새로운 Peer-to-peer 미디어 스트리밍 프로토콜 설계 및 구현

*정태준, *이홍래, *서광덕, **김성혜, **강신각
*연세대학교 컴퓨터정보통신공학부, **한국전자통신연구원
*jeung86@naver.com

Design and Implementation of Novel Peer-to-peer Media Streaming Protocol

*Jung, Tae-Jun, *Lee, Hong-Rae, *Seo, Kwang-deok, **Kim, Sung-Hei, and **Kang, Shin-Gak
*Computer and Telecommunications Engineering Division, Yonsei Univ., **ETRI

요약

향후 인터넷 기반 애플리케이션에는 확장성, 보안 및 신뢰성, 새로운 서비스에 대한 유연성 및 QoS 등의 요구사항이 중요하다. 기존 클라이언트-서버 방식에서 이러한 요구 사항의 만족을 위해서는 복잡성 및 고비용 문제가 제기된다. 반면, 모든 형태의 분산 자원 접근이 가능한 P2P 통신 방식에서는 보다 간단한 해결방안을 제시함으로써 인터넷 기반 애플리케이션에 새로운 가능성을 제시하고 있다. 인터넷의 전통적인 클라이언트-서버 패러다임과는 달리 완전 분산 및 자율 조직 특성을 가진 P2P 개념은 확장성 및 신뢰성 측면에서 장래의 애플리케이션, 시스템 요소, 인프라 서비스를 위한 기본 설계지침으로 제시되고 있다. 본 논문에서는 P2P 실시간 스트리밍 프로토콜의 새로운 모델인 PREP 프로토콜을 설계 및 구현한다. 제안된 P2P 실시간 스트리밍 프로토콜을 NS-2를 통해서 설계하고 시뮬레이션을 통해 P2P 기반의 스트리밍 서비스의 가능성을 확인한다.

1. 서론

최근 들어 인터넷 상에서 P2P (peer-to-peer) 파일 공유 응용 프로그램이 활발하게 사용되면서 P2P에 대한 관심이 더욱 커지고 있다. 1999년 5월 파일 공유를 위한 Napster의 소개로 시작된 Peer-to-Peer(이하 P2P) 네트워킹 기술은 분산 컴퓨팅, 인터넷 전화(Skype)에 성공적으로 적용됨에 따라, 현재 가장 관심 있는 인터넷상 새로운 통신방식으로 떠오르고 있다.

기존의 P2P 시스템과 P2P 미디어 스트리밍은 데이터 공유 모드에 따라 구별되며 “Open-After-Downloading” 과 “Play-While-Downloading”이 있다. 미디어 파일을 모두 다운로드 받은 후에 재생하는 P2P 응용 프로그램은 전자에, 다운로드와 동시에 재생을 하는 P2P 미디어 스트리밍은 후자에 속한다. P2P 미디어 스트리밍에서 미디어 요청 피어는 재생과 동시에 미디어 데이터를 저장하고 스트리밍이 끝나면 다른 피어에게 미디어 파일을 제공하는 새로운 소스 피어가 된다. P2P 미디어 스트리밍은 P2P 파일 공유에 비해 도전해야 할 과제가 많다. 그 이유는 P2P 네트워크를 구성하는 각 피어는 서로 다른 네트워크 대역폭, 업로드 속도, 컴퓨터 성능을 가지므로 임의의 하나의 피어로부터 데이터를 수신하면, 사용자의 QoS를 만족시키지 못할 가능성이 높기 때문이다. 뿐만 아니라 P2P 네트워크의 특성상, 피어는 언제든 온라인과 오프라인을 반복할 수 있어서 피어의 네트워크 이탈이 잦다. 따라서 하나의 소스로부터 데이터를 받는 기존의 P2P 패러다임을 P2P 미디어 스트리밍에 효과적으로 적용하기는 어렵다. 현재 미디어 스트리밍의 주체인 송신 피어 (소스 피어) 여러 개로부터 미디어를 수신하는 멀티 소스 스트리밍의 연구가 진행되고 있다 [4]. P2P 멀티 소스 스트림에 관한 연구로는 PALS, GNUSTREAM 등이 있다.

