멀티미디어 스트림 제어 및 관리를 위한 CORBA 기반 분산 멀티미디어 통신 시스템

정회원 박 상 윤*, 정 길 호*, 박 상 현**, 엄 영 익*

CORBA-based Distributed Multimedia Communication Systems for Controlling and Managing Multimedia Streams

Sang Yun Park*, Gil Ho Joung*, Sang-hyun Park**, Young Ik Eom* ***Regular Members***

요 약

분산 환경에서 멀티미디어 서비스가 활성화됨에 따라 다양한 멀티미디어 통신 기술 개발의 필요성이 제기되고 있다. 분산 환경에서 멀티미디어 스트림의 효율적인 전송과 멀티캐스팅 및 멀티미디어 응용의 상호운용성 등을 지원하기 위하여 개방형 멀티미디어 전용 통신 시스템의 개발과 멀티미디어 스트림의 제어 및 관리 방안의 제공이 요구되고 있다.

분산 환경에서 멀티미디어 스트림의 실시간 전송을 지원하기 위하여 RTP, RSVP, RTSP 등의 프로토콜들이 표준화된 바 있고, 멀티미디어 응용의 상호운용성과 스트림의 제어 및 관리 기능을 지원하기 위한 분산 처리를 위한 플랫폼으로서 OMG의 CORBA, DAVIC의 DSM-CC 등이 발표된 바 있다.

본 논문에서는 멀티미디어 스트림의 실시간 전송, 제어 및 관리와 멀티미디어 응용의 상호운용성을 지원할 수 있는 CORBA 기반의 분산 멀티미디어 통신 시스템을 소개하고, 본 시스템과 VoD 시스템의 연동을 통한 QoS 측정 결과를 제시한다.

ABSTRACT

According to the popular use of the multimedia services in distributed environments, the need for the variable multimedia communication technologies have been presented today. In order to support the efficient transmission of media streams, the multicasting, and the interoperability between multimedia applications in distributed environments, the development of the open exclusive multimedia communication systems and the provision of the control and management methodologies for media streams have been required.

In order to support the realtime transmission of multimedia stream in distributed environments, the protocols such as RTP, RSVP, and RTSP have been standardized. For supporting the interoperability between multimedia applications and the control and management of streams, the distributed processing platforms such as CORBA by OMG and DSM-CC by DAVIC were published.

In this paper, we introduce the CORBA-based distributed multimedia communication systems which support the realtime transmission, control, and management of multimedia streams and provide the interoperability between multimedia applications. Finally, we present the QoS measurement results by the interoperability between our systems and VoD systems.

* 성균관대학교 전기전자 및 컴퓨터공학부 분산시스템연구실 ** 현대정보기술
논문번호: 010218-0821, 접수일자: 2001년 8월 21일

Ⅰ. 서 론

네트워크 기반 멀티미디어 서비스가 활성화됨에 따라 네트워크를 통한 멀티미디어 스트림의 실시간 전송을 지원하는 스트림 전송 기술들이 발표되고 있다. 또한, 분산 환경을 위한 멀티미디어 응용의 개발, 멀티미디어 스트림의 제어 및 관리, 멀티미디어 응용간의 상호운용성 등을 지원하기 위하여 분산 객체 기술과 멀티미디어 기술의 연동이 시도되고 있다.

분산 환경에서 멀티미디어 스트림을 실시간으로 전송하기 위해서 스트림의 실시간 전송을 지원하는 RTP (Realtime Transport Protocol), 자원 예약을 통한 스트림 전송을 지원하는 RSVP (Resource reSerVation Protocol), 스트림의 실시간 전송과 실시간 제어 및 관리를 지원하는 RTSP (Real-Time Streaming Protocol) 등의 프로토콜들이 표준화된 바 있고, 멀티미디어 응용의 상호운용성과 스트림의 제어 및 관리 기능을 지원하기 위해서 분산 객체 기술과 멀티미디어 기술의 연동 플랫폼으로서 OMG (Object Management Group)의 CORBA (Common Object Request Broker Architecture), DAVIC (Digital Audio VIsual Council)의 DSM-CC (Digital Storage Media- Command and Control) 등이 발표된 바 있다.

본 논문에서는 멀티미디어 스트림의 실시간 전송, 스트림의 제어 및 관리와 멀티미디어 응용의 상호운용성을 지원할 수 있는 CORBA 기반의 분산 멀티미디어 통신 시스템의 구조와 기능을 소개하고, 본 시스템과 VoD (Video on Demand) 응용을 통합한 VoD 시스템의 구조와 동작 원리를 설명하며, VoD 시스템을 통해 산출된 QoS (Quality of Service) 측정 결과를 제시한다.

본 논문의 2장에서는 멀티미디어 통신 분야의 관련 연구를 소개하고, 3장에서는 분산 멀티미디어 통신 시스템의 구조와 기능을 설명한다. 4장에서는 분산 멀티미디어 통신 시스템과 VoD 응용을 통합한 VoD 시스템의 구조 및 기능, 연동에 의해 산출된 QoS 기반 성능 측정 결과를 제시한다. 5장에서는 요약 및 분산 멀티미디어 통신 시스템의 기대 효과를 기술한다.

Ⅱ. 멀티미디어 통신을 위한 기존 기술

1. 멀티미디어 통신 프로토콜

1.1 RTP

RTP는 오디오, 비디오, 시뮬레이션 데이터와 같은 실시간 특성을 갖는 데이터를 네트워크 종단간 배달하는 역할을 수행한다. RTP는 실시간 특성을 갖는 데이터의 형식을 정의하는 표준 프로토콜들과의 조합을 통해 완성될 수 있는데, RTP 패킷의 탑재 데이터를 구성하는 프로토콜들에게 인코딩을 위한 토대를 제공한다. RTP의 조력자 역할을 하는 RTCP (Real-time Transport Control Protocol)는 선택적으로 전송되며, RTP 패킷의 배달 감시와 QoS 및 참가자 정보 관리 기능 등을 제공한다. RTP와 RTCP는 하위 전송 및 네트워크 계층에 독립적으로 설계되었으며, RTP 패킷의 변환을 수행하는 번환기 (translator)와 RTP 패킷들간의 조합을 수행하는 혼합기 (mixer)를 지원한다 [1].

RTP는 탑재 유형 식별, 순서 번호화, 타임스탬핑, 배달 감시 등의 서비스를 제공하지만, 정시간 배달, QoS 보장 배달, 비순서화 배달 보장, 신뢰성 있는 하위 네트워크 및 패킷의 순서화 전송 등은 제공하지 않는다. RTP는 통상 하위 전송 및 네트워크 프로토콜로써 UDP와 IP를 사용하는데 이에 따라 멀티플렉싱, 헤더 체크섬, 멀티캐스팅과 같은 기능을 하위 프로토콜 계층으로부터 획득할 수 있다. RTP의 대상 응용은 오디오 및 비디오 회의, 분산 시뮬레이션 모델, 실시간 제어 및 측정 등이다 [2].

RTP는 실시간 데이터 배달, 응용 단계 프레임화, 통합 계층 처리 등의 특성을 가지며 RTCP는 QoS 감시, 세션간 참가자들의 정보 전달 등의 기능을 수행하는 멤버쉽 제어와 셋업이 필요 없는 'loosely controlled' 메커니즘을 지원한다 [3].

1.2 RSVP

RSVP는 수신자가 종단간에 데이터 전송에 대한 QoS를 지정할 수 있는 네트워크 제어 프로토콜이다. 실시간 응용은 전송이 시작되었을 때 요구된 대역폭을 보장하기 위하여 RSVP를 사용하여 전송 경로 상에 존재하는 라우터들의 자원을 필요한 만큼 예약한다.

RSVP에서 하나의 자원 예약은 송신 측과 수신 측 중 하나에 대한 단 방향 예약을 의미한다. RSVP는 멀티캐스트와 유니캐스트를 모두 지원하고 세션에 참여하는 멤버의 변동이나 경로의 변경에 대한 적응력을 가진다. 자원 예약은 수신자에 의해

시발되고 자원 예약 상태는 가변적이기 때문에 RSVP는 쉽게 세션의 멤버쉽과 경로를 동적으로 제어할 수 있다. 세션 참여를 원하는 호스트는 IGMP (Internet Group Management Protocol) 메시지를 송신하여 멀티캐스트 그룹에 참여할 수 있고 RSVP의 자원 예약은 송신자에 대한 과부하 없이 큰 규모의 멀티캐스트 그룹을 구성하도록 지원할 수 있다 [4].

