps와 17Mbps, 25Kbytes일 때 26.5Mbps와 18.5Mbps이다. 데이터 크기에 따른 성능 측면에서 볼 때 우리 모델은 약 25Mbps~28Mbps 정도이고, 계층 모델은 약 17Mbps~20Mbps 정도이다. 즉 데이터 크기가 커질수록 어느 정도 까지 비례하여 성능이 좋아지다가 포화 상태에서 맴돌고 있음을 알수 있었다. (그림 10)에서 데이터 크기가 25KByte까지 표시했지만, 30KByte와 50Kbyte일 때에도 포화 상태에 있음을 알 수 있다.

우리 모델은 고정 패킷 구조를 사용함으로써 더많은 망의 대역폭을 사용하고 있다. 우리 모델(고정 패킷)과 계층 모델(OSI 패킷)의 망 대역폭을 계산하기 위하여 표현 헤더(PH), 세션 헤더(SH), 수송 헤더(TH),



(그림 10) 데이타 크기에 따른 성능 측정
(Fig. 10) Performance measures : throughput *versus* data size

망 헤더(NH) 등의 크기가 같고, PPDU 크기와 SPDU 크기의 비율이 m, SPDU 크기와 TPDU 크기의 비율이 n, TPDU 크기와 NPDU 크기의 비율이 k이라고 가정한다.

- 패킷 크기가 MTU 크기 보다 작거나 같으면 OSI 패킷이나 고정 패킷이 둘다 하나의 패킷만을 구성하므로 대역폭 오버헤드는 같다.
- 패킷 크기가 MTU 크기 보다 클 경우 고정 패킷의 대역폭 오버헤드는 PH +mnk(SH +TH +NH)이고, OSI 패킷의 대역폭 오버헤드는 PH +mSH +mnTH +mnkNH이다. 고정 패킷의 대역폭 오버헤드에서 OSI 패킷의 대역폭 오버헤드를 빼면 m(nk−1)SH +mn(k−1)TH이다. 따라서 우리 모델은 계층 모델 보다 m(nk−1)SH +mn(k−1)TH 만큼 대역폭을 더 소비한다.

6. 결 론

분산 멀티미디어 응용 중에서 영상회의 시스템을 분석한 결과 다양한 전송 품질을 요구하지만, 가장 중요한 특성은 높은 성능과 적은 전송 지연이었다. 고속 통신망은 Gbps 수준의 전송능력을 갖고 있지만 멀티미디어 응용까지 제공되지 못하고 있다. 이러한 문제점을 해결하기 위하여 통신 구조상 허리 부분을

수평 구조화 하는 모델을 제시하고 프로토타입핑 하였다. 이를 위하여 우리는 고정 패킷을 사용하였다. 고정 패킷은 OSI 프로토콜 계층 간의 순서상 제약 조건(예로서 분할 및 재조립)을 제거함으로써 계층간 독립성을 확보하고, 망 계층에서부터 표현 계층 까지 자기가 처리할 헤더와 사용자 데이타의 위치를 알고 병렬로 수행할 수 있었다. 각 계층의 프로토콜 기능을 변경시키지 않고, 수평 구조화된 프로토콜 계층을 프로토타입핑 함으로써 순차적인 OSI 계층 구조 프로토콜 보다 약 61%의 성능을 향상시켰다.

본 논문에서의 장점은 고정 패킷 구조를 OSI 프로토콜에 적용시킴으로써 패킷의 해석과 프로토콜 처리 메카니즘이 간단해지고, 비용이 절감될 수 있고, 하드웨어화 하는데 용이하고, 각 프로토콜 계층을 다중 프로세서로 매핑함으로써 병렬 처리에 용이하다는 점 등이다. 또한, 프로토콜 계층이 수평 구조화로 되기 때문에 망으로 부터 패킷을 받자마자 표현 번역을 수행할 수 있다는 점이다. 그러나, 본 논문에서의 문제점은 고정 패킷을 사용함으써 발생한다. 고정 패킷의 전체 길이는 MTU 크기에 따라 결정되기 때문에 표현 연결 설정시에 MTU 크기를 알아야 한다. 고정 패킷은 고정 필드 구조이기 때문에 헤더 디코딩 시간을 줄이는 반면 선택 필드(option field)를 통한 다양한 서비스를 제공할 수 없다. 고정 패킷은 항상 세션 헤더와 수송 헤더를 포함하므로 분할 및 재조립이 발생하는 OSI 패킷에 비해 망의 대역폭을 더많이 소모한다. 그러나, 프로토콜을 고속화하기 위하여 망의 대역폭을 더많이 소모한다 하더라도 처리 시간을 단축시킬 수 있다면 의미가 있다고 생각한다.

