일반논문-11-16-1-15

HTTP Streaming 환경에서 User Profile 기반 Seamless Framework 제공방법

김 정 한[a], 이 장 원[a], 김 규 헌[a]‡, 서 덕 영[a]

User Profile Based Seamless Framework under HTTP Adaptive Streaming Environment

Junghan Kim[a], Jangwon Lee[a], Kyuheon Kim[a]‡, and Doug Young Suh[a]

요 약

최근 방송시스템의 디지털화와 통신기술의 발달로 기존의 고정형 디스플레이 장비와 전용채널을 통해 주로 이루어지던 콘텐츠 소비 형태가 변화하고 있다. 기존 고정형 디스플레이 장비를 통해 정해진 시간과 장소에서만 소비 할 수 있었던 미디어 콘텐츠는, 최근 언제/어디서나 IP(Internet Protocol)망에 접속하여 콘텐츠를 선택/소비 할 수 있는 단말기의 등장과 유동적인 IP환경에서도 사용자의 전송채널에 따라 적응적으로 스트리밍 서비스를 제공하는 HTTP(Hypertext Transfer Text Protocol) 기반의 적응형 스트리밍 기술로, 사용자는 과거의 제약에서 벗어나 콘텐츠를 소비 할 수 있게 되었다. 이와 같은 방송환경의 변화로, 사용자는 자신이 보유한 단말기들을 환경에 따라 변경해 가면서 콘텐츠를 소비 할 수 있게 되었다. 하지만, 현재 제공되고 있는 HTTP 적응형 스트리밍 서비스에서는 사용자가 소비하던 콘텐츠를 단말기를 변경하여 연속해서 소비하고자 하는 경우, 사용자가 콘텐츠의 종료 시점을 기억하여, 직접 변경된 단말기에 적용해야 해야 한다는 문제점이 있다. 따라서, 본 논문은 이러한 문제점을 해결하기 위해, 동일한 콘텐츠를 소비하는 단말간의 연속된 콘텐츠 소비환경을 위한 데이터를 기술하는 유저 프로파일(User Profile)을 정의하고, 이 파일을 변경 된 단말기와 공유하여, 단말기 사이의 연속적인 콘텐츠 소비를 가능하게 하는 방법을 제안하고자 한다. 또한, 본 논문은 제안한 방법을 검증하기 위해 HTTP 적응형 스트리밍 서비스 중에 하나인 Smooth Streaming 기반의 시스템을 구현하여 제안한 방법을 검증하고자 한다.

Abstract

Recently, with the digitalization of a broadcasting system and the development of a communication technology, the existing trend of consuming media contents throughout the fixed-displayer and the dedicated channel is being changed. The existing user only could consume media contents under limited time and places because of the fixed-displayer and the dedicated channel. However, with advent of the IP-based terminals and HTTP adaptive streaming which transfer the media sequence according to the user's transmission condition, users become possible to enjoy the media content anytime anywhere. As the result of the alteration of the broadcasting surrounding, users can enjoy the media content while changing his terminals according to their preferences and circumstances. However, in case that users try to consume consecutively the content from last view-point ended in the previous terminal under current HTTP adaptive streaming environment, a user has to remember the last view-point, and then has to apply the view-point to the changed terminal. Thus, for solving this problem, this paper defines "User Profile" for describing the metadata for the chained content consume environment between the terminals. Also, for proving the proposed method, this paper try to demonstrate the proposed method throughout the realization of the system based on Smooth Streaming from Microsoft.