이질적인 멀티캐스트 그룹들에 속한 수신자들은 각각 다른 용량과 단계의 QoS를 갖는다. 수신자 기반 RSVP 자원 예약 방식은 이질적인 멀티캐스트 그룹에 속한 수신자들의 개별적인 QoS에 대한 관리를 가능하게 한다. 수신자들은 자신의 QoS 단계를 선택하고 원하는 만큼 자원 예약 상태를 지속할 수 있다. 송신자는 통신량을 다양한 QoS 단계로 구분하여 제공하고 수신자는 송신자의 다양한 QoS 단계의 흐름 중 하나 이상을 선택할 수 있다. 이러한 접근 방식은 이질적인 수신자들의 상이한 QoS 요구에 대한 처리를 가능하게 한다.

RSVP는 IPv4 또는 IPv6 상에서 모두 동작할 수 있다. RSVP는 새로운 기술에 대한 호환성 및 미지원 지역에 대한 전송 투명성을 위해 통신량 제어와 정책 제어 메시지 등을 스트림 형태로 전송한다.

1.3 RTSP

RTSP는 인터넷을 통한 멀티미디어 스트림의 제어된 전송을 지원하는 클라이언트/서버 멀티미디어 프리젠테이션 프로토콜이다. RTSP는 멀티미디어 스트림에 대해 일시중지, 탐색, 되감기 및 재생 위치 지정 등과 같은 원격 제어 기능을 제공한다.

RTSP는 인터넷을 통한 스트림 전송 기능을 제공하는 RTP, RSVP 등을 하위 프로토콜로 하여 이들과 상호작용하는 응용 계층 프로토콜로 설계되었고, 스트림 전송을 위해 UDP나 TCP와 같은 배달 채널이나 RTP 기반의 배달 메커니즘 등을 선택적으로 사용하며 유니캐스팅과 멀티캐스팅을 모두 지원한다 [5].

RTSP는 HTTP와 유사한 문법과 연산 방식을 따르지만 멀티미디어를 대상으로 하는 프로토콜이다. RTSP는 HTTP와 같이 URL을 사용하고 RTSP 서버는 다양한 메소드를 사용하여 전송 상태를 관리하며 상태 정보는 스트림 전송을 위한 대역 외 채널을 통해 교환된다. RTSP는 데이터 전송 프로토콜과는 구별되며 HTTP와는 다르게, 서버와 클라이언트 모두 요구를 시작할 수 있다. RTSP는 다양한 운영체제 플랫폼에서 동작할 수 있도록 고안되었고 서로 다른 개발사의 클라이언트와 서버들간의 상호운용성을 지원한다.

2. 멀티미디어 제어 및 관리를 위한 플랫폼

2.1 OMG CORBA

OMG에서 제시한 CORBA의 멀티미디어 제어 및 관리 표준에서는 멀티미디어 스트림의 제어와 관리에 필요한 세부 기능들을 IDL (Interface Definition Language) 인터페이스를 통해 정의하고 있다.

이 표준에서는 멀티미디어의 제어 및 관리를 위해 스트림의 공급과 소비, 제어를 담당하는 스트림 종점을 정의하고 있는데, 스트림 종점은 세 가지 논리적 구성 요소들로 이루어져 있다. 스트림 인터페이스 제어 객체 (stream interface control object)는 스트림을 제어 및 관리하기 위한 IDL 정의를 포함하고 있으며 객체 어댑터 (OA : Object Adaptor)를 통해 제어 메시지를 송·수신한다. 스트림 소스 (stream source)와 스트림 싱크 (stream sink)는 멀티미디어 스트림을 생산하거나 소비하는 최종 종점이며, 스트림 어댑터 (stream adaptor)는 멀티미디어 스트림의 송·수신 기능을 담당한다 [6].

제어 흐름은 신뢰성이 요구되기 때문에 TCP/IP 기반의 IIOP (Internet Inter-ORB Protocol) 프로토콜을 사용하여 전송한다. 멀티미디어 데이터들은 스트림 형태로 전송되는데, 스트림은 각각의 미디어에서 발생한 플로우 (flow)들로 이루어지고, 플로우는 단방향으로 전송되는 프레임 (frame)들의 집합으로 구성되어 있다. OMG의 멀티미디어 스트림 제어 및 관리 표준에서는 이러한 멀티미디어 스트림의 전송을 위해 스트림 전송에 적합한 다양한 전송 프로토콜을 지원하고 컨텐트의 전송 형식에 독립적인 구조를 제공하는 SFP (Simple Flow Protocol)라는 메시지 레벨의 프로토콜을 권고하고 있다 [7].

분산 멀티미디어 서비스에서는 동적으로 변화하는 네트워크 상태에 따라 서비스의 질이 유동적일 수 있다. 따라서 각 응용들은 주어진 환경에서 최대한 사용자의 요구를 충족시키기 위해서 미디어의 압축율이나 프레임율 등의 멀티미디어 속성들을 동적으로 제어하는 정책을 사용할 수 있다. 이를 위해 OMG의 멀티미디어 스트림 제어 및 관리 표준에는 연결 설정을 위한 응용 레벨의 QoS과 네트워크 레벨의 QoS를 정의하고 있다 [8][9].

2.2 DAVIC의 DSM-CC

DAVIC에서 제시한 DSM-CC는 이기종 통신망 환경에서 대화형 비디오 응용을 지원하기 위해 정의된 프로토콜이다. DSM-CC는 MPEG-1과 MPEG-2 스트림과 관련된 제어 채널의 개발을 위한 도구로서, 비디오 수신의 제어와 빠른 검색, 되감기 및 일시 정지 등과 같은 VCR (Video Cassette Recorders)과 유사한 기능 등을 제공한다. 또한 패킷 데이터 전송을 포함하는 다양한 분야에서 널리 사용될 수 있다. DSM-CC는 멀티미디어 제어 및 전송 프로토콜인 RSVP, RTSP, RTP 및 SCP (Session Control Protocol) 등과 연동하여 사용될 수 있다.

DSM-CC 클라이언트는 서버들에게 다중 세션을 통한 다중 서비스 요청을 동시에 할 수 있고 이질적인 환경에 존재하는 다중 서버들은 서로 상호운용 할 수 있다. 서버의 자원은 다중 클라이언트에 의해 순차적 또는 병렬적으로 접근 가능하다. DSM-CC 환경에서의 클라이언트들과 서버들의 연결 방식은 서버와 임의의 다중 클라이언트들로의 연결로서 broadcast 방식, 서버와 클라이언트의 일-대-일 연결로서 point to point 방식, 서버와 선택된 다중 클라이언트들간의 연결로서 multicast 방식 및 다중 서버들과 다중 클라이언트들의 다-대-다 연결로서 multi-point to multi-point 등이 있다 [10].

DSM-CC 클라이언트와 서버와의 통신은 다양한 네트워크 프로토콜을 거쳐 이루어지는데, 이 때 상호작용하는 프로토콜들은 DSM-CC 응용에 투명한 서비스를 제공해야 한다. 또한, DSM-CC 프로토콜은 범용 응용, MPEG 응용 및 스크립트 언어 등에게 연결을 수립하거나 종료하는 User-Network 프리미티브와 클라이언트와 서버간의 통신을 지원하는 User-User 프리미티브 등을 제공해야 한다. DSM-CC는 MPEG1 시스템 스트림, MPEG2 전송 스트림 또는 MPEG2 프로그램 스트림 등에 포함되는 스트림으로 동작하거나 기타 TCP 또는 UDP 프로토콜과 연동하여 동작할 수 있다 [11].

Ⅲ. 분산 멀티미디어 통신 시스템

본 논문에서 소개하는 분산 멀티미디어 통신 시스템은 멀티미디어 스트림과 QoS 정보의 전송을 위하여 RTP, RTCP 프로토콜을 사용하며 멀티미디어 세션의 제어 및 관리를 위하여 CORBA 기반의 제어 및 관리 표준을 준수한다.

1. 시스템 구조 및 기능

분산 멀티미디어 통신 시스템은 클라이언트와 서버 측의 실시간 응용들간의 멀티미디어 스트림의 전송, 스트림 전송에 대한 QoS 정보의 교환, 클라이언트와 서버 측의 제어 및 관리 정보의 교환, 실시간 응용들간의 서비스 상호운용 등의 기능을 지원한다.

