멀티미디어 스트리밍 프레임워크에서 세션 관리자의 설계 및 구현

임익진° 이승룡
경희대학교 전자계산공학과
{limga, sylee}@oslab.kyunghee.ac.kr

Design and Implementation of Session Manager in Multimedia Streaming Framework

Eakjin Lim° Sungyoung Lee
Dept. of Computer Engineering, Kyung Hee University

요 약

본 논문에서는 통합 멀티미디어 스트리밍 프레임워크(ISSA)의 주요 모듈인 전송 및 세션 관리자에 대한 개발 경험을 소개한다. ISSA(Integrated Streaming Service Architectur)[1][2]는 유니캐스팅/멀티캐스팅 환경의 VOD 시스템과 실시간 방송시스템(라이브캐스팅)과 같은 통합 멀티미디어 스트리밍 서비스 응용을 개발하기 위한 스트리밍 프레임워크이며 RTP(Real-Time Protocol)/RTCP(Real-Time Control Protocol)[3][4], RTSP 등의 표준 실시간 전송 프로토콜을 사용함으로써 사용자에게 범용성을 제공한다. ISSA는 다양한 형태의 미디어를 지원하며, 이기종 운영체제와 네트워크에 독립적이며, 실시간 멀티미디어 데이터베이스와 연동하여 사용자에게 데이터베이스 서비스를 제공할 수 있다. 세션 관리자는 미디어 채널의 생성과 제어를 담당하는 기능과 멀티미디어 데이터베이스를 위한 트랜잭션 전송기능을 담당하는 기능을 제공하며 각각 RTSP와 RTTP 프로토콜을 이용하여 구현되었다.

1. 서론

일반적으로 스트리밍 환경에서의 세션 관리자는 스트리밍 응용 프로그램 사이의 세션을 새로 생성 또는 소멸하거나 재생 그리고 멈춤 등과 같은 기능을 제공함으로써 스트리밍되는 미디어를 제어할 수 있는 기능을 제공한다. ISSA의 세션 관리자 역시 세션 제어 기능을 주된 목표로 설계되었으며 이러한 기능은 RTSP(Real-Time Streaming Protocol)[5]을 이용하여 구현되었다. 또, RTSP의 스트리밍은 데이터 베이스와의 연동을 고려하지 않았기 때문에 멀티미디어 데이터 베이스인 BeeHive의 트랜잭션을 원격 호스트에서 처리하기 위해 RTTP (Real-Time Transaction Protocol)를 설계하고 구현하였으며, 원격 호스트에서의 스트리밍 서버 관리를 위한 TCP 포트를 통한 문자열 기반의 통신 프로토콜인 SCP(Streaming Control Protocol)[7]등을 지원한다. 그리고, 스트리밍 서버를 웹과 연동하기 위해 웹 엔진을 내장 할 수 있으며 이것을 통해 HTTP 프로토콜[8] 처리도 가능하다. 실제적인 스트리밍을 담당하는 전송 관리자의 프로토콜과는 별도로 동작하며 자체 제작한 Network Interface를 통해 플랫폼 및 네트워크 하위단에서의 독립성을 보장받고 있다.

본 논문에서는 ISSA의 주요 모듈인 세션 관리자의 설계와 구현에 대한 개발 경험을 소개한다. 2장에서는 RTSP를 사용한 관련 연구에 대해서 살펴보고, 3장에서는 세션 관리자의 모듈에 대해 소개되며, 4장에서는 세션 관리자에서 여러 서비스를 지원하기 위해 추가된 프로토콜과 기능등에 대해 알아보겠다. 5장은 구현 사례를 소개하며, 6장에서 결론을 내린다.

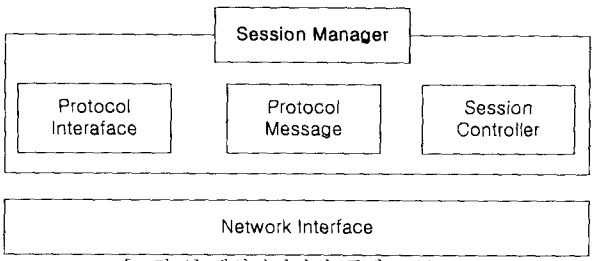
2. 관련 연구

Entera의 TeraCAST서버[9]는 상용 서버로서 1999년에 개발되었다. RTP/RTSP 상에서 대용량의 미디어 시장을 목표로 개발되었으며 서버는 유닉스 계열과 윈도우를 지원하며 Mac 용은 현재 개발 중이다. 클라이언트는 JMF 2.0과 애플의 Quick Time Player를 지원하며 미디어 수용성에 아직 한계를 가지고 있으며, 데이터 베이스와의 연동 기능은 아직 제공되지 않고 있다.

