RTSP에 기반한 스트리밍 캐쉬 서버

오재학* 차호정**
*광운대학교 컴퓨터과학과
**연세대학교 컴퓨터과학과
ojh@cs.kwangwoon.ac.kr hjcha@yonsei.ac.kr

A RTSP-based Streaming Cache Server

Jaehak Oh* Hojung Cha**
*Dept. of Computer Science, Kwangwoon University
**Dept. of Computer Science, Yonsei University

요 약

본 논문은 멀티미디어 스트리밍 환경에서 RTSP 표준화 프로토콜을 기반한 스트리밍 캐쉬 서버를 설계하고 구현한다. 또한 기존의 스트리밍 시스템과 호환성을 유지하기 위해 캐쉬 서버의 On-Demand 스트리밍 서비스를 지원하는 RTSP 세션 제어 기능의 응용과 절차에 대하여 기술한다.

1. 서론

최근에 멀티미디어 스트리밍 환경의 발전으로 네트웍 상에 대용량 콘텐츠가 증가하고 있다. 이와 같은 현상은 네트웍 트래픽를 증가시켜 콘텐츠의 실시간 전송을 어렵게하는 원인이 되고 있다. 이에 대한 해결책으로 지역 네트웍에 캐쉬 서버를 위치시키는 방안이 집중적으로 연구되고 있다[1].

기존에 스트리밍 미디어 캐싱은 클라이언트 관점에서 콘텐츠를 클라이언트의 내부에 저장하는 내려받기와 실시간 전송을 위한 On-Demand 방식으로 분류할 수 있다. 내려받기 방식은 웹 캐싱의 정적인 콘텐츠의 캐싱과 동일한 방식으로 사용자는 내려받는 시간 동안 대기 상태에 있어야 하고 한정된 저장용량에 서비스의 품질을 제한 받는다. On-Demand 서비스는 클라이언트와 서버의 일대일 관계에서 서비스 가능한 대역폭 내에 임계의 서비스 거리에 있거나 서버의 정책에 따라 저품질의 서비스로의 변환을 받아야하는 한계가 있다. 이와 같은 문제에 대하여 그림 1과 같은 캐쉬 서버의 배치는 네트웍 트래픽의 균일화와 사용자 QoS의 보장을 위해 제시되고 있다. 이미 대부분의 서비스 네트웍에서는 캐쉬 서버의 배치를 네트웍 구성의 기본 원리로 적용하고 있다. 그러나 스트리밍 미디어 캐싱은 기존에 개발된 스트리밍 서비스와의 호완성과 캐쉬 프로토콜의 개발 측면에서 캐쉬 서버의 구성을 어렵게 한다[2].

본 논문은 멀티미디어 스트리밍 환경에서 RTSP[3] 프로토콜을 기반한 스트리밍 캐쉬 서버를 설계하고 구현한다. 또한 기존의 스트리밍 시스템과 호환성을 유지하기 위해 캐쉬 서버의 On-Demand 스트리밍 서비스를 지원하는 RTSP 세션 제어 기능의 응용과 절차에 대하여 기술한다. 본 논문의 구성은 2절에서 RTSP에 기반한 스트리밍 캐쉬 서버의 구성을 기술하고, 3절에서 RTSP의 세션 처리를 기술한다. 4, 5절에서 개발환경과 결론 및 향후 과제를 제시한다.

• 본 연구는 정보통신부에서 지원하는 대학기초연구지원사업으로 수행하였음 (과제번호 : 2001-076-3)

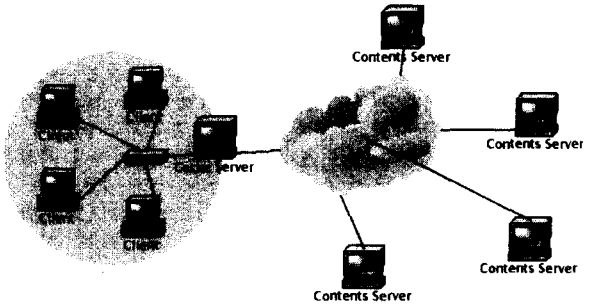

그림 1 캐쉬 서버의 개요

2. RTSP에 기반한 스트리밍 캐쉬 서버 구성

본 논문에서 제시하는 캐쉬 서버는 전방위 캐쉬 모델로 설계되었다. 전방위 캐쉬(이후 캐쉬)는 미디어를 대상으로 하는 실시간 트래픽에 대해서 네트웍 트래픽을 효율적으로 관리할 수 있는 네트웍 모델이다. 캐쉬 서버는 네트웍 부하를 감소시키는 대신 클라이언트와 서버의 양자에 연결하는 구조로 한 서비스 요구에 대해서 두 개의 세션을 유지해야하는 문제가 있다. 따라서 캐쉬 서버의 세션 증가는 시스템의 부하를 의미하므로 유연한 세션 관리 및 전송 구조로 설계해야 한다.

