

QoS와 재전송을 지원하는 MPEG-4 미디어 재생기의 설계 및 구현

전형국⁰ 마평수
 한국전자통신연구원 정보가전연구부
 hkjun@etri.re.kr pmah@etri.re.kr

Design and Implementation of MPEG-4 Media Player supporting QoS and Retransmission

Hyung-Kook Jun⁰ Pyeong-Soo Mah
 Dept. of Information Appliance, Electronics and Telecommunications Research Institute

요 약

본 논문에서 구현한 MPEG-4 미디어 재생기는 네트워크를 통하여 스트리밍 서버에 메시지를 전송하고, 스트리밍 서버로부터 전송된 미디어 스트림을 처리하여 사용자의 화면에 보여주는 역할을 수행한다. 미디어 재생기는 서버에게 VCR(Play, Pause, Stop, Jump) 이벤트를 전송하며, 리눅스 운영체제에서 동작한다.

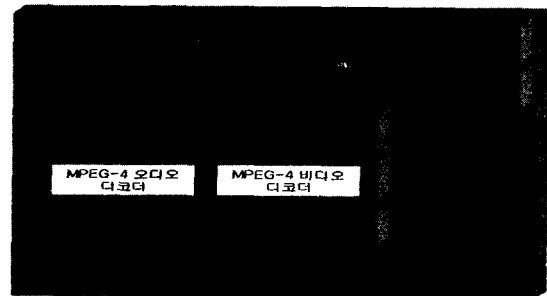
본 논문에서 제안하는 미디어 재생기는 VCR 연산을 위한 RTSP(Real Time Streaming Protocol)를 지원하며 데이터의 실시간 전송을 위하여 RTP(Real-time Transport Protocol)/RTCP(real-time transport control protocol)를 지원한다. RTP/RTCP를 통하여 네트워크 QoS와 관련된 정보를 획득하며 그 정보에 따라서 데이터 전송율을 조절하고 RTCP를 이용하여 에러 패킷에 대한 재전송을 사용한다. 재전송시 미디어 재생기의 버퍼링 상태와 멀티미디어 데이터의 재생시간을 고려하여 멀티미디어 데이터의 끊김이 없도록 한다.

1. 서 론

본 논문에서 제안하는 미디어 재생기는 VCR 연산을 위한 RTSP(Real Time Streaming Protocol)를 지원하며 데이터의 실시간 전송을 위하여 RTP(Real-time Transport Protocol)/RTCP(real-time transport control protocol)를 지원한다. RTP/RTCP를 통하여 네트워크 QoS와 관련된 정보를 획득하며 그 정보에 따라서 데이터 전송율을 조절하고, RTCP를 이용하여 에러 패킷에 대한 재전송을 사용한다. 재전송시 미디어 재생기의 버퍼링 상태와 멀티미디어 데이터의 재생시간을 고려하여 멀티미디어 데이터의 끊김이 없도록 한다.

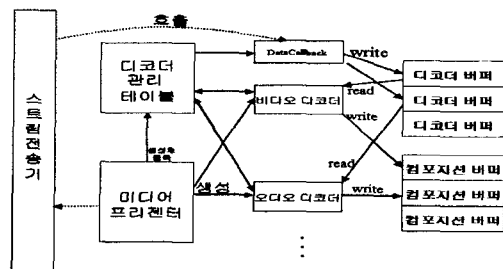
2. MPEG-4 미디어 재생기

미디어 재생기의 모듈구조는 서버로부터 MPEG-4 미디어 스트림을 전송받아 오디오/비디오 디코더 버퍼에 전달하는 스트림 전송기, 각 스트림을 디코딩하는 MPEG-4 오디오/비디오 디코더, 디코딩된 미디어 스트림을 GTK+ 그래픽 APIs를 이용하여 사용자 인터페이스에 보이는 MPEG-4 오디오/비디오 프리젠타가 있다. 또한 스트림 전송기와 디코더 사이의 스트림을 처리하기 위한 디코더 버퍼, 디코더와 프리젠타 사이에서 MPEG-4 스트림을 처리하기 위한 컴포지션 버퍼가 있다. 디코더 버퍼와 컴포지션 버퍼에서 스트림을 주고받기 위한 API가 존재하며, 미디어 재생기는 사용자 인터페이스에서 이벤트가 발생하면 이를 처리하기 위한 이벤트 처리기를 지원한다. 이는 스트림 전송기와 미리 정의된 서버통신 메시지를 이용하여 통신한다.