오픈 소스 Foundation에서 오픈 소스 프로젝트의 일환으로 제작중인 Free Expression 프로젝트[10]는 다양한 전송 프로토콜 위에 RTSP 프로토콜을 사용할 목적으로 개발이 진행중인 스트리밍 서비스 프로젝트이다. 1999년 초에 프로젝트를 시작하였지만 아직 개발 성과가 나오지 않은 상황으로 여러 스트리밍 서버의 역공학(Reverse-Engineering) 작업이 진행중이다. 하지만, Free Expression 프로젝트가 현재 시작 단계라 하지만, 세계적으로 폭넓은 지지층과 사용자층을 가지고 있는 OSF에서 개발하는 스트리밍 서버라는 사실만으로도 벌써 주목을 받고 있다.

3. 세션관리자의 설계

세션 관리자는 프로토콜 인터페이스 모듈, 프로토콜 메시지, 세션 컨트롤러로 구성되어 있으며, 네트워크 하위단에서는 Network Interface 모듈을 사용하고 있다 [그림 1].



[그림 1] 세션관리자의 클래스 구조도

프로토콜 인터페이스 모듈은 세션 관리자에서 사용되는 여러 프로토콜을 사용할 때 응용단과의 인터페이스를 담당한다. 인터페이스는 각 프로토콜마다 다르게 존재하며, 서버와 클라이언트가 구별되어 구현되었다. 또, 멀티캐스트를 위한 인터페이스는 독립적인 인터페이스가 마련되었다. 프로토콜 메시지 모듈은 RTSP, RTTP, HTTP등 세션 관리자에서 사용하는 프로토콜들의 메시지가 구현된 모듈이며, 메시지 클래스는 크게 요청 메시지와 응답 메시지로 나뉘어져 있고, 텍스트 기반의 프로토콜을 중심으로 구현되었다. 세션 컨트롤러 모듈은 서버 측에서 사용될 경우 참가자의 여러 세션을 관리하는 역할을 하며, 클라이언트 측에서는 커넥션의 유지만을 담당한다.

네트워크 인터페이스 모듈은 네트워크 관련 함수를 필요로 하는 세션 관리자와 전송 관리자에서 사용되며, 현재 Windows의 Winsock 인터페이스와 Unix계열의 Berkeley Socket의 인터페이스를 지원하며, 차후 TLI(Transport Layer Interface), ATM(Asynchronous Transfer Mode) 등과 같은 인터페이스를 지원할 예정이다.

4. 세션관리자의 지원 프로토콜

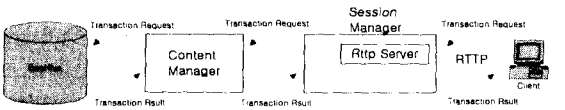
세션 관리자는 본래 세션 제어의 목적으로 구현되었으나, 개발 단계에서 변화하는 환경을 수용하기 위해 많은 서비스들을 위한 기능이 추가되었다. 이러한 서비스들은 업계 표준으로 사용하는 프로토콜을 우선하여 구현하였으며, 그렇지 못한 경우에는 자체적으로 제작된 프로토콜로 구현하였다. 세션관리자에서 사용한 표준 프로토콜은 대표적으로 RTSP, SCP등이 있으며, RTTP와 같이 Univ. of Virginia와 공동으로 제작한 프로토콜도 사용하고 있다.

- **RTSP**

RTSP는 오디오/비디오와 같이 시간적 동기화가 필요한 단수 혹은 복수 개의 연속된 미디어의 채널 설정과 제어를 위한 프로토콜로 멀티미디어 서버의 "network remote control"과 같은 기능을 제공한다. 서버는 각 세션에 식별자를 지정하여 관리하며, 연결지향형(Connection-Oriented)나 비연결형 (Connectionless)에 상관없이 동작하며 실제적인 미디어의 전송을 담당하는 전송 계층과는 독립적으로 작동한다.