그림 2는 캐쉬 서버의 구조를 보여준다. 캐쉬 서버의 구조는 세션 관리자, 콘텐츠 관리자, 캐쉬 관리자, 프로토콜 제어기로 구성된다. 세션 관리자는 클라이언트의 요구를 받아 캐쉬와 콘텐츠 서버의 상태에 따라 수용 여부를 결정하고 설정된 세션의 종료 때까지 세션을 유지 관리한다. 또한 세션 관리는 기존

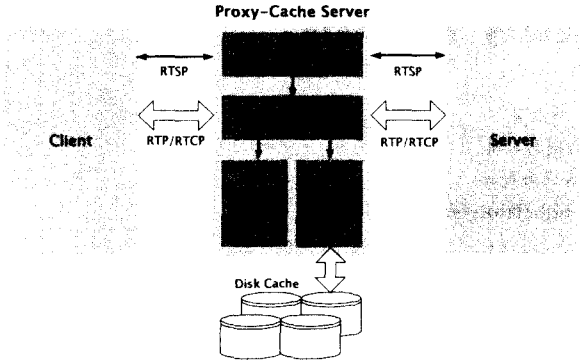

그림 2 스트리밍 캐쉬 서버의 구조

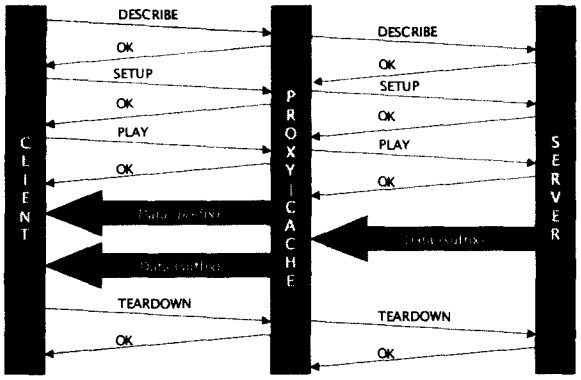

그림 3 RTSP 세션 처리와 데이타 교환

의 스트리밍 프로토콜을 이용함으로 서버 세션과 클라이언트 세션을 별개로 유지하고 캐싱될 콘텐츠 정보를 콘텐츠 관리자에게 제공한다. 콘텐츠 관리자는 세션 관리자로부터 수용된 세션에 대해서 클라이언트로 전송할 캐싱된 콘텐츠를 제공하고 서버로부터 받은 콘텐츠를 캐싱된 콘텐츠와 일관성있는 관리를 담당한다. 콘텐츠의 전송시 클라이언트의 성능 및 네트웍 대역폭에 따라 제공될 콘텐츠의 품질을 결정하고 캐싱될 콘텐츠에 대해 손실 관리를 담당한다. 프로토콜 제어기는 상위 세션 관리자와 콘텐츠 관리자의 RTSP/RTP/RTCP 프로토콜 처리를 담당한다. 세부적으로 보면 클라이언트와 서버에서 오는 세션 관리 프로토콜에 대한 번역하고 클라이언트에 대한 응답과 서버에 대한 요구 프로토콜을 생성한다. 캐쉬 관리자는 캐싱과 관련된 정책을 수행하고 세션 관리자와 콘텐츠 관리자에 세션유지를 위한 캐싱된 콘텐츠 정보를 전달한다. 이와 같은 캐쉬 서버의 구성은 기존의 스트리밍 서버의 서비스 방식인 내려받기와 On-Demand 방식에 착안한 것이다. 즉, 캐쉬 서버의 On-Demand 서비스는 서버 세션에 대하여 내려받기 방식으로 콘텐츠를 캐싱하고 클라이언트에는 On-Demand 방식 서비스를 적용한 것이다.