<그림 1> 미디어 재생기 구조

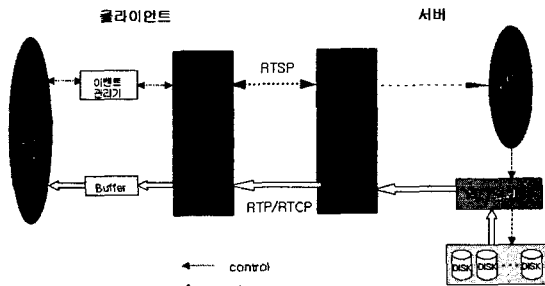
미디어 재생기가 각 디코더를 수행 시키면, 스트림 전송기에서 DataCallback을 통해 수신된 미디어 데이터를 디코더 버퍼에 저장하게 된다. 디코더는 버퍼 관리 API를 통해 해당 디코더 버퍼의 데이터를 가져온다. 이를 디코딩 해 버퍼 관리 API를 통해 컴포지션 버퍼에 디코딩 된 데이터를 저장한다. 프리젠타는 컴포지션 버퍼에 있는 데이터를 이용해 재생한다.



343 <그림 2> 디코더와 미디어 프리젠타와의 동작

3. 스트림 전송기

미디어 서버와 재생기 사이의 서비스를 제공하기 위해 설정되는 세션 관리를 제공한다. 미디어 재생기에서 요구하는 객체는 데이터 전송 채널을 통하여 전송되고 전송이 종료되면 채널 관리에 의해 데이터 전송 채널을 제거한다. 스트림 전송기에서 비디오 정보, 오디오 정보와 같은 스트림 정보를 얻는다. 스트림 전송기는 미디어 재생기의 이벤트 관리기로부터 받은 제어를 제어 전송용 네트워크 채널을 이용하여 스트리밍 서버에 전송하고, 데이터 전송용 네트워크 채널을 통해 스트리밍 서버로부터 미디어 재생기의 버퍼에 데이터를 전달하게 된다. 제어 전송은 신뢰성을 보장해야 하므로 TCP를 통해 이루어지며 RTSP를 사용하고 데이터 전송은 UDP를 통해 이루어지며 RTP를 사용한다.

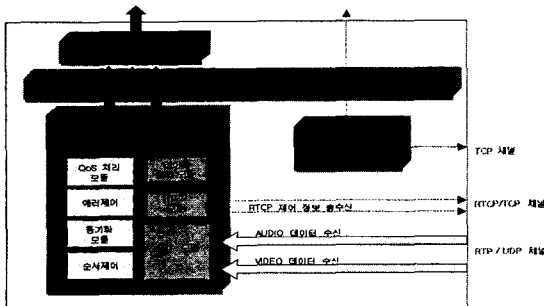


<그림 3> 스트림 전송기 구조도

4. 클라이언트측 스트림 전송기

클라이언트 측 스트림 전송기는 미디어 재생기로부터 스트리밍 서버로의 요청을 전달하고 서버로부터의 데이터를 버퍼에 저장한다. 또한 미디어 데이터에 대한 동기화, 순서제어, 에러제어, QoS 정보를 처리하는 모듈로 구성되어 있어 이에 해당되는 내용을 처리한다.

클라이언트에 필요한 채널은 메시지 전송을 위한 RTSP 채널과 음성, 영상으로 나뉜 각각의 데이터 전송채널, 이에 해당하는 제어정보 채널인 RTCP채널로 이루어져 총 5개의 채널을 설정하게 된다. 데이터 전송은 RTP모듈을 통해 UDP를 통하여 전송되며 제어정보 전송은 RTCP모듈을 통해 TCP전송을 하게 된다.



<그림 4> 클라이언트 스트림 전송기 블록 구조도

- RTSP interface: 사용자의 요청에 의하여 서버에게

RTSP제어메시지를 송신하는 모듈이며, 제어 메시지 송수신 모듈을 통해서 수신된 RTSP 메시지들을 처리하는 모듈로 세션 설정과 처리가 수행되며, 미디어에 대한 VCR 처리 기능을 제공한다.