- **RTTP**

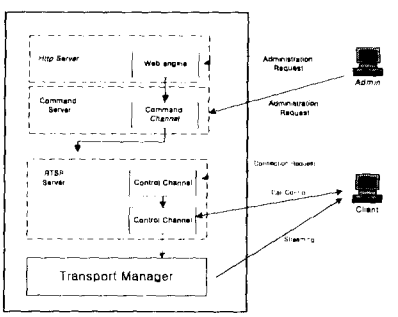
RTTP는 ISSA와 BeeHive[6]와의 연동을 위해서 제안된 프로토콜이다. 이는 RTSP와 유사하게 구현되었고 트랜잭션 요청의 전송 역할만을 담당하며, 실제적인 트랜잭션은 컨텐츠 관리자에서 데이터베이스에 요청 시 이루어지게 된다. [그림 2]는 BeeHive와의 연동 시 트랜잭션의 흐름을 보여준다. 클라이언트는 RTTP를 통해 세션 관리자에 요청하는 트랜잭션을 보내게 되면 세션 관리자는 이를 해석하여 컨텐츠 관리자에 통보하게 된다. 이렇게 전해진 데이터베이스의 결과는 다시 컨텐츠 관리자에서 세션 관리자로 넘어오게 되며, 다시 세션 관리자는 RTTP를 통해 클라이언트에게 결과를 보내게 된다.



[그림 2] 실시간 트랜잭션 프로토콜의 흐름

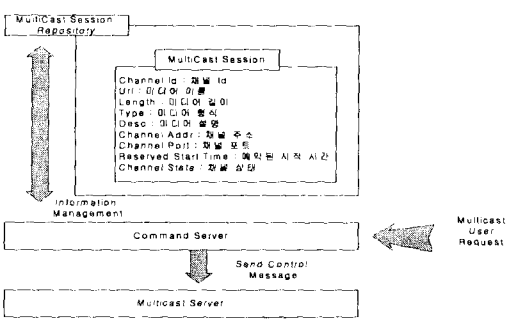
- **기타 지원 프로토콜**

명령(Command) 서버는 서버의 행정 (Administrate) 기능을 위해 개발되었으며, 기존에 개발된 모듈을 사용하여 간략하게 구현되었다. 명령 서버를 이용하여 관리자는 원격에서 스트리밍 서버를 관리하기 위한 포트를 생성하고 Telnet, Web등 TCP를 기반으로 하는 네트워크 경로 위에서 SCP를 통해 언제든지 관리를 할 수 있는 장점이 있다.[그림 3]



[그림 3] 스트리밍 서버의 관리 경로

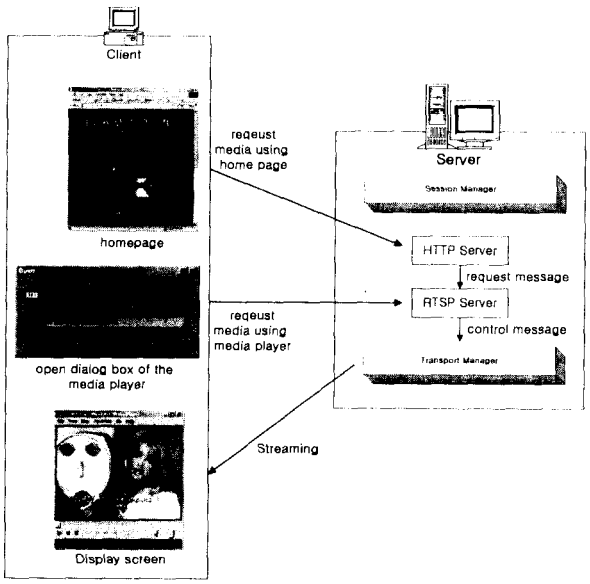
멀티캐스트 세션은 멀티캐스트 채널의 생성과 시작, 종료, 리스트, 관리 기능을 제공하며 [그림 4]에서 보듯이 명령 서버를 통해서 관리할 수 있다. 미디어 목록, 채널 생성, 채널 시작, 채널 종료의 기능은 멀티캐스트 세션 저장소(Repository)로부터 하나의 채널에 대한 정보를 얻어 오거나, 그곳에 저장할 수 있으며, 실제적인 스트리밍 관리는 멀티캐스트 서버를 통해 이루어진다. 미디어 목록에서는 현재 멀티캐스트 서버가 서비스 할 수 있는 미디어 목록을 제공해 주며, 채널 생성의 기능은 멀티캐스트 채널 주소와, 포트 번호, 예약된 시작 시간을 입력받아 멀티캐스트 세션 저장소에 저장하게 된다. 채널 생성과 채널 종료를 원하면 멀티캐스트 세션은 즉각적으로 채널을 제어한다.