Keyword : HTTP Adaptive Streaming, User Profile(유저 프로파일), Fragment, DASH, Smooth Streaming

Ⅰ. 서 론

오늘날, 멀티미디어 서비스의 패러다임은 시간과 장소에 제한받던 고정형 디스플레이 장비를 이용한 콘텐츠 소비 형태에서, 인터넷망 접속을 통해 언제/어디서나 고화질의 콘텐츠를 선택해서 소비하는 서비스 패러다임으로 변화하고 있다. 현재 상기와 같은 서비스를 지원하는 휴대용 노트북, 스마트 폰과 같은 다양한 단말기가 시장에 선보이고 있으며, 이러한 서비스를 제공하기 위한 다양한 어플리케이션이 제공되고 있다. 하지만, QoS(Quality of Service)를 보장하지 않는 IP 기반 전송환경에서의 고화질 미디어 스트리밍 서비스는 콘텐츠를 시청하는 소비자 관점에서 필수적으로 보장되어야 하는 끊김 없는 영상서비스를 제공하는데 제약이 있다. 따라서, 최근 이러한 문제를 해결하기 위한 방법으로 사용자의 요청(Request)과 이에 대한 응답(Response)의 구성되는 통신방식을 통해 서비스를 제공하는 HTTP 기반의 적응형 스트리밍 서비스가 제안되었다. 종래의 스트리밍 서비스 환경에서는 사용자가 스트리밍 서비스를 요청하고, 이에 대해 서버가 일방적으로 콘텐츠를 전송하는 방식 때문에, 사용자는 자신의 전송환경에 적합한 콘텐츠를 선택하여 소비하는 것이 쉽지 않았다. 하지만, 그림 1에 나타난 바와 같이, 제안된 HTTP 적응형 스트리밍은 종래의 스트리밍 서비스처럼 서버가 일방적으로 콘텐츠를 전송하는 방식이 아닌, 클라이언트가 서버에 준비되어 있는 다양한 퀄리티의 미디어 시퀀스(Media sequence)를 자신의 전송채널에 따라 실시간으로 직접 요청해서 가지고 옴으로서, 클라이언트의 전송환경에 최적화된 미디어 스트리밍 서비스를 제공 한다[1][2][3][4][5].

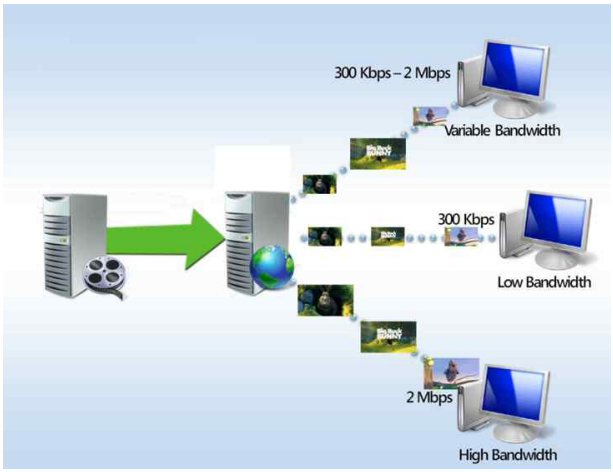


그림 1. HTTP Adaptive Streaming 서비스
Fig. 1. HTTP Adaptive Streaming service

이러한 HTTP 기반의 적응형 스트리밍 서비스의 등장으로, 서비스 제공자는 IP망을 통해 안정적으로 스트리밍 서비스를 제공 할 수 있게 되었고, 사용자는 자신이 보유한 단말기를 통해 언제, 어디서나 안정적으로 콘텐츠를 소비할 수 있게 되었다. 하지만, 개인이 다수의 단말기를 보유하게 되면서, 사용자는 상황에 따라 자신이 보유한 단말기를 변경해 가면서 콘텐츠를 소비하게 되었다. 하지만, 기존의 HTTP 적응형 스트리밍 서비스에서는 사용자가 콘텐츠를 소비하던 도중 종료하고, 단말기를 변경하여 콘텐츠를 연속해서 소비하는 경우, 사용자가 변경 전 단말기의 콘텐츠 종료 시점을 변경 된 단말기에 직접 적용해야 한다는 문제점을 있다. 따라서, 본 논문은 상기 언급한 문제를 해결하기 위해, 그림 2의 서비스 시나리오와 같이 단말기 간의 연속된 콘텐츠 소비를 가능하게 하는 정보를 제공하는 유저 프