(그림 1)에서 예시하는 바와 같이 분산 멀티미디어 통신 시스템은 클라이언트 측과 서버 측으로 구분된다. 클라이언트 측의 실시간 응용은 미디어의 디스플레이를 지원하는 플레이어 및 코덱을 포함하고 서버 측의 실시간 응용은 미디어 생성을 지원하는 미디어 장치, 코덱 및 저장된 미디어의 관리를 위한 DB 연동 장치 등을 포함한다. 클라이언트 측과 서버 측은 내부 구성 요소로서 제어 및 관리 모듈과 스트림 통신 모듈을 포함하는데, 스트림 통신 모듈은 스트림을 RTP 프로토콜을 통해 전송하거나 스트림의 QoS 정보를 RTCP 프로토콜을 통해 전송한다. 클라이언트 측의 제어 및 관리 모듈은 미디어 플레이어로부터 발생한 미디어 제어 이벤트를 CORBA 기반의 ORB 버스를 통해 서버 측으로 전송하고 서버 측의 제어 및 관리 모듈은 이를 수신하여 코덱을 제어하여 미디어의 재생, 정지, 되감기 및 검색 등의 기능을 수행한다.

클라이언트 측과 서버 측은 미디어와 QoS 데이터 전송을 위한 멀티미디어 연결과 제어 및 관리 정보의 전송을 위한 CORBA 기반 연결을 공유한다. 멀티미디어 연결은 미디어의 성능 및 실시간 특성 등을 고려한 UDP 프로토콜 기반의 RTP/RTCP 프로토콜 메시지를 전송하고, CORBA 기반 연결은 제어 요구의 신뢰성을 고려한 TCP 프로토콜 기반의 IIOP 프로토콜 메시지를 전송한다.

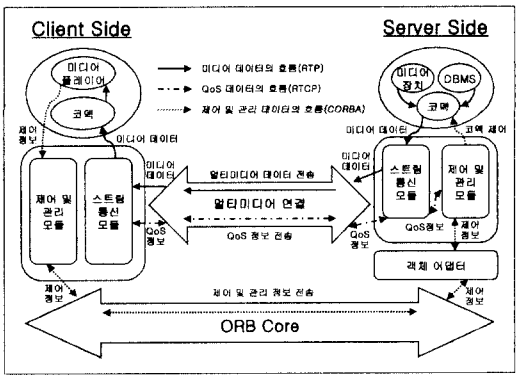


그림 1. 분산 멀티미디어 통신 시스템의 구조

분산 멀티미디어 통신 시스템의 기능은 다음과 같다.

- 멀티캐스트 및 유니캐스트를 지원하는 멀티미디어 세션의 생성 및 종료
- RTP/RTCP 패킷의 인코딩 및 디코딩
- 실시간 스트림에 대한 RTP/RTCP 패킷의 전송
- QoS 정보의 생성, 가공 및 적재
- RTCP 패킷을 통한 QoS 정보 교환
- QoS 분석을 통한 실시간 응용의 제어
- CORBA 기반 세션의 생성과 종료
- CORBA 기반 세션을 통한 제어 및 관리 정보 전송

2. 스트림 통신 모듈

미디어 스트림의 전송과 스트림 전송에 대한 QoS 정보 교환을 지원하는 스트림 통신 모듈은 응용 연결 인터페이스 블록, 스트림/QoS 통신 블록 및 네트워크 인터페이스 블록 등으로 구성된다 [11][12].

2.1 구성 요소

2.1.1 응용 연결 인터페이스 블록

(그림 2)에서 예시하는 바와 같이 응용 연결 인터페이스 블록은 실시간 응용과 스트림 통신 모듈의 내부 구성 블록들간의 인터페이스를 제공한다. 또한 실시간 응용의 요구에 따라 스트림/QoS 통신 블록을 초기화하고 세션을 생성시키거나 종료하며 RTP/RTCP 패킷을 송·수신하는 등 스트림/QoS 통신 블록을 구동하여 실시간으로 스트림을 송·수신하고 QoS를 관리하여 이를 실시간 응용에 반영할 수 있게 하는 종합적인 관리 블록이라 할 수 있다.

2.1.2 네트워크 인터페이스 블록

(그림 2)에서 예시하는 바와 같이 네트워크 인터페이스 블록은 스트림/QoS 통신 블록과 하위 네트워크간의 인터페이스를 제공한다. 본 블록은 하위 네트워크 프로토콜로서 UDP/IP를 사용하기 때문에 비연결지향의 데이터그램을 생성하여 송·수신하며 UDP 소켓 인터페이스를 사용하여 네트워크 계층에 접근한다. 네트워크 인터페이스 블록은 RTP 송신기, RTP 수신기, RTCP 송신기, RTCP 수신기, 데이터그램 처리기 등으로 구성되는데, RTP 수신기, RTCP 송·수신기 등은 쓰레드 형태로 세션 초기화 과정에서 구동되어 패킷 송·수신 대기 상태에 들어가고 RTP 송신기는 실시간 응용에서 비 정기적으로 발생하는 실시간 스트림의 생성 여부에 따라 호출되어 동작한다. 데이터그램 처리기는 스트림/QoS 통신 블록의 RTP/RTCP 패킷 처리기로부터 전달된 인코딩된 RTP 패킷 스트림을 데이터그램 형태로 생성하고 RTP 송신기로 전달하거나 수신된 데이터그램으로부터 RTP 패킷 스트림을 추출하여 RTP/RTCP 패킷 처리기에서 디코딩 될 수 있게 한다. RTCP 송신기는 최초 세션의 시작시에 활성화되어 타이머를 사용하여 주기적으로 RTCP 복합 패킷을 전송하므로 세션간에는 응용 연결 인터페이스 블록과의 상호작용이 없이 자체적으로 스트림/QoS 통신 블록의 QoS 정보 관리기와 주기적으로 연동하여 동작한다. RTP 수신기 및 RTCP 수신기는 수신된 데이터그램을 데이터그램 처리기를 통하여 패킷을 추출한 후에 QoS 정보 관리기에 관련 QoS 정보를 전달하고 RTP 수신기의 경우, 추출된 실시간 스트림을 응용 연결 인터페이스 블록을 통하여 실시간 응용에 전달한다.

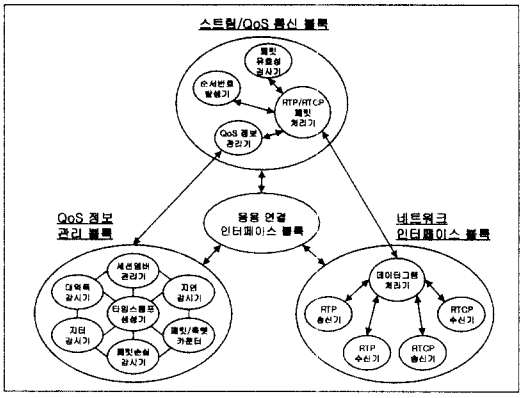


그림 2. 스트림 통신 모듈의 구조

2.1.3 스트림/QoS 통신 블록

(그림 2)에서 예시하는 바와 같이 스트림/QoS 통신 블록은 RTP/RTCP 패킷의 구성 및 인코딩/디코딩을 담당하는 RTP/RTCP 패킷 처리기와 스트림 관련 QoS 정보를 관리하는 QoS 정보 관리기로 구성된다.