본 논문에서는 P2P 실시간 스트리밍 프로토콜의 새로운 모델인 PREP (P2P-based Real-time strEaming Protocol) 프로토콜의 설계를 제안한다. 제안된 PREP 프로토콜을 NS-2 시뮬레이션 프로그램을 통해서 구현하고 시뮬레이션을 통해 P2P 스트리밍 서비스의 가능성을 확인한다.

2. PREP 개요

PREP에 의해 구성된 네트워크 구조는 그림 1과 같다 [2].

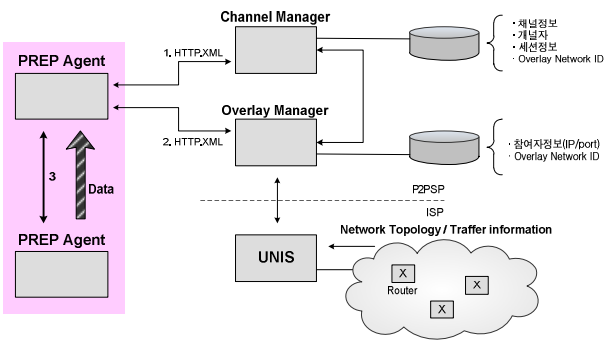


그림 1. PREP 네트워크 구조.

기본적으로 PREP Agent는 스트리밍 서비스를 수신하며, 다른 PREP Agent와 서비스 데이터를 공유한다. 여기서 채널매니저는 스트리밍 채널에 대한 정보를 가지고 있는 서버로 서비스 사용자가 접속하여 시청을 원하는 채널을 선택할 수 있게 해준다. 오버레이 매니저는 기존의 P2P 시스템에 있어서 Tracker 서버와 같은 기능을 하는 모든

채널을 관리하고 각 채널당 접속한 PREP Agent 들을 관리하는 서버이다. PREP 기반의 오버레이 네트워크는 하나의 스트리밍 피어와 하나 이상의 PREP client로 구성된다. 스트리밍 피어는 단방향으로 데이터를 전송하며, 각 PREP client는 다른 PREP client의 PREP Agent 부분과 통신하여 데이터를 공유한다.

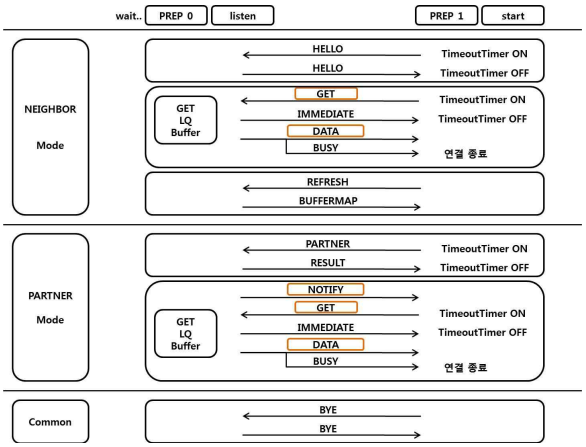


그림 2. PREP 프로토콜 흐름도.

임의의 두 PREP Agent간 관계는 Neighbor 와 Partner로 구분되며, 관계에 따라서 PREP Agent의 동작이 달라진다. Neighbor 관계의 두 PREP Agent는 동등한 관계로 서로 필요한 데이터를 주고받을 수 있다. Partner 관계는 둘 중 하나의 PREP Agent가 일방적으로 데이터를 보내주는 관계이다.

3. PREP 구조 및 설계

3.1 PREP 메시지 구조

메시지 헤더는 PREP Agent간 교환하는 모든 메시지에 사용되고 구조는 다음 그림 3과 같다.

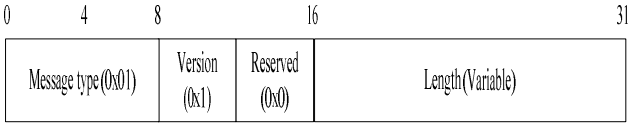


그림 3. PREP 헤더 구조.