- 제어메시지 송수신 모듈 : 응용계층에서 요청한 RTSP 메시지를 TCP채널을 통해 전송하며 서버로부터의 메시지를 응용으로 전달해주는 역할을 담당한다.
- 데이터 수신 모듈 : 서버로부터 전송되어 온 데이터를 수신하여 오디오, 비디오 버퍼에 저장하는 기능을 수행한다.
- 순서제어 모듈 : RTP헤더내의 sequence number를 보고 순서가 맞게 패킷이 도착한 경우 응용 계층으로 보내고, 일정 시간 내에 도착하지 않은 패킷의 경우 이를 손실된 것으로 보고 에러제어 모듈을 호출한다.
- 에러제어 모듈 : RTP를 통해서 전달된 데이터 패킷의 에러를 분석하고 잃어버린 패킷에 대하여 RTCP 메시지를 통해서 재 전송을 요구한다. 에러제어는 순서제어 모듈과 연계하여 수신 버퍼를 관리한다.
- 동기화 모듈 : RTP의 timestamp를 통해 동기화에 관련된 정보를 추출하여 오디오와 비디오 데이터의 동기를 맞추어 응용층에 넘겨준다.
- QoS 처리 모듈 : RTP/RTCP모듈을 통해 얻어진 QoS 정보를 응용계층이 요구하는 경우 이를 처리해 주는 기능을 담당한다. QoS 정보에 관련된 정보는 RTCP 패킷을 통해 얻는다.
- RTP/RTCP 처리 모듈 : 전송받은 데이터의 RTP 헤더를 분리하여 순서제어에 해당하는 Sequence number를 얻어 순서제어 및 에러제어 모듈에서 처리하도록 하며, 동기화를 위한 timestamp 또한 동기화 모듈로 값을 넘겨준다. 마찬가지로 RTCP헤더를 분리하여 QoS 정보를 처리하기 위한 값을 얻는다.
- 데이터 처리 모듈 : 전송된 데이터의 채널 ID를 통해 각각의 저장버퍼로 저장한다.

4.1 RTSP 인터페이스

요청한 측에서는 RTSP 메시지를 보내고 요청받은 측에서는 RTSP OK 메시지를 통해 그 결과를 보낸다. RTSP 메시지는 request와 매핑되고RTSP OK는Response 메시지와 매핑될 수 있다. Request에 사용되는 메소드와 ID값은 <표 1>과 같고, 각 구현에 따라 메시지 형태로 제공될 수 있다.

<표 1> 스트림 전송기 간의 메시지의 messageId

Message ^o	messageId ^o
RTSP_Play ^o	0x10 ^o
RTSP_Pause ^o	0x11 ^o
RTSP_Stop ^o	0x20 ^o
RTSP_Jump ^o	0x21 ^o
RTSP_Teardown ^o	0x30 ^o

다음은 미디어 재생기가 스트리밍 서버로의 요청을 스트림 전송기를 통해 전달할 때의 인터페이스를 정의한다.

- int setupRTSPStreams() : 서비스 세션 시작을 요청하기 위해 미디어 재생기에 의해 호출되는 것으로서, 미디어 재생기와 서버간의 제어와 데이터 전송을 위해 TCP/UDP 소켓을 생성함으로써 세션을 설정하고 소켓으로부터 데이터를 읽어 들이는 태스크를 생성한다. 또한 각 비디오 오디오에 대한 채널을 설정한다. URL은 서버측 스트림 전송기의 주소와 서비스 이름을 포함하고 있다. 클라이언트 측 전송. 관리자는 서버로부터의 response가 정상이라면 해당 세션에 대해 sessionId를 할당하여 미디어 재생기에 전달한다.
- int TeardownRTSPStream() : 현재 설정된 세션의 종료를 요청하기 위해 호출된다. sessionId는 종료를 요청하고자 하는 세션의 sessionId이다.
- int startPlayingRTSPStream() : 설정된 세션 정보를 바탕으로 미디어 스트림에 대한 재생 요청을 수행한다. 세션 ID는 제어를 제공하기 위한 세션 정보가 된다.
- int startMETHODstream() : 미디어에 대한 제어 정보를 전송하기 위한 함수이다. 현재PAUSE, STOP, JUMP METHOD를 전송하기 위한 함수로 사용된다.