RTP/RTCP 패킷 처리기는 순서번호 발생기와 패킷 유효성 검사기로 구성되며 RTP 고정 헤더와 RTCP SR (Sender Report), RR (Receiver Report), SDES (Sender DEScription), BYE, APP (APPlication specific) 등의 RTP와 RTCP 패킷을 구성하여 송신시에는 패킷을 생성하고 수신시에는 필드의 값을 추출하여 다른 블록에 전달하며 생성된 패킷을 통신상에 적합한 스트림 형태의 자료 표현으로 변환하거나 스트림 형태의 통신 자료 표현에서 원시

패킷 필드로의 변환하는 인코딩/디코딩 과정을 수행한다. 전송 자료 표현은 패킷 필드의 데이터 유형에 따라 데이터 형별 바이트 순서 (byte ordering)를 맞추고 데이터 형의 스트림 상의 위치를 정렬하는 등의 작업들로 구성된다. RTP/RTCP 패킷 처리기는 패킷 생성시에 내부 순서 번호 발생기로부터 순서 번호를 획득하여 RTP 고정 헤더에 배정하며 수신시에는 패킷 유효성 검사기를 통해 패킷의 유효성을 검증하게 된다. 순서 번호 발생기는 RTP 고정 헤더의 순서 번호 필드에 순서 번호를 배정한다. 순서 번호의 초기 값은 임의로 배정되지만 그 다음부터는 RTP 패킷의 생성시 마다 단조 증가하는 값이 배정된다. 패킷 유효성 검사기는 수신된 RTP 패킷의 고정 헤더에 대해 RTP 버전, 탑재 유형, 패딩 필드, 확장 필드, 순서 번호 및 SSRC에 대한 유효성 검사를 하며 유효하지 않을 경우 이를 QoS 정보 관리기에 통보한다.

QoS 정보 관리기는 RTP/RTCP 패킷으로부터 전달된 정보를 바탕으로 QoS 정보를 생성하고 적재하며, 이를 실시간 응용 및 세션의 다른 참가자들에게 반영하는 피드백 메커니즘을 제공한다. 실시간 응용은 QoS 정보 관리기로부터 전달된 QoS 정보를 바탕으로 코덱을 제어하여 비디오 및 오디오 스트림의 생성시 탑재 유형을 변환하거나 인코딩 데이터 비율을 조절하여 세션 QoS를 향상시킬 수 있다. 세션의 다른 참가자들은 피드백된 지터, 대역폭 및 지연 정보 등의 QoS 정보를 바탕으로 실시간 응용을 제어하며 세션의 모든 참가자들은 QoS에 관한 상호 조율을 할 수 있다. QoS 정보 관리기는 세션 멤버 관리기, 타임스탬프 생성기, 지연 감시기, 대역폭 감시기, 지터 감시기, 패킷 손실 감시기, 패킷/옥테트 카운터 등으로 구성된다. 세션 멤버 관리기는 현재의 세션의 참가자 정보를 관리하는데 RTCP 패킷를 통해 획득된 참가자 정보 목록은 BYE 패킷이 전달되었을 때나, 지속적으로 참가자에 대한 RTCP 패킷이 도착하지 않아서 비활성 참가자로 판별되었을 때 세션 참가자 목록에서 삭제한다. 참가자 목록은 RTCP SR 또는 RR을 전송할 때 수신 대상이 되므로 RTCP 패킷에 의한 네트워크 부하를 고려하여 적정수준으로 유지해야 하며 이를 위해 RTCP 패킷 수신 간격의 조정에 따른 비활성 참가자의 판별은 매우 중요한 참가자 목록의 유지 방법이 된다. 타임스탬프 생성기는 RTP 고정 헤더 및 RTCP의 SR, RR 등의 RTP 타임스탬프, NTP 타임스탬프, LSR (Last SR), DLSR (Delay since Last SR) 등을 생성 및 관리하는데 RTP/RTCP 패킷 처리 블록의 요청에 따라 RTP/RTCP 패킷 생성을 위한 정보를 제공하거나 RTCP 수신시 전달되는 정보를 바탕으로 LSR, DLSR 등을 가공 후 적재하기도 한다. 지연 감시기는 RTCP SR 패킷의 도착시간, LSR, DLSR 등을 통하여 전송 지연 시간을 계산할 수 있고 이 시간은 현재 참가자별 전송 현황을 파악하는 자료가 된다. 따라서 지연 감시기는 도착하는 RTCP 패킷에 대한 전송 지연 시간의 평가를 통하여 참가자별 네트워크 QoS를 관리할 수 있다. 지터 감시기는 RTCP SR 및 RR에 의해 전달된 도착간 지터 정보를 바탕으로 RTP 패킷 전송에 대한 통계학적 변화를 추정할 수 있다. 도착간 지터는 타임스탬프에 근거하여 송신자 패킷에 비교한 수신자 패킷의 패킷 간격 차이를 표준 편차로 나타낸 것이다. 도착간 지터의 계산은 (표 1)과 같다.

표 1. 도착간 지터의 계산

Si, Sj	: 송신 측의 패킷 i, j의 RTP 타임 스탬프
Ri, Rj	: 수신 측의 패킷 I, j의 출발 시간
D(i, j)	: 패킷 i, j에 대하여 송신 측과 비교한 수신 측의 패킷 간격 차이
J	: 도착간 지터

$$D(i,j) = (Rj - Ri) - (Sj - Si) = (Rj - Sj) - (Ri - Si)$$

$$J = J + (| D(i-1, i) | - J) / 16$$

대역폭 감시기는 RTCP 패킷을 전송할 간격을 결정하기 위해 대역폭을 검출하고 이를 RTCP 타이머에 반영하여 RTCP 패킷이 전체 대역폭의 5% 이내에서 전송될 수 있도록 한다. 대역폭은 초당 비트수를 단위로 하며 대역폭을 검출하는 주기는 시스템의 성능에 따라 동적으로 결정한다. 패킷 손실 감시기는 손실된 RTP 패킷의 손실율 정보를 제공하며 손실율을 계산하기 위해서는 수신 예상 패킷 수와 수신한 패킷 수를 알아야 한다. 수신 예상 패킷 수는 수신자에 의해 수신된 최대 순서 번호와 수신된 첫 번째 순서 번호의 차로써 계산할 수 있으며, 수신한 패킷 수는 실제로 수신한 RTP 패킷의 수를 사용한다. 패킷/옥테트 카운터는 RTCP SR 및 RR의 패킷 카운트 및 옥테트 카운트 필드에 적용되는 정보를 관리한다. 패킷 카운트는 전송이 시작된 이후 본 SR 패킷이 생성될 때까지 송신자에 의해 전달된 RTP 데이터 패킷의 수를 명시하며, 옥테트 카운트는 전송이 시작된 이 후, 본 SR 패킷이 생성될

때까지 송신자에 의해 송신된 RTP 데이터 패킷의 총 옥테트 수를 명시한다.

3. 스트림 제어 및 관리 모듈

스트림 제어 관리 모듈은 스트림 제어 관리 객체와 클라이언트 및 서버 측의 스트림 제어 객체들로 구성된다. 스트림 제어 관리 객체는 클라이언트와 서버 사이에 위치하여 양 종단 간의 스트림 연결 설정과 미디어 제어와 같은 사용자의 요구를 처리한다. 클라이언트 측의 스트림 제어 객체들은 클라이언트 측의 멀티미디어 장치들을 가상적으로 표현한 객체들과 플로우 종단을 표현한 객체 등 스트림 제어 관리 객체의 제어를 받는 객체들로 구성된다. 서버 측의 스트림 제어 객체들은 스트림 제어 관리 객체의 제어를 받아 서비스 관련 기능을 수행한다. 클라이언트 측과 서버 측은 각 종단의 제어 객체들을 생성하여 관련 정보를 클라이언트와 서버 사이의 스트림 제어 관리 객체에 전달한다. 스트림 제어 관리 객체의 생성은 클라이언트의 스트림 제어 관리 객체 생성 요구가 ORB의 객체 어댑터에게 전달된 후, 객체 어댑터에 의해 이루어진다 [13].

3.1 구성 요소

3.1.1 스트림 제어 관리 객체

스트림 종단 간의 연속 미디어 흐름을 제어하기 위한 StreamCtrl 인터페이스에는 스트림 종단간의 연결 설정이나 스트림의 시작과 정지와 같은 미디어 제어 등의 기능 등이 정의되어 있다. StreamCtrl 객체는 서버에 대한 클라이언트의 연결 설정 요구에 의해 ORB에 의해서 활성화된다. 활성화된 StreamCtrl 객체는 클라이언트 측과 서버 측의 MMDevice 객체 정보를 전달받아 연결을 설정하고, 클라이언트의 제어 요구에 따라 해당 스트림에 대한 제어 기능을 수행한다. StreamCtrl 객체는 기본적인 제어 기능 외에 MediaControl 인터페이스를 상속받아 시작과 정지 기능을 포함한 다양한 미디어 제어 기능을 지원할 수 있다.