3. RTSP 세션 처리

다음에는 RTSP 세션 관리를 위한 세부 절차와 On-Demand 캐쉬 서비스를 위한 응용을 제시한다. 캐쉬 서버의 세션 설정 과정은 *describe, setup, play, teardown* 메소드로 구성되고 그림 3에서와 같이 단계별 세션 상태가 정의 된다. 세션 설정 과정은 메소드의 용도에 따라 두 단계로 나눠진다. 첫 번째 단계는 세션의 준비 단계로 *describe* 메소드가 사용된다. *describe*는 클라이언트가 요구하는 서비스의 유무를 판단함으로서 세션을 설정할 것인지를 결정한다. 즉 세션의 설정과 무관하게 콘텐츠에 대한 정보만을 얻기위해 *describe* 단계를 수행한다. 캐쉬 서버에서는 이 단계의 클라이언트의 요구를 받아 캐쉬 히트와 미스를 판단하여 서비스 방식을 결정한다. 두 번째 단계는 *setup, play, teardown*의 단계 구성으로 서비스 세션의 개설과 종료에 관계된다. 이 단계에서는 클라이언트와 서버가 스트리밍 관점에서 자원 할당과 해제 과정을 수행한다.

그림 3는 RTSP 메소드 교환 과정 이외에 캐쉬의 기본 정책 중 캐쉬 히트 상황에 대한 시나리오를 보여준다. 캐쉬 서버는 *play* 메소드 이후 요구된 콘텐츠가 캐쉬되어 있다면 Prefix 콘텐츠를 클라이언트에 전송하고, 동시에 서버에 Suffix 콘텐츠를 받아서 Prefix 콘텐츠에 연결한다. 캐쉬 미스인 경우, 캐쉬 서버는 서버에서 전송된 Suffix 콘텐츠에 대해서 캐싱을 수행하고 바로 클라이언트에 전달하는 구조를 갖는다. RTSP의 메소드별 세부 절차를 알아보고 캐쉬의 히트와 미스를 나누어 캐쉬 서버의 세션 설정 과정을 살펴본다. 다음은 *describe* 요청의 예이다.

```
DESCRIBErtsp://parcom1.kwangwoon.ac.kr:554/powermac\_g4\_cube
-h.mov RTSP/1.0
CSeq: 1
Accept: application/sdp
Bandwidth: 2147483647
Accept-Language: en-US
User-Agent: QTS (qtver=5.0b11;os=Windows NT 5.0Service Pack 1)
```

클라이언트의 *describe* 요청은 요구된 콘텐츠의 URI 정보와 클라이언트의 상태 정보를 가지고 있다. 상태 정보에는 프로토콜, 대역폭, 언어 인코딩 그리고 클라이언트 버전 정보를 담는다. 캐쉬 서버에서는 *describe*의 URI 정보를 통해 현재의 캐쉬 히트을 판단하고 서버로 보낼 콘텐츠 요구를 결정한다. 다음은 서버의 *describe* 응답이다.

```
RTSP/1.0 200 OK
Server: QTSS/3.0Preview [v240]-Linux
. . .
Content-Type: application/sdp
x-Accept-Retransmit: our-retransmit

Content-Base:rtsp://parcom1.kwangwoon.ac.kr:554/powermac\_g4\_cu
be-h.mov/
v=0
s=/powermac\_g4\_cube-h.mov
. . .
c=IN IP4 128.134.64.111
. . .
m=audio 0 RTP/AVP 96
b=AS:47
a=rtpmap:96 X-QDM/22050/2
a=control:trackID=3
a=x-bufferdelay:3.76
m=video 0 RTP/AVP 97
b=AS:767
a=rtpmap:97 X-QT/600
a=control:trackID=4
```

describe 응답은 서버 및 콘텐츠 관리에 대한 RTSP 메시지와 콘텐츠 내역을 기술하는 SDP 프로토콜로 나누어 진다. RTSP

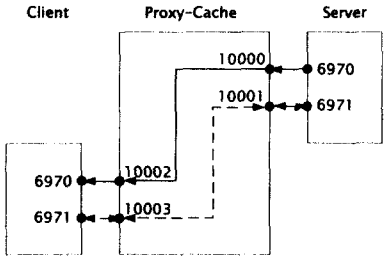

그림 4 Port Tunneling

메시지는 콘텐츠의 URI 확인과 유효성 정보, 프로토콜 등을 기술한고 SDP 프로토콜에서는 콘테츠에 대한 상세 정보로 콘텐츠의 실제 위치와 클라이언트를 위한 미디어 혹은 프리젠테션의 유형, 트랙 번호등을 기술한다. 다음은 *setup* 요청이다.