Message type 은 어떠한 메시지인지 메시지 종류에 대한 정보를 담고 있다. Version 은 PREP의 버전정보를 포함하고 있고, Reserved 필드의 경우는 현재는 사용하지 않지만 예약 필드로서 차후에 어떻게 사용될지 결정되지 않은 필드이다. Length 는 메시지 길이에 대한 정보를 담고 있다.

3.1.1 Hello 메시지

Hello 메시지의 목적은 PREP Agent가 다른 PREP Agent와 연결 시 교환하는 메시지로 Agent가 어떤 데이터를 저장하고 있는지에 대한 정보를 담고 있는 버퍼맵을 교환함으로써 부족한 데이터가 어떤 부분이 있는지에 대한 내용을 교환하고 확인하는 것이다. Overlay ID는 Overlay Network의 ID를 포함하고 있다. Agent ID는 PREP Agent의 ID를 포함하고 있다. Hello 메시지는 Buffer map 의 정보를 담고 있는데 STB와 SHB로 나뉘어진다. STB는 Stored buffer로 현재 송수신 중인 버퍼구간 (SHB)에 속하지는 않지만 이미 지나간 구간에 대한 공유를 위해서 보유하고 있는 구간을 말한다. SHB는 Sharing buffer로 현재 송수신 중인 구간으로 재생을 위한 구간이다 특히, 원활한 재생을 위해서 Playback index를 포함한 Urgent section은 먼저 수신한다. 그림 4는 STB와 SHB 그리고 Urgent section을 보여주고 있다.

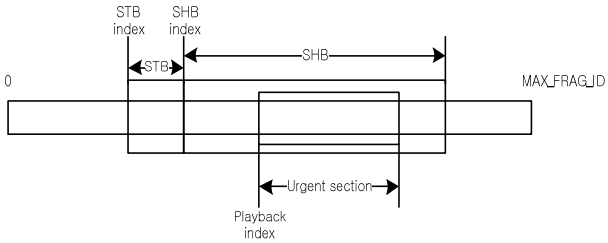


그림 4. Buffermap 구조.

STB index는 STB의 시작 index를 뜻한다. STB length는 STB의 길이를 뜻하고 있고, SHB index는 SHB의 시작 index를 뜻한다. SHB length는 SHB 버퍼의 길이정보를 담고 있다. SHB map의 경우는 SHB에 대한 buffermap 정보를 담고 있고 필드길이는 buffermap 정보에 따라서 유동적으로 바뀌게 된다 [1].

3.1.2 GET 메시지

HELLO 메시지를 교환한 후, 상대방이 자신에게 필요한 데이터를 가지고 있는 경우에 해당 데이터를 요청하기 위한 메시지이다. Fragment 단위로 요청하며, fragment 내의 block 을 지정하여 자신이 필요로 하는 block 부터 수신할 수 있다.

3.1.3 DATA 메시지

데이터 전송요청에 대해서 수락한 경우에 데이터를 전달하기 위한 메시지이다. Fragment 를 구분하기 위한 index와 시작 offset, 크기를 함께 기술하여 데이터와 함께 전송한다. Fragment index에는 전송하는 fragment 의 index가 있다. Block offset은 전송되는 block의 byte offset에 대한 정보를 포함하고 있다.

3.1.4 BUSY 메시지

데이터 전송요청에 대해서 거부를 하는 경우에 전송하는 메시지이다. Reason code 즉 전송요청에 대한 거부를 하는 이유에 대한 정보가 담겨있다. 전송요청에 대한 거부는 여분의 대역폭이 없거나 허용된 전송 수 초과 또는 CPU Busy 정도로 볼 수 있다.

3.1.5 DATACANCEL 메시지

상대방 PREP Agent로부터 데이터를 전달받고 있는 경우에 데이터 전송의 취소를 요청하기 위해서 전송한다. 전송취소를 요청하는 fragment의 index의 정보를 포함하고 있다.