분산 멀티미디어 통신 시스템은 스트림 내의 플로우들을 개별적으로나 집단적으로 제어 및 관리할 수 있도록 OMG의 멀티미디어 스트림 제어 및 관리 표준에서 제시한 풀 프로파일 형식의 정의를 준수한다. 이러한 풀 프로파일 형식의 적용은 라이트 프로파일 보다 정교한 스트림 제어를 가능하게 한다.

스트림 내의 각각의 플로우들은 플로우 별로 할당되는 StreamCtrl의 FlowConnection 객체에 의해 제어된다. FlowConnection 객체에 의해 제어되는 각 플로우의 종단에는 FlowEndPoint 인터페이스가 정의된다.

3.1.2 스트림 제어 객체

클라이언트 측과 서버 측 양 종단의 MMDevice 인터페이스는 하나 이상의 멀티미디어 장치를 추상화하여 나타내고 있다. MMDevice 인터페이스는 스트림 종단에서 멀티미디어 디바이스의 속성을 설정하기 위한 VDev 객체와 스트림 종단의 연결 설정을 위한 StreamEndPoint 객체 등을 생성한다.

분산 멀티미디어 통신 시스템은 MMDevice 객체를 클라이언트 측과 서버 측에서 각각 생성한다. 서버 측의 MMDevice 객체는 클라이언트의 생성 요구시에 ORB를 통해 활성화되며, 생성된 클라이언트 측과 서버 측의 MMDevice 객체는 StreamCtrl 인터페이스를 구현한 제어 객체에게 매개변수로 전달되고, 전달된 MMDevice 객체는 StreamCtrl 객체가 스트림 종단 간의 연결을 설정하는데 필요한 VDev 객체와 StreamEndPoint 객체를 생성한다. VDev 객체와 StreamEndPoint 객체는 멀티미디어 스트림을 제어한다. 이외에도 분산 멀티미디어 통신 시스템의 MMDevice 객체는 StreamEndPoint 객체가 관리해야 할 플로우 수만큼 가상의 플로우 장치인 FDev 객체를 생성하여 각각의 FDev 객체에게 요구된 속성을 부여한다.

3.2 동작 시나리오

분산 멀티미디어 통신 시스템에서 Streamctrl 객체는 스트림에 대한 제어 및 관리를 수행한다. (그림 3)에서 예시하는 바와 같이 분산 멀티미디어 통신 시스템의 제어/관리 구성 요소 및 동작 시나리오는 다음과 같다.

(1) 클라이언트는 서버와의 세션 수립 이전에 StreamCtrl 객체 생성을 ORB에게 요구한다.

(2, 3) 생성된 StreamCtrl 객체는 클라이언트와 서버의 연결 수립을 위해 양 스트림 종단에서 생성된 MMDevice 객체에 대한 정보를 수신한다.

(4) 클라이언트와 서버의 MMDevice 객체는 StreamCtrl 객체의 연결 수립 절차에 따라 VDev 객체와 StreamEndPoint 객체를 생성한다.

(5) StreamEndPoint 객체는 MMDevice 객체의 요구에 따라 플로우 제어를 위한 FDev 객체를 생성한다.

(6) FDev 객체는 add_fdev() 메소드를 통해 해당 MMDevice에 등록된다.

(7) FDev 객체는 플로우의 종단인 FlowEndPoint 객체를 생성하는데, FlowEndPoint 객체가 클라이언트 측에 있을 경우에는 FlowConsumer 객체를 생성하고, 서버 측에 있을 경우에는 Flow Producer 객체를 생성한다.

(8, 9, 10, 11, 12) FlowEndPoint 객체는 add_fep() 메소드를 통해 StreamEndPoint 객체에 등록되고, set_format(), related_sep() 등의 메소드를 통해 StreamEndPoint 객체로부터 관련 정보를 획득한다.

(13, 14) StreamEndPoint 객체는 get_Property_Value(), get_fep() 등의 메소드를 통해 StreamCtrl 객체로부터 관련 정보를 획득한다.

(15) 클라이언트 측과 서버 측에 생성된 FlowEndPoint 객체들을 상호 연결 및 제어하기 위해 FlowConnection 객체가 생성된다.

(16, 17) FlowConnection 객체는 set_flow_connection() 메소드를 통해 StreamCtrl 객체에 등록된다.

(18, 19, 20) 플로우의 연결 설정을 위해 FlowConnection 객체는 StreamEndPoint 객체들에 포함되어 있는 FlowEndPoint 객체를 획득하고, FlowConsumer 객체의 go_to_listen() 메소드와 FlowProducer 객체의 connect_to_peer() 메소드를 통해 플로우의 연결이 수립된다. 모든 연결 설정 과정이 끝나면 RTP 통신 모듈이 구동되어 멀티미디어 스트림이 송·수신된다.

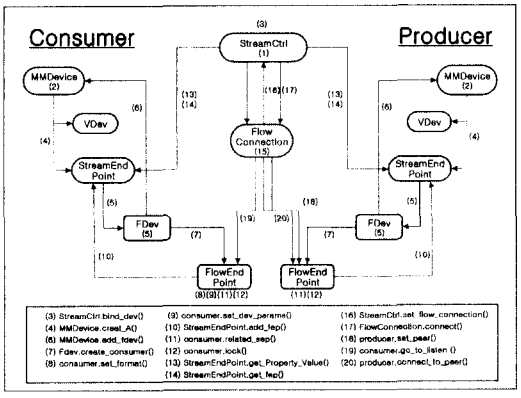


그림 3. 제어 및 관리 구조의 구성 요소와 동작 시나리오

분산 멀티미디어 통신 시스템은 연결 설정이 필요한 플로우의 수 만큼 (5) 단계에서 (20) 단계를 반복하며, 각 플로우들에 대한 제어는 플로우의 이름을 통해 StreamCtrl 객체의 제어에 따라 Flow Connection 객체가 수행한다. 사용자로부터 미디어 제어 인터페이스를 통해 미디어 제어 정보가 전달되면, StreamCtrl 객체는 해당 플로우의 Flow Connection 객체에게 미디어 제어 정보를 전달한다. FlowConnection 객체는 전달받은 제어 정보에 따라 서버 측의 FlowEndPoint 객체의 해당 메소드를 호출하여 미디어 송·수신을 제어한다.

4. 상호운용성

분산 멀티미디어 통신 시스템은 CORBA 환경을 기반으로 동작하기 때문에 분산 객체 시스템간의 상호운용성에 따라 이질적인 환경의 분산 멀티미디어 응용들은 실시간으로 상호운용 할 수 있다. (그림 4)은 이질적인 도메인 A, B, C에 존재하는 화상회의 서버들이 분산 멀티미디어 통신 시스템을 통해 상호운용하는 모델을 예시한다.

각 도메인의 ORB는 분산 객체 기반 화상 회의 서버와 분산 멀티미디어 통신 시스템을 응용 서버로 탑재하고 지역 도메인의 클라이언트들을 관리한다. 클라이언트와 ORB, ORB와 ORB간에는 분산 멀티미디어 통신 시스템을 통한 RTP/RTCP 기반 멀티미디어 연결과 IIOP 기반 제어 및 관리 연결이 수립된다. 이질적인 도메인에 속한 동일 세션의 클라이언트들은 지역 ORB의 화상회의 서버로부터 세션 관련 정보를 획득하고 도메인 내부 또는 다른 도메인에 위치한 클라이언트들과 멀티미디어 연결을 수립한 후 스트림을 실시간으로 교환한다. 이 때 각 클라이언트에서 발생하는 제어 정보는 IIOP 기반 제어 및 관리 연결을 통해 타 도메인의 클라이언트와 상호연동하고, 각 클라이언트에서 발생하는 스트림 관련 QoS 정보는 분산 멀티미디어 통신 시스템을 통해 교환되어 각 도메인의 화상회의 서버에 축적되며 제어 및 관리 연결을 통해 송신 클라이언트들의 코덱에 반영되어 실시간 QoS 관리를 수행한다.