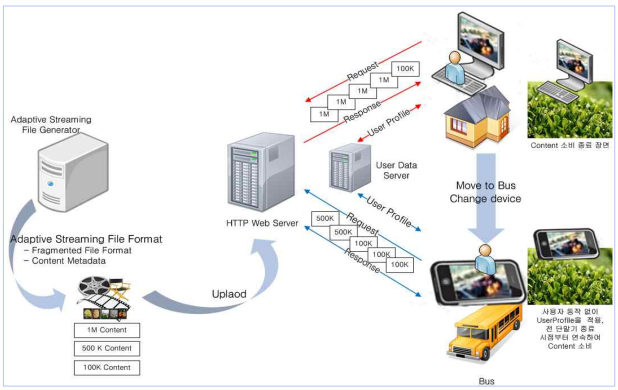


그림 2. 유저 프로파일 기반 콘텐츠 소비 환경유지 서비스 시나리오
Fig. 2. Service scenario for keeping the content consume environment between terminals, based on User Profile

a) 경희대학교 전자정보대학
College of Electronics and Information, Kyung Hee University
‡ 교신저자 : 김규헌(kyuheonkim@khu.ac.kr)
※ '본 연구는 지식경제부 및 정보통신산업진흥원의 대학IT연구센터 지원사업의 연구결과로 수행되었음'(NIPA-2011-(C1090-1011-0001))
· 접수일(2010년12월14일),수정일(2011년1월13일),게재확정일(2011년1월13일)

로파일(User Profile)을 정의하고, 해당 파일을 사용자 데이터 서버(User Data Server)를 통해 공유함으로써, 사용자가 보유한 단말기 사이에서 연속적인 콘텐츠 소비를 가능하게 하는 방법을 제안한다. 본 그림에 나와 있는 유저 데이터 서버는 클라이언트의 요청에 따라 유저 프로파일을 식별하여 저장 또는 전송 할 수 있는 기능을 제공하는 서버로서, 앞서 언급한 기능을 제공 할 수 있는 모든 종류의 서버로 대체 될 수 있다. 또한, 서비스 제공 방법에 따라 콘텐츠를 제공하는 하는 서버가 유저 데이터 서버의 기능을 같이 할 수 있다.

II. 배경기술

1. HTTP 적응형 스트리밍(HTTP Adaptive Streaming)

방송 시스템의 디지털화와 통신기술의 발달로 IP 기반 전송채널을 통한 고화질의 미디어 스트리밍 서비스 제공이 가능하게 되었다. 하지만, 시시각각 변화하는 IP 전송채널을 통한 고화질의 미디어 스트리밍 서비스는 소비자들이 콘텐츠를 소비하는 도중 전송지연(Transmission delay)으로 인한 영상 끊김 문제를 발생시켰다. 따라서, 최근 이러한 문제를 해결하기 위한 방안으로 그림 3과 같이 클라이언트 요청과 이에 대한 서버의 응답으로 구성되는 통신방식을 통해 서비스를 제공하는 HTTP 적응형 스트리밍 서비스가 제안 되었다[2][4][5].

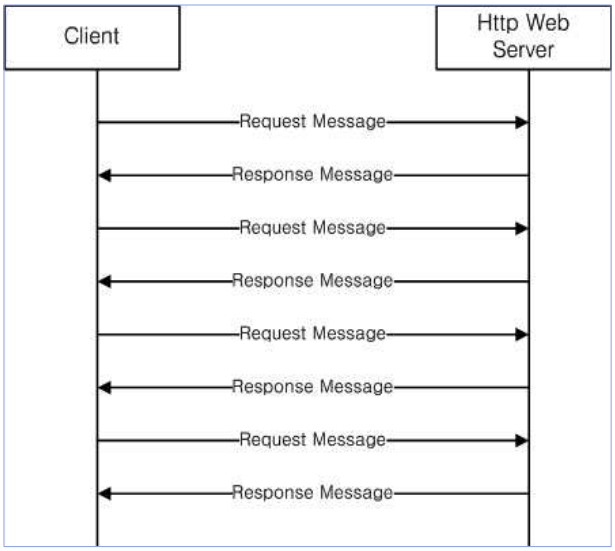


그림 3. HTTP 통신 패턴(Request & Response)
Fig. 3. HTTP communication pattern(Request & Response)