3.1.6 BUFFERMAP 메시지

PREP Agent간에 데이터를 공유하기 위해서 각 PREP Agent가

보유하고 있는 데이터의 상황을 알리기 위한 메시지이다. Sliding window에 속하는 fragment에 대한 정보를 알리기 위해서 window 관련 정보 (window point index) 를 포함하고 있다. 또한 sliding window 외에 연속적으로 보유하고 있는 fragment 의 정보를 알리기 위해서 starting point index와 complete number로 시작되는 fragment 번호와 개수를 알린다. STB index 는 STB 시작 index정보를 포함하고 있다. STB length는 STB 버퍼의 길이정보를 포함하고 있다. SHB index 는 SHB시작 index 정보를 포함하고 있고 SHB length 는 SHB 버퍼의 길이정보를 포함하고 있다. SHB map의 SHB에 대한 buffermap 의 정보를 담고 있기 때문에 필드 길이는 데이터의 양에 따라서 유동적으로 바뀌게 설정되었다.

3.1.7 REFRESH 메시지

상대방 PREP Agent에게 버퍼맵을 요청하는 메시지로서 필요로 하는 fragment에 대한 갱신을 하기 위해서는 refresh 메시지 요청을 하면 현재 요청 받은 Agent의 fragment의 상태에 대해서 응답을 보내주는 메시지이다. Fragment index에는 어떤 fragment들이 수정이 되었는지에 대한 정보가 들어있고 Number of fragment에는 fragment의 개수에 대한 정보가 포함되어 있다

3.1.8 BYE 메시지

BYE메시지는 연결을 종료하기 위한 메시지이다. BYE메시지를 보낸 후에 연결되어있는 Agent 와 접속을 끊는 기능을 하고 있다.

3.1.9 PARTNER 메시지

파트너로 등록을 요청하는 메시지이다. GET과 DATA의 비율이 5:1 이상일 경우 PARTNER메시지를 전송하도록 설정이 되어있다.

3.1.10 RESULT 메시지

파트너 요청의 결과를 알려주는 메시지이다. PARTNER관계가 되었는지 실패했는지에 대한 정보를 담고 있다. RESULT필드에 그에 대한 정보가 포함되어있다.

3.1.11 NOTIFY 메시지

새로운 fragment를 수신 시에 자신의 파트너에게 알리기 위한 메시지이다. NOTIFY메시지는 PARTNER관계일때만 동작 하며 해당 Fragment index위치에 Fragment 가 존재할 경우 NOTIFY 메시지를 전송한다. Fragment index필드는 fragment 정보에 대한 내용이 담겨져 있다.

3.2 타이머 작동원리

현재 NS2 시뮬레이션 환경은 multi thread가 돌아가지 않기 때문에 Timer를 사용함으로써 thread 처럼 구동이 가능하게 할 수 있다 [3]. 타이머의 작동 원리는 그림 5와 같이 메시지가 전송되면서 Timer가 동작한다. 일정시간이 지나도 응답이 없으면 Expire() 함수를 호출하게 되고 Expire() 함수가 호출되고 나면 정의된 함수가 호출되게 되고 Timeout()함수가 호출된다. Timeout()함수 호출 후에는 다음 Timer 동작시간이 설정된다.

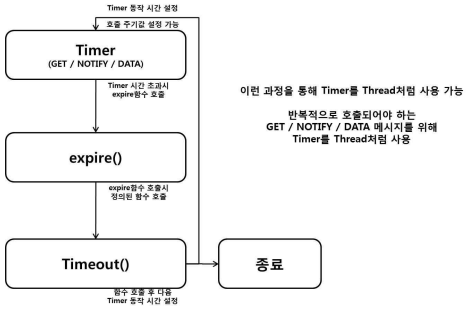


그림 5. Timer 함수의 작동원리.