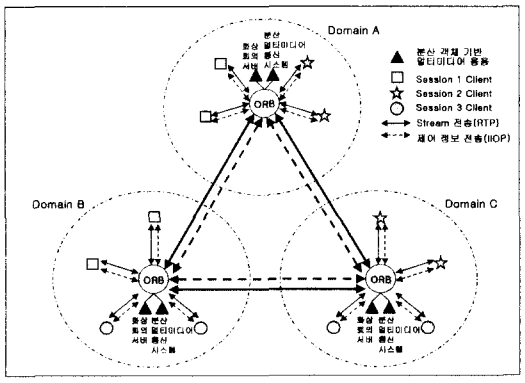


그림 4. 분산 멀티미디어 통신 시스템의 상호운용 구조

Ⅳ. VoD 시스템과의 연동 및 성능 측정

분산 멀티미디어 통신 시스템의 동작과 성능의 평가를 위해 본 논문에서는 CORBA 기반의 VoD 시스템을 설계 및 구현하고 분산 멀티미디어 통신 시스템과 연동하였다.

1. VoD 시스템과의 연동

1.1 VoD 시스템 구조

VoD 시스템은 클라이언트 측과 서버 측으로 구분되는데, VoD 서비스를 요구하는 클라이언트 측은 사용자에게 VoD 서비스를 제공하고 멀티미디어 플레이어를 포함하는 미디어 사용자 인터페이스, 서버와의 제어 정보 송·수신을 위해 분산 멀티미디어 통신 시스템의 제어 및 관리 모듈과 연동하는 VoD 서비스 지원 객체, 멀티미디어 스트림과 QoS 정보를 송·수신하는 분산 멀티미디어 통신 시스템의 스트림 통신 모듈, 수신한 멀티미디어 스트림을 디코딩하여 플레이어에 전달하는 코덱 등의 네 부분으로 구성된다. (그림 5)는 VoD 시스템의 클라이언트 측의 구조를 예시한다 [14].

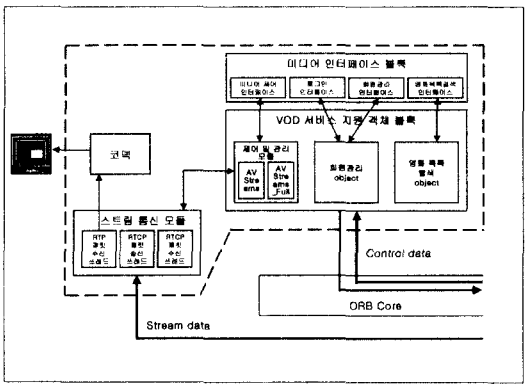


그림 5. VoD 시스템의 클라이언트 측의 구조

VoD 서비스를 제공하는 서버 측은 VoD 시스템 관리자가 서버를 관리하기 위해 필요로 하는 인터페이스들, 영화와 회원들의 정보를 저장하기 위한 데이터베이스, 멀티미디어 스트림의 송신과 QoS 정보의 송·수신을 위한 분산 멀티미디어 통신 시스템의 스트림 통신 모듈, 클라이언트 측의 요구를 지원하기 위한 VoD 서비스 지원 객체들, 비디오 저장장치로부터 전달된 멀티미디어 데이터를 임시 저장하는 버퍼 등 다섯 부분으로 구성된다. (그림 6)는 VoD 시스템의 서버 측의 구조를 예시한다.

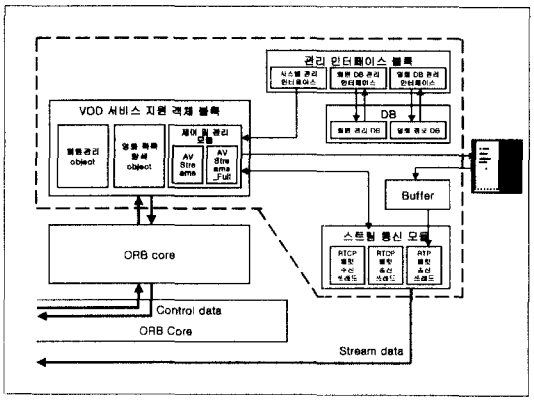


그림 6. VoD 시스템의 서버 측의 구조

1.2 미디어 재생 및 데이터베이스 연동

1.2.1 미디어 재생

VoD 시스템은 VoD 시스템의 미디어 재생을 위해 Sun 등에서 개발한 JMF (Java Media Framework) 2.0 라이브러리를 사용하였다. VoD 시스템에서 JMF 라이브러리 외에 추가된 클래스들로는 미디어 제어 인터페이스의 생성과 미디어 재생 시 발생하는 이벤트 처리를 위한 JMFRTPPlayer 클래스, 미디어 데이터의 접근 정보를 유지하는 RTPMediaLocator 클래스, 미디어 플레이어에게 스트림을 공급하는 RTPStream 클래스, 스트림 통신 모듈을 통해 수신한 멀티미디어 데이터를 RTPStream 클래스에게 전달하는 RTPDataSource 클래스 등이 있다.

RTPStream 클래스는 미디어 플레이어 구동 시 멀티미디어 데이터를 버퍼링하기 위해 임의의 간격 동안 재생을 대기하고, 이 후에 RTPStream 객체 내의 순환 큐를 검사하여 적정량의 스트림이 적재되었는지를 확인한다. RTPStream 객체 내에 있는 순환 큐는 미디어 플레이어에 전달할 스트림을 일시 적재하는 역할을 한다. RTPStream 클래스는 순환 큐에 플레이어에 전달할 스트림이 없거나 부족할 경우 RTPDataSource 클래스의 readBuffer() 메소드를 사용하여 스트림 통신 모듈 내의 버퍼에서 수신된 스트림을 가져온다.

1.2.2 데이터베이스 연동

VoD 시스템의 서버 측에서는 VoD 시스템의 사용자 관리와 인증, 영화 정보 제공 등의 서비스를 처리하기 위해 데이터베이스를 사용하여 사용자 및

영화 정보를 관리한다. 데이터베이스 내의 회원 관리 DB는 회원등록, 회원 삭제, 사용시간 정보 반환, 회원 검색 등의 기능을 제공하고 회원 ID, 회원이름, 패스워드, 이전까지 시청한 영화목록 등의 레코드를 유지한다. 영화 정보 DB는 새로운 영화 데이터의 추가 및 삭제, 데이터 검색 등의 기능을 제공하며 제목, 장르, 등급, 국적, 영화제작일시, 주인공 등의 영화 관련 정보 레코드를 포함한다. VoD 시스템에서는 회원 관리 DB와 영화 정보 DB의 구축을 위해 MS SQL 서버 7.0을 사용하였다. 또한, 제어 및 관리 모듈 등이 회원 관리 DB와 영화 정보 DB에 접근하여 필요한 정보를 획득할 수 있도록 JDBC를 사용하였다.

1.3 QoS 피드백을 통한 동적 미디어 제어

1.3.1 QoS metric의 정의

VoD 시스템의 송신 측은 미디어 송신 후 분산 멀티미디어 통신 시스템을 통해 수신 측으로부터 전달되는 QoS 정보를 수신한다. 피드백되는 정보는 패킷의 에러, 전송지연 등의 정보가 포함되는데, 본 시스템은 지터를 기준으로 코덱의 인코딩율을 동적으로 제어하였다. (표 2)는 코덱 인코딩율 제어를 위한 알고리즘을 예시한다.

표 2. 코덱 인코딩율 제어 알고리즘

$$J_{avg(t)} = \left[\frac{\sum_{i=0}^{R_{cnt}} J_i}{R_{cnt}} \right]_t \qquad J_{avg} = \frac{\sum_{i=0}^{(t-1)} J_{avg(i)}}{n}$$

```
if( | J_avg(t) − J_avg | > α)
    if( J_avg(t) − J_avg > 0)
        if( ( Encoding_rate + θ) ≤ MAX)
            Encoding_rate += θ
        else
            if( ( Encoding_rate − θ) ≥ MIN)
                Encoding_rate −= θ
```

시간 t 구간의 RR 보고에 의한 지터를 J_i라고 하고 t 구간 동안 도착한 RR 보고의 횟수를 R_{cnt}라고 할 때, t 구간의 평균 지터 $J_{avg(t)}$는 t 구간 동안 수신한 J_i의 합을 R_{cnt}로 나눈 값이라고 할 수 있다. t - 1 구간 까지의 전체 구간의 평균 지터 J_{avg}는 첫 구간으로부터 t - 1 구간까지의 각 구간의 평균 지터 $J_{avg(i)}$를 누적한 합을 구간 수 n으로 나누면 구해질 수 있다.