HTTP 적응형 스트리밍 서비스는 기존의 전형적인 스트리밍 방식에서 클라이언트가 서비스를 요청하고 이에 대해 서버가 일괄적으로 데이터를 전송하는 방식이 아닌, 클라이언트가 직접 자신의 전송채널의 가용 대역폭에 해당하는 미디어 시퀀스를 직접 선택/요청하여 서버로부터 가지고 오는 방식을 사용하여, 클라이언트의 전송환경에 최적화된 스트리밍(Streaming) 서비스를 제공한다.

그림 4는 앞에 설명한 스트리밍 서비스를 위해 주로 이용되고 있는, 기존의 전형적인 스트리밍 서비스의 형태를 보여주는 그림으로, 클라이언트의 최초 서비스 요청에 서버가 일괄적으로 단일 퀄리티의 미디어 시퀀스를 전송하는 형태를 보여준다[3].

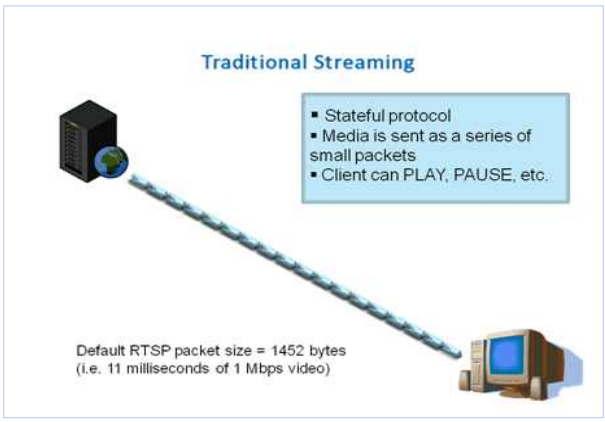


그림 4. 전형적인 스트리밍 서비스(RTP)
Fig. 4. Traditional streaming service(RTP)

이와 달리, 사용자의 요청에 적응적으로 미디어 시퀀스를 전송하는 HTTP 적응형 스트리밍 서비스는 그림 5에 나타난 바와 같이, 클라이언트가 서버에 준비되어 있는 다양한 퀄리티의 미디어 시퀀스 중에 자신의 전송채널에 적합한 미디어 시퀀스를 직접 선택/요청하여 소비하는 것을 볼 수 있다.