3.3 Fragment 와 Block의 관계

Fragment 는 비디오 전송시 미디어 파일을 Fragment로 쪼개게 된다. Fragment는 advertisement 단위이고 하나의 크기는 16,384 byte로 설정되어 있다. Block은 Fragment 보다 작은 조각으로 하나의 Fragment를 다시 여러 개로 Block으로 쪼갠다. Fragment가 수신이 완료가 되는 조건은 Fragment내의 Block이 수신완료 되는 시점부터 수신완료라고 한다.

3.4 전송 요청이 필요한 Fragment 식별방법

Buffermap은 Fragment의 정보를 포함하고 있으며 자기 자신이 어떤 Fragment를 가지고 있고 어떤 Fragment를 요청해야 하는지에 대한 정보를 가지고 있다. 하지만 피어가 상대방 피어의 Buffermap 정보만 가지고 어떤 Fragment를 요청해야 하는지 식별하는 방법이 필요하다. 그림 6과 같이 상대방의 Buffermap을 수신하게 되면 자기 자신의 Buffermap 과 XOR연산을 한다. 그렇게 되면 그 결과만 가지고는 나에게 없는 Fragment 정보를 구별할 수가 없다. 하지만 자신의 Buffermap 정보를 그 결과와 다시 XOR 연산을 한번 더 취하게 되면 어떤 Fragment를 요청해야 하는지 전송요청이 필요한 Fragment 정보를 얻어낼 수 있다.

그림 6. 전송 요청이 필요한 Fragment 구별법.

3.5 Buffermap 이동조건

Buffermap에서 고려해야 할 또 다른 조건은 Buffermap 이동조건이다. Notify의 경우에는 해당위치에 Fragment가 존재하는지 확인을 하고 존재하게 되면 PARTNER index위치가 SHB_Buffermap 에서 STB_Buffermap 을 뺀 위치인지 확인을 한다. 뺀 위치에 있는지 확인

이 되면 Buffermap의 위치가 이동 가능한지 확인을 하게 되고 이동하게 된다. GET(REFRESH요청)의 경우에는 GET을 요청할 Fragment가 존재하지 않는지 확인하게 되고 존재하지 않으면 PARTNER_index 위치가 SHB_index와 STB_Buffermap의 합보다 큰지 확인하고 크면 REFRESH 메시지를 요청하거나 대기한다. REFRESH 메시지가 요청되면 이동가능한지 확인후 Buffermap을 이동한다 (그림 7 참조).

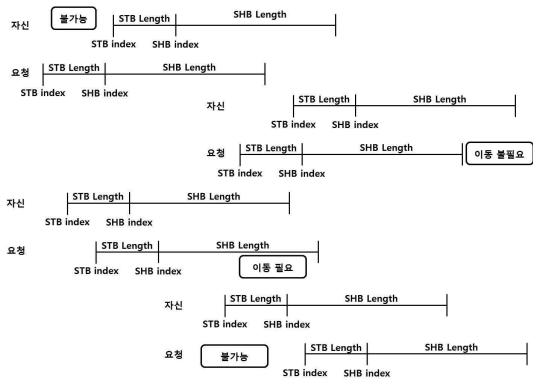


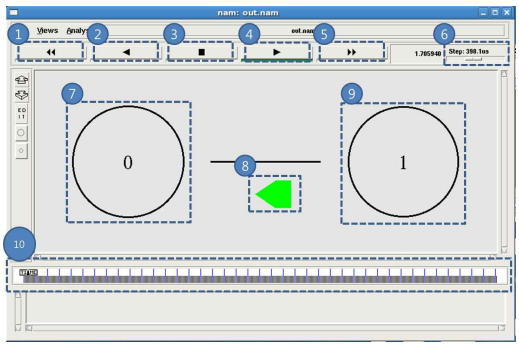
그림 7. Buffermap 이동조건.