현재 구간과 전체 구간의 평균 지터의 차 ($|J_{avg(t)} - J_{avg}|$)는 현재 구간의 지터 변화를 의미하는데 지터 변화가 임계값 α를 넘으면 코덱의 인코딩율을 조정한다. 즉, 현재 구간의 평균 지터가 전체 구간의 평균 지터 보다 크면 코덱의 인코딩율을 임계값 θ만큼 증가시키고 작으면 θ만큼 감소시킨다. 이 때 임계값 α와 θ는 네트워크 상태를 고려하여 유동적으로 할당될 수 있고 $Encoding_{rate}$는 인코딩율의 최대값, 최소값 사이에서만 조정 가능하다.

2. QoS 기반 성능 측정

멀티미디어 스트림의 전송, 서버 측의 코덱에 의한 미디어 인코딩 및 클라이언트 측의 미디어 재생 등에 수반되는 QoS 요소들의 변화를 모니터링하여 분산 멀티미디어 통신 시스템과 VoD 시스템에 대한 동작을 검증하고 성능을 측정한다.

2.1 성능 측정 모델

VoD 시스템의 성능 측정을 위하여 분산 멀티미디어 통신 시스템, 코덱 및 멀티미디어 플레이어 등을 연동한 클라이언트 측과 분산 멀티미디어 통신 시스템, 코덱, CORBA ORB 및 DBMS 등을 연동한 서버 측의 VoD 시스템을 인터넷에 연결된 Windows 2000 운영체제 환경의 PC 두 대에 각각 설치하였다.

동일한 조건에서 코덱의 인코딩율/디코딩율의 변화에 대한 QoS 요소들의 변화를 관찰하기 위해 동일한 클라이언트와 서버 환경에서 DBMS에 의해 관리되는 동일한 AVI 형식의 VoD 파일을 대상으로 인터넷을 통한 원격 VoD 서비스를 동시에 3회 실시하였다. 각각의 VoD 시스템들은 0.0에서 1.0 범위의 값을 가지는 인코딩/디코딩 quality를 0.5로 하여 동일한 조건에서 실행을 시작하고 10분이 경과한 후 인코딩 quality를 각각 0.0, 0.5, 1.0으로 조정한다. 성능 측정 대상 QoS 요소들은 클라이언트 측에 해상도, 비트율, 프레임율 및 프레임 지연 등과 서버 측에 지터 및 패킷 손실율 등이다.

2.2 측정 결과

2.2.1 비트율의 변화

(그림 7)은 인코딩 quality의 변화에 대한 비트율 변화의 결과를 예시한다. 10분 경까지 인코딩 quality 값이 동일했던 VoD 시스템들은 900kbps 정도의 유사한 비트율을 유지했다. 10분 경 이후부터 quality 값이 1.0으로 조정된 VoD 시스템은

1800kbps 부근까지 비트율이 상승하였으며, 0.0으로 조정된 VoD 시스템은 300kbps 부근까지 하락하였다. 즉, 인코딩 quality가 상승하여 플레이어의 해상도가 개선될수록 단위 시간에 전송되는 비트의 양은 증가함을 알 수 있다.

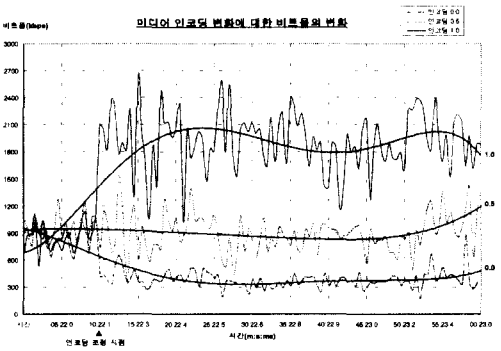


그림 7. 미디어 인코딩 변화에 대한 비트율의 변화

2.2.2 프레임율의 변화

(그림 8)은 인코딩 quality의 변화에 대한 프레임율 변화의 결과를 예시한다. 각각의 VoD 시스템들은 비트율과 마찬가지로 10분 경까지 유사한 프레임율을 유지했다. 10분 경 이후부터 quality 값이 0.0으로 하락한 VoD 시스템과 0.5를 유지한 VoD 시스템은 인코딩 quality의 변화와 관계없이 6.5fps 부근에서 프레임율을 유지하였으나, quality 값이 1.0으로 상승한 VoD 시스템은 1.7fps 부근까지 프레임율이 급격히 하락하였다. 즉, 프레임율은 일정 수준의 인코딩 quality 상승에는 영향을 받지 않지만 대역폭 및 CPU의 처리 능력 등에 의한 한계 quality에 도달하면 그 이상의 quality 값의 상승에는 반비례함을 알 수 있다. 또한, 인코딩 quality가

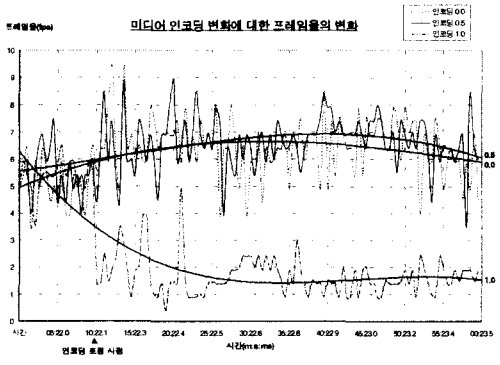


그림 8. 미디어 인코딩 변화에 대한 프레임율의 변화

1.0으로 상승했던 VoD 시스템은 프레임율 저하로 인해 다른 두 VoD 시스템들 보다 플레이어에서의 프레임 재생 지연을 유발하여 재생 시간을 증가시켰다.

2.2.3 지터의 변화

(그림 9)은 인코딩 quality의 변화에 대한 지터 변화의 결과를 예시한다. 각각의 VoD 시스템들은 다른 QoS 요소들과 마찬가지로 10분 경까지 유사한 지터를 유지했다. 10분 경 이후부터 quality 값이 0.0으로 조정된 VoD 시스템은 1000 부근까지 지터가 상승하였으며, 1.0으로 조정된 VoD 시스템은 300 부근까지 하락하였다. 또한, 0.0으로 조정된 VoD 시스템이 1.0으로 조정된 VoD 시스템 보다 지터가 더 큰 폭으로 변화하였다. 따라서, 인코딩 quality가 감소할수록 지터는 증가하고 지터의 최대값과 최소 값의 간격도 커짐을 알 수 있다.

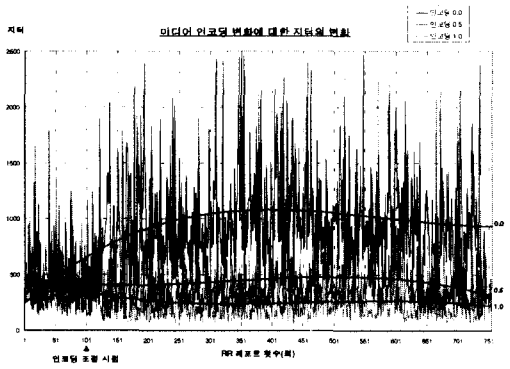


그림 9. 미디어 인코딩 변화에 대한 지터의 변화

실험 결과, 인코딩율과 QoS 요소들의 관계에서 비트율은 인코딩율에 비례하였고, 지터는 인코딩율에 반비례하였으며, 프레임율은 인코딩율에 조건부로 반비례함을 알 수 있었다. 결과적으로 각 QoS 요소들간에도 비트율과 지터는 반비례하고 프레임율과 지터는 조건부 비례하며 비트율과 프레임율은 조건부 반비례함을 알 수 있다.

2.2.4 QoS 피드백에 의한 인코딩율의 동적 제어

(그림 10)은 지터 피드백에 대하여 (표 2) 알고리즘에 의거 인코딩율을 동적으로 조정한 경우와 고정한 경우의 송신 측의 프레임율 변화를 예시한 것이다. 두 경우에 대하여 실험은 10분 경까지 특별한 부하 없이 진행되다가 10분 경에서 19분 경까지 컴퓨팅 및 네트워크 부하를 가한 후 19분 경 이후에는 가해진 부하를 해제하도록 설정되었다.