SETUPrtsp://parcom1.kwangwoon.ac.kr:554/powermac_g4_cube-h.
. . .
Transport: RTP/AVP;unicast;client_port=6970-6971
x-retransmit: our-retransmit
x-transport-options: late-tolerance=1.867000
User-Agent: QTS (qtver=5.0b11;os=Windows NT 5.0Service Pack 1)
Accept-Language: en-US

클라이언트의 *setup* 요청은 *describe*에 기술된 URI와 요청하는 미디어의 트랙번호를 기술한다. 트랙에 대한 정보는 *describe* 응답의 SDP 부분에 기술되어 있다. *Transport* 인자에는 클라이언트에서 지원하는 프로토콜을 나열하고 해당 프로토콜에 대한 포트를 배정해 서버가 참조할 수 있게 한다. 위 예에서 클라이언트 포트는 RTP : 6970, RTCP : 6971을 배정하였다. 서버의 *setup* 응답에는 클라이언트 포토 배정에 대한 응답으로 서버 포트를 배정하고 클라이언트가 참조할 수 있도록 통보하는데 그림 4와 같다. 다음은 *setup* 응답의 예이다.

RTSP/1.0 200 OK
Server: QTSS/3.0Preview [v240]-Linux
. . .
Session: 8726587826244070630
Date: Tue, 27 Mar 2001 02:12:44 GMT
Expires: Tue, 27 Mar 2001 02:12:44 GMT
Transport:RTP/AVP;unicast;client_port=10000-10001;source=128.134.64.111;
server_port=6970-6971
x-Retransmit: our-retransmit

캐쉬 서버의 *setup* 요청에 대한 서버의 응답이다. *setup* 요구가 처음이므로 서버는 *Session* 필드를 통해 현 세션에 대한 고유 식별자를 할당한다. *setup* 응답에서 캐쉬 서버에 의해서 서버의 포트가 10000-10001로 할당되었음을 알 수 있고, 클라이언트는 6970-6971이 할당되었다. 다음은 세션을 완성하고 서비스의 시작을 알리는 *play* 과정이다.

PLAYrtsp://parcom1.kwangwoon.ac.kr:554/powermac_g4_cube-h.m
. . .
Range: npt=0.000000-360.985000
x-prebuffer: maxtime=2.000000
Session: 8726587826244070630

play 요청은 기술된 URI로 요구된 콘텐츠에 대한 스트리밍과 시작 및 끝 시간을 *Range*를 통해서 *npt* 혹은 *smpte*로 설정할 수 있다. Suffix 캐싱인 경우 전송받을 콘텐츠의 위치를 나타낸다. 다음은 *play* 응답이다.

그림 5 클라이언트

RTSP/1.0 200 OK
. . .
RTP-Info:
url=trackID=3;seq=17115;ssrc=1116617086;rtptime=1495207282,
url=trackID=4; seq=20706; ssrc=253917952;rtptime=2095478881

Play의 *RTP-Info*는 *url* 인자 값으로 *trackID*, *seq*, *ssrc*, *rtptime*을 할당함으로써 캐쉬 콘텐츠의 일관성 유지를 위한 정보로 활용한다.

4. 결론

클라이언트와 서버는 Apple사에서 제공하는 Quick-Time Player Pro와 Darwin 서버를 활용하였고, 콘텐츠는 Quick-Time Hinted Track를 사용하였다. 그림 5는 클라이언트의 실행모습을 보여준다.

본 논문에서는 멀티미디어 콘텐츠와 사용자의 증가에 따른 네트웍 대역폭의 부족 현상을 보완하고 표준 프로토콜 RTSP을 지원하는 On-Demand 캐쉬 서비스 환경 구성에 있어 핵심 요소인 멀티미디어 캐쉬 서버를 개발하고 설계하였다. 세션 운영 기법은 클라이언트와 캐쉬 서버, 캐쉬 서버와 서버의 관계가 캐싱 시스템를 중심 매개체로한 인터페이스 구조로 설계되었다.

향후 연구과제는 비손실 전송기법, 효율적인 콘텐츠 저장 정책과 관리정책등이 있으며 상용화된 제품과 호환되도록 프로토콜을 다양화해야 한다.

참고문헌

[1] R. Rejaie, M. Handley, H. Yu, and D. Estrin, "Proxy caching mechanism for multimedia playback streams in the Internet", *Proceedings of 4th International WWW Caching Workshop*, Mar 1999.

[2] Brain D. Davison, "A Survey of Proxy Evaluation Techniques", *Proceedings of 4th WWW Caching Workshop*, San Diago, March 1999.

[3] Schulzrinne. H., Real Time Streaming Protocol (RTSP), RFC 2326, April 1998.