4. PREP 구현 결과

4.1 NS-2 시뮬레이터 실행

NS-2에서 PREP 시뮬레이션을 시작하게 되면 nam을 통해서 어떻게 패킷이 주고 받는지에 대한 동작 절차와 과정을 다음과 같은 시각적인 표현을 통해 확인할 수 있다.[3]

① 시뮬레이션 뒤로감기 ② 시뮬레이션 거꾸로 재생
③ 시뮬레이션 정지 ④ 시뮬레이션 시작
⑤ 시뮬레이션 빠르게 감기 ⑥ 시뮬레이션 시간간격 조절바
⑦ 도착피어
⑧ 전송이 되고 있는 데이터 (길이가 데이터량에 따라 바뀜)
⑨ 출발피어 ⑩ 시뮬레이션 시간바: 원하는 시간대로 조절가능



4.2 Trace 파일 분석

NS-2 에서는 자체적으로 Trace 파일을 분석할 수 있도록 Trace 파일을 남기게 설계되어 있다.[3] 하지만 기존의 NS-2에서 제공하는 Trace 파일 즉 피어간 데이터가 오고간 로그파일을 가지고는 각 Peer 별로 메시지 전송 및 네트워크 관련 정보를 분석하기가 어렵다. 그렇기 때문에 표 1과 같은 별도의 Trace 파일을 기록하여 PREP 프로토콜을 분석하는데 있어서 용이하게 하였다.

표 1. Trace 파일 정보.

status	패킷의 송수신 상태 표현 (S/R)
Overlay ID	Overlay network의 식별정보
Peer ID	Overlay network에 참가한 Peer식별정보
Dest. ID	Fragment 가 전달될 목적지 정보
Message Type	전달되는 Message type
Buffermap Index	전달되는 Fragment Index
Block Index	전달되는 메시지에 포함되는 block index
Packet Size	전달되는 메시지 크기
Peer Status	Peer 상태 정보
Packet ID	전달되는 패킷의 sequence index

NS-2 시뮬레이션 결과 Trace 파일의 정보는 아래의 그림 8에서와 같이 피어에 대한 정보들을 가지고 있다.

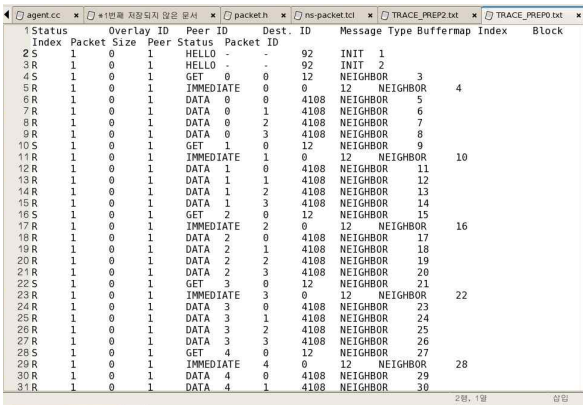


그림 8. Trace 파일 정보.

5. 결론

기존의 P2P 기반 스트리밍 서비스의 문제점을 도출하여 이를 기반으로 새로운 프로토콜인 PREP을 설계하고 NS-2 시뮬레이터를 통해 구현하였다. 그러나, 유동적인 피어의 참여와 이탈로 인해서 지속적이고 안정적으로 피어가 유지 되도록 프로토콜을 보완할 필요가 있다. PREP 기반의 P2P 스트리밍 기술은 아직 초기단계이지만 차세대 네트워크의 핵심적인 역할을 담당할 것으로 기대한다.

6. 참고 문헌

[1] J. Liu, S. G. Rao, B. Li, and H. Zhang “Opportunities and challenges of peer-to-peer internet video broadcast,” *Proc. of the IEEE*, vol. 96, no. 1, pp. 11-24, Jan. 2008.
[2] B. Li, H. Yin “Peer-to-peer live video streaming on the internet : issues, existing approaches, and challenges,” *IEEE Commun. Magazine*, vol. 45, no. 6, June 2007.
[3] K. Fall, K. Varadhan “The NS manual (formerly ns notes and documentation)” The VINT Project-A collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC. May 2010
[4] D. Ren, Y.-T. H. Li, S.-H. G. Chan “On reducing mesh delay for peer-to-peer live streaming,” in Proc. *IEEE INFOCOM*, pp. 1058-1066, 2008.