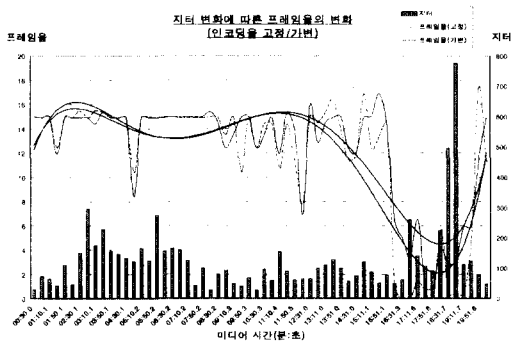


그림 10. 지터 변화에 따른 프레임율의 변화
(인코딩율 고정/가변)

네트워크 및 컴퓨팅 부하에 대하여 인코딩율을 고정한 경우는 최소 0 프레임까지 프레임율이 감소하였으나 인코딩율을 동적으로 조정한 경우는 고정한 경우 보다 상대적으로 적은 폭의 감소를 보임을 알 수 있다. 네트워크 및 컴퓨팅 부하가 발생한 후 지터는 현저한 증가 추세를 보였는데 인코딩율을 동적으로 조정한 경우에는 인코딩율을 조정하여 프레임율의 감소를 방지할 수 있었다. 가시적인 미디어 실행에서도 프레임율의 감소 폭이 적었던 경우에는 일부 끊김 현상이 발생하였지만 동영상의 플레이는 계속된 반면 인코딩율을 고정한 경우에는 플레이가 일시 정지되기도 하였다.

동적으로 변화하는 네트워크 상태에 유동적으로 대처할 수 있는 원격 멀티미디어 서비스의 제공을 위하여 분산 멀티미디어 통신 시스템 및 VoD 시스템은 비트율, 프레임율, 지터 등의 QoS 정보를 실시간으로 모니터링하고 대역폭 및 CPU 등의 자원 감소로 인한 프레임율 등의 저하가 발생할 경우 미디어의 인코딩율을 동적으로 제어하여 미디어의 끊김 현상을 예방하였다. 또한 사용자가 QoS 요소들의 우선 순위를 지정하게 함으로서 거래 관계(trade-off)에 있는 QoS 요소들간의 우선 제어를 보장하였다. 따라서 분산 멀티미디어 통신 시스템 및 VoD 시스템은 이러한 동적 QoS 제어를 바탕으로 주어진 환경에 대하여 효율적인 사용자 중심의 실시간 QoS를 제공한다.

V. 결 론

네트워크를 통한 멀티미디어 스트림의 실시간 전송, 분산 환경에서 멀티미디어 스트림의 제어 및 관리, 멀티미디어 응용간의 상호운용성 등을 지원하기 위하여 분산 객체 기술과 멀티미디어 기술의 연동이 시도되고 있다.

본 논문에서는 멀티미디어 스트림의 실시간 전송, 스트림의 제어 및 관리와 멀티미디어 응용의 상호운용성을 지원할 수 있는 CORBA 기반의 분산 멀티미디어 통신 시스템의 구조와 기능을 소개하고, 분산 멀티미디어 통신 시스템과 VoD 시스템과의 연동을 통하여 분산 멀티미디어 통신 시스템의 동작원리와 QoS 측정 결과를 제시하였다.

분산 멀티미디어 통신 시스템은 멀티미디어 스트림의 실시간 전송은 물론 이질적인 환경에서 분산 멀티미디어 응용의 개발을 용이하게 하고 응용간의 상호운용을 확대할 수 있는 플랫폼을 제공한다. 또한, QoS 정보에 따른 스트림의 제어 및 관리 기능을 통해 주어진 환경에서 사용자에게 효율적인 멀티미디어 서비스를 제공할 수 있다.

현재까지 네트워크 장비의 발전 및 네트워크 인프라의 확장이 급속도로 진행되어 있으나 원활한 분산 멀티미디어 서비스를 위해서는 전송 속도 및 관리 측면에서 많은 개선을 필요로 한다. 특히, 이동 네트워크 플랫폼에서는 이동 네트워크가 갖는 특성에 의해 많은 문제점이 산재한 것이 사실이다. 따라서, 본 논문에서 소개한 분산 멀티미디어 통신 시스템은 이러한 요구를 만족시키기 위해 다양한 환경을 고려한 개선을 지속할 예정에 있다.

참 고 문 헌

[1] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, *IETF*, RFC 1889, Jan. 1996.

[2] H. Schulzrinne, *RTP Profile for Audio and Video Conferences with Minimal Control*, *IETF*, RFC 1890, Jan. 1996.

[3] M. Mauve, V. Hilt, C. Kuhmunch, and W. Effelsberg, "RTP/I-toward a common application level protocol for distributed interactive media," *IEEE Transactions on Multimedia*, Vol. 3, March 2001.

[4] R. Braden, L. Zhang, S. Berson, and S.Herzog, *Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*, *IETF*, RFC 2205, Sep. 1997.

[5] A. Benslimane, "Real-time multimedia services over Internet," *ECUMN 2000. 1st European Conference on Universal Multiservice Networks*, 2000.

[6] OMG, *The Common Object Request Broker: Architecture and Specification Revision 2.2*, *OMG*, Feb. 1998.

[7] T. H. Yun, J. Y. Kong, and J. W. Hong, "A CORBA-Based Distributed Multimedia System," *Proc. of the Fourth Pacific Workshop on Distributed Multimedia Systems*, Vancouver, Canada, pp. 1-8, Jul. 1997.

[8] D. Le Tien, O. Villin, and C. Bac, "Resource managers for QoS in CORBA," *ISORC '99 Proceedings. 2nd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, 1999.

[9] J. C. Guerri, A. Pajares, M. Esteve, C. Palau, and A. Leon, "A feedback packet-level error control for real-time applications in wireless networks," *1999. Vehicular Technology Conference*, Vol. 2, 1999.

[10] Changwoo Jee and K. G. A. Shin, "DAVIC Video-on-Demand system based on the RTSP," *2001. Symposium on Applications and the Internet*, 2001.

[11] J. Meggers and R. Subramaniam, "A new feedback error control scheme for block based video communication in packet switched wireless networks," *1999. Proceedings. IEEE International Symposium on Computers and Communications*, 1999.

[12] 박상현, 박상윤, 김명준, 엄영익, "응용 독립적인 RTP 통신 모듈의 설계 및 구현," 정보처리논문지, *한국정보처리학회*, Vol. 6, No. 9, Sep. 1999.

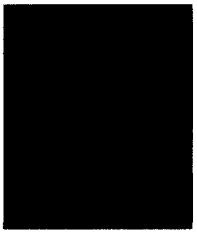
[13] G. Fortino and L. Nigro, "A cooperative playback system for on-demand multimedia sessions over Internet," *ICME '2000 IEEE International Conference on Multimedia and Expo*, Vol. 1, 2000.

[14] Myung-Ki Shin and Jin-Ho Hahm, "Applying QoS guaranteed multicast audio and video to the Web," *1999. IEEE International Conference on Multimedia Computing and Systems*, Vol. 2, 1999.

박 상 윤(Sang Yun Park) 정회원

1997년 2월 : 동국대학교 전자계산학과 학사
1999년 2월 : 성균관대학교 전기전자 및 컴퓨터공학부 석사
1999년 3월 ~ 현재 : 성균관대학교 전기전자 및 컴퓨터공학부 박사과정
<주관심 분야> 이동 컴퓨팅 시스템, 멀티미디어 통신, 이동 컴퓨팅 보안, 분산 시스템

정 길 호(Gil Ho Joung)



2000년 2월 : 상명대학교 정보통신학부 학사
2000년 3월 ~ 현재 : 성균관대학교 전기전자 및 컴퓨터공학부 석사과정, 마크애니 부설 연구소 연구원
<주관심 분야> XML, 이동 컴퓨팅 시스템, 전자상거래

박 상 현(Sang-hyun Park)

1998년 2월 : 성균관대학교 전기전자 및 컴퓨터공학부 학사
2000년 2월 : 성균관대학교 전기전자 및 컴퓨터공학부 석사
2000년 2월 ~ 현재 : 현대정보기술 연구원
<주관심 분야> 분산 시스템, 멀티미디어

엄 영 익(Young Ik Eom)

1983년 2월 : 서울대학교 계산통계학과 학사
1985년 2월 : 서울대학교대학원 전산과학전공 석사
1991년 8월 : 서울대학교대학원 전산과학전공 박사
현재 : 성균관대학교 전기전자 및 컴퓨터공학부 교수
<주관심 분야> 분산 시스템, 이동 컴퓨팅 시스템, 분산 객체 시스템