4.2 RTP/RTCP 인터페이스

RTP/RTCP는 시간 정보와 각종 QoS정보를 제공하여 실시간 데이터를 전송하기 위한 프로토콜이며 전송된 데이터의 구별능력과 패킷의 순서에 대한 결정, 다중 미디어에 대한 동기화 기능, 패킷 분실에 대한 에러 처리를 수행한다. 현재 본 논문에서 구현한 시스템에서 구현한 RTP 인터페이스로는 RTP 헤더 패킷 생성, RTP 세션 초기화 및 종료, 패킷 전송을 위한 소켓 개설, RTP 데이터 전송로 구성되어 있다. 또한 RTCP 인터페이스로는 RTCP 헤더 패킷 생성 및 정보 생성, RTCP 제어정보 처리, QoS 처리, 재전송 관련 인터페이스등을 정의하고 있다. 다음은 본 논문에서 구현한 QoS 관련 인터페이스를 정의한다.

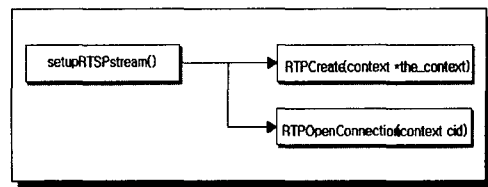
- Loss_rate(Loss_packet_number, Sender_packet_number) 패킷의 손실율을 나타내는데 사용된다.
- Sender_Count_packet(send_packet) : 송신측에서 Count한 패킷의 수를 나타낸다.
- Receive_Count_packet(receive_packet) : RTP에서 수신된 수신측의 패킷의 수를 받는다.
- Check_Interval_Jitter() : 송신자로부터 보내어지는 패킷의 수신 도착시간의 변위를 측정하여 RR에 실어 보낸다.
- Check_RR_receive_time() : 송신자에게 RR이 도착한 시간이다.
- Delay_Account() : RR이 송신자에게 수신된 시간, 즉 Check_RR_receive_time()에서 반환된 값에서 RR의 Last_Time_SR()의 값과 Delay_LTS()의 값을 뺀 값이다.

다음은 본 논문에서 구현한 재전송과 관련된 인터페이스를 정의한다.

- Not_orderRetrieveddata() : not-ordered된 데이터 중 요청하는 데이터가 있는지 확인하는 함수

- Not_orderArrivaldata() : not-ordered된 데이터를 정렬하기 위한 함수이다.
- check_data(retrieve_data, id) : 현재 수신된 데이터가 순서적으로 요청하는 데이터인지 검증하는데 사용된다.
- Not_orderArrivaldata()호출
- updata_loss(ave_loss, loss) : 손실을 계산
- updata_lat(ave_lat, lat) : latency 계산
- updata_rtt(ave_rtt, rtt) : round trip 계산
- difftime(cur_time, arrive_time) : 현재 시간과 패킷의 수신 시간 계산
- wait_rto(max_time) : retransmission time out을 계산하는 함수이다.

RTSP 함수인 setupRTSPStreams()은 서비스 연결을 요청함으로써 세션 확립을 하며 내부적으로는 RTP 세션을 열어주기 위한 RTPCreate 함수를 호출해야 하고, RTP/RTCP 소켓을 개설하여 세션 셋업이 완료된다.



<그림 5> 서비스 연결 요청

RTSP_teardown()함수는 서비스를 종료시키는 함수로서, RTPCloseConnection() 을 호출하여 개설된 RTP/RTCP 소켓을 닫고 RTPDestroy()를 통해 남아있는 세션관련 정보들의 자원할당을 거둬들인다.

6. 결론 및 향후 연구 과제

본 논문에서 제안하는 미디어 재생기는 RTP/RTCP를 통하여 네트워크 QoS와 관련된 정보를 획득하며 그 정보에 따라서 데이터 전송율을 조절하고, RTCP를 이용하여 에러 패킷에 대한 재전송을 사용한다. 재전송시 미디어 재생기의 버퍼링 상태와 멀티미디어 데이터의 재생시간을 고려하여 멀티미디어 데이터의 끊김이 없도록 하였다.

앞으로의 연구과제로는 Scalable과 관련된 데이터 송수신 처리를 수행하고 현재 제공되지 못하고 있는 QoS에 따라 중요 비디오(I 프레임에 대한)에 대한 선택적 재전송을 구현하려고 한다.

7.참고 문헌

- [1] " RTP Payload Format for MPEG-4 Audio/Visual Stream", rfc3016, Internet Engineering Task Force, November 2000
- [2] " RTP:A Transport Protocol for Real-Time Applications", rfc1889, Internet Engineering Task Force, January 1996
- [3] " Real Time Streaming Protocol", rfc 2326, Internet Engineering Task Force, November 1998