[그림 4] 멀티캐스트 세션

5. 구현사례

본 논문에서 소개된 세션 관리자와 함께 ISSA의 전송 관리자, 미디어 관리자를 이용하여 오디오 및 비디오의 다양한 스트리밍 서버와 클라이언트를 제작 할 수가 있는데, 이것을 위해 클라이언트의 어플리케이션 경우 응용으로 미디어 관리자는 Windows Media Player를 위한 Source Filter와 WinAmp를 위한 Plug-in을 지원한다. 이러한 미디어 응용에서 세션 관리자가 사용되는 예를 다음 [그림 5]에서 설명하고 있다. 클라이언트는 홈페이지나 Windows Media Player를 직접 사용하여 서버에 스트리밍 요청을 보낼 수 있으며, 홈페이지를 이용할 경우는 웹 인터페이스를 이용할 수 있고, 직접 서버에 요청을 할 경우 RTSP 인터페이스를 통하여 스트리밍이 이루어진다.



[그림 26] 스트리밍 서비스에서의 세션 관리자

6. 결론

본 논문에서 언급하였듯이 ISSA의 세션관리자는 세션의 성립, 제어의 기능을 위해 업계에서 표준으로 사용되는 RTSP을 구현하였으며, 기존의 스트리밍 서비스에서 데이터 베이스의 연동 기능을 고려하지 않았기 때문에 실시간 데이터베이스의 트랜잭션 처리를 위한 RTTP를 설계 구현하였으며, SCP를 지원하는 명령 서버, 웹 인터페이스의 제공 등 응용의 하위단에서 여러 가지 경로를 통해 스트리밍 서비스를 지원한다.

하지만, 세션 관리자에서 이러한 여러 기능들을 포함하면서 발생한 세션 관리자의 다양한 역할은 세션 관리자의 성능 저하로 이어질 수 있기 때문에 이러한 기능들을 외부 모듈로 분리해 경량화를 우선적으로 해야할 것으로 보인다. 작은 용량의 세션관리자는 스트리밍 서버보다는 스트리밍 클라이언트에 탑재 될 시에 클라이언트의 이동성에 보다 유용하기 때문이다.

7. 참고 문헌

[1] 정찬균, 이승룡,""통합 스트리밍 프레임워크의 설계", 제 11회 한국 정보처리학회 춘계 학술발표 논문집, pp. 319-322, 1999년 4월.

[2] 정찬균, 김형일, 홍영래, 임익진, 이승재, 이승룡, 정병수, "분산 스트리밍 시스템 설계", 제 4회 한국 멀티미디어 학회 추계 학술발표 논문집, pp. 338-343, 1999년 11월.

[3] H. Schulzrinne, "RTP: A Transport Protocol for Real-Time Applications", IETF RFC 1889, Jan. 1996

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

[5] H. Schulzrinne, "RTSP: Real-Time Streaming Protocol", IETF RFC 1890, Apr. 1998

[6] J. Stankovic and S. H. Son, "An Architecture and Object Model for Distributed Object-Oriented Real-Time Databases," *Journal on Computer Systems Science and Engineering*, Special Issue on Object-Oriented Real-Time Distributed Systems, vol. 14, no. 4, pp 251-259, July 1999.

[7] Shanwei Cen, Calton Pu, Jonathan Walpole, "Flow and Congestion Control for Internet Media Streaming Applications", Tech Report in Oregon Institute of Science and Technology, Mar. 1997

[8] R. Field, UC. Irvine, "HyperText Transfer Protocol - HTTP 1.1", IETF RFC 2068, Jan. 1997

[9] http://streaming.entera.com/

[10] http://www.free-expression.org/