참 고 문 헌

[1] Z. Hass, "A Protocol Structure for High-Speed Communication over Broadband ISDN," *IEEE Network Magazine*, pp.64-70, Jan. 1991.

[2] D. Clark and D. Tennenhouse, "Architectural Consideration for a New Generation of Protocols," *ACM SIGCOMM'90*, pp.200-209, Philadelphia Pennsylvania, Sep. 24-27, 1990.

[3] D. B. Hehmann, M. G. Salmony and H. J. Stuttgen, "High-Speed Transport Systems for Multimedia Applications," *Proc. IFIP WG6.1/6.4 Int. Workshop on Protocols for High Speed Networks*, pp.303-321, Zurich Switzerland, May 9-11, 1989.

[4] A. Danthine, "Esprit Project OSI 95: New transport services for high-speed networking," *Computer Networks and ISDN Systems*, Vol.25, pp. 384-399, 1992.

[5] W. Doeringer and et. al, "A Survey of Lightweight Transport Protocols for High-Speed Networks," *IEEE Trans. on Communication*, Vol.38, No.11, pp.2025-2039, Nov. 1990.

[6] M. Zitterbart, "High-Speed Transport Components," *IEEE Network Magazine*, pp.54-63, Jan. 1991.

[7] S. Choi and K. Chon, "Partial ASN.1/BER Decoding Scheme in a Multiprocessor Environment," *Computer Communications*, 1994, accepted for pubilication.

[8] O. G. Koufopavlou, A. N. Tantawy, and M. Zitterbart, "Analysis of TCP/IP for High Performance Parallel Implementations," *Proc. 17th Conference on Local Computer Networks*, pp.576-585, Minneapolis Minnesota, Sep. 13-16, 1992.

[9] G. Chesson, "XTP/PE Design Considerations," *Proc. IFIP WG6.1/6.4 Int. Workshop on Protocols for High Speed Networks*, pp.27-33. Zurich Switzerland, May 9-11, 1989.

[10] S. Choi and K. Chon, "Implementing OSI Protocol Stack in a Multiprocessor Environment," *IEICE Trans. on Communications*, 1995, submitted.

[11] 이재영외 4명, "다자간 영상회의 시스템," *정보과학회지 제14권 제5호*, pp.60-74, May 1996.

[12] S. Choi and K. Chon, "Analysis on High Speed Communication Architecture," *WCCW'92*, pp. 11-20, Jan. 1992.

[13] M. Zitterbart, B. Stiller, and A. N. Tantawy, "A Model for Flexible High-Performance Communication Subsystems," *IEEE JSAC*, Vol.11, No.4, pp.507-518, May 1993.

[14] M. Zitterbart, "High-Speed Protocol Implementations Based on a Multiprocessor Architecture,"

Proc. IFIP WG6.1/6.4 Int. Workshop on Protocols for High Speed Networks, pp.151-163, Zurich Switzerland, May 9-11, 1989.

[15] D. Clark and et. al, "An Analysis of TCP Processing Overhead," *IEEE Communication Magazine*, pp.23-29, Jun. 1989.

[16] INMOS Ltd., *Transputer Reference Manual*, Prentice Hall, 1988.

[17] Parsec Developments, *Par. C System:User's Manual and Library Reference*, Parsec Developments, 1990.

[18] Perihelion Software Ltd., *The Helios Parallel Operating System*, Prentice-Hall, 1991.

김 평 중

1985년 충남대학교 계산통계학과(이학사)
1995년 KAIST 전산학과(공학석사)
1995년 정보처리기술사 취득
1987년~1988년 포항종합제철(주) 전산시스템부 4급
1988년~현재 한국전자통신연구소 멀티미디어연구부 선임연구원
관심분야:컴퓨터네트워크, 프로토콜 공학, 그룹웨어, 분산 멀티미디어 등

박 치 항

1974년 서울대학교 응용물리학과(이학사)
1980년 한국과학원 전산학과(이학석사)
1987년 파리 6대학 전산학과(공학박사)
1974년~1978년 한국과학기술연구소 연구원
1978년~1985년 한국전자기술연구소 실장
1985년~현재 한국전자통신연구소 컴퓨터연구단 단장
관심분야:데이타베이스, 분산시스템,컴퓨터네트워크, 멀티미디어, 에이전트기반시스템, 네트워크 컴퓨팅, 등.