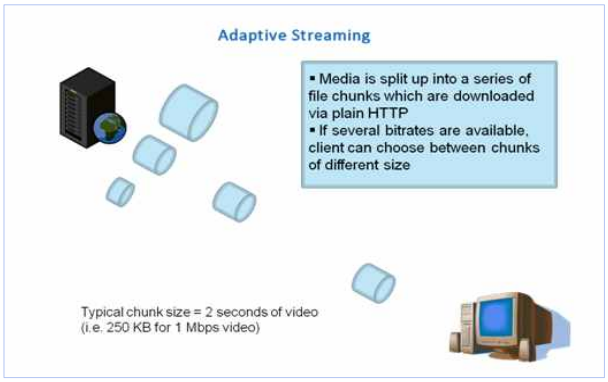


그림 5. HTTP 적응형 스트리밍 서비스
Fig. 5. HTTP Adaptive Streaming Service

하지만, 클라이언트가 자신의 가용 대역폭에 해당하는 미디어 시퀀스를 직접 선택하는 이러한 서비스 방식에서, 클라이언트는 서버에 준비되어 있는 미디어 시퀀스를 요청하기 위한 'URL', 'bitrate' 등과 같은 부가정보를 필요로 한다. 따라서, HTTP 적응형 스트리밍에서는 이러한 부가정보를 제공하기 위해, 서버에 준비되어 있는 다양한 퀄리티의 미디어 시퀀스에 대한 부가정보를 통합하여 기술하는 인덱스 파일(Index File)을 정의하고 있다. 인덱스 파일은 서버에 준비되어 있는 다양한 퀄리티의 미디어 시퀀스를 요청 할 수 방법 및 서비스를 위한 부가정보가 포함하고 있으며, 클라이언트는 이러한 인덱스 파일에 기술되어 있는 정보를 이용해 자신의 전송속도에 해당하는 미디어 시퀀스를 요청 할 수 있는 정보를 얻는다[1][3].

2. HTTP 적응형 스트리밍 서비스 형태 및 특징

2.1 DASH(Dynamic adaptive streaming over HTTP)

2.1.1 DASH 소개

현재 HTTP 적응형 스트리밍은 미디어 시퀀스 저장방식과 서비스 페이지(Service Page)에 따라 두 가지 형태로 구분될 수 있다. 첫 번째는, 세계적인 표준 단체인 MPEG (Moving Picture Experts Group)에서 표준화가 진행되고 있는 DASH(Dynamic adaptive streaming over HTTP)로서, 클라이언트가 서버에 저장되어 있는 파일을 절대주소를 통해 직접 요청하는 스태틱 페이지(Static Page) 기반의 서비스를 제공한다. 따라서, DASH는 각각의 미디어 시퀀스를 통합하여 단일 파일로 저장하는 형식이 아닌, 한 개 이상의 미디어 시퀀스를 포함하는 미디어 세그먼트(Media Segment)를 이용하여, 미디어 시퀀스를 단일파일로 저장하는 형태를 가진다. 표 1은 현재 DASH에서 서비스를 제공하기 위한 서비스 파일의 형태 및 특징을 나타내는 표로, 여기에 나와 있는 파일은 모두 개개의 파일로 서버에 저장된다.

표 1에 나타나 있는 각각의 서비스 파일들은 그림 6과 같이 일련의 미디어 세그먼트들이 퀄리티 별로 저장되며, 각각의 퀄리티에 해당하는 미디어 세그먼트들을 복호화 하기 위한 초기화 정보를 제공하는 이니셜라이세이션 세그먼트(Initialisation segment)가 퀄리티 별로 한 개씩 존재한다. 또한, 이니셜라이세이션 세그먼트와 미디어 세그먼트의 접근 정보 및 서비스 부가정보를 제공하는 MPD(Media Presentation Description)파일이 서버에 존재하는 것을 그림을 통해 볼 수 있다[1][5].

실례로, 그림 6에 나와 있는 것과 같이, 500Kbps와 1024Kbps로 인코딩 된 일련의 미디어 세그먼트들이 각각 퀄리티 별로 서버에 저장되어 있는 것을 볼 수 있으며, 미디어 세그먼트의 초기화 정보를 제공하는 이니셜라이세이션 세그먼트는 퀄리티 별로 있는 것을 볼 수 있다. 또한, 각각

표 1. DASH 서비스 파일 형태
Table 1. DASH service file type

구 분	특 징
MPD (Media Presentation Description) : 인덱스 파일	DASH 서비스를 위한 부가 데이터 제공 파일 (각 서비스 콘텐츠 별 1개씩 존재)
Initialisation segment (이니셜라이세이션 세그먼트)	미디어 세그먼트 초기화 정보 제공 파일 (각 퀄리티 별 1개씩 존재)
Media segment (미디어 세그먼트)	실제 미디어 데이터를 포함하는 파일 (각 퀄리티 별 1개 이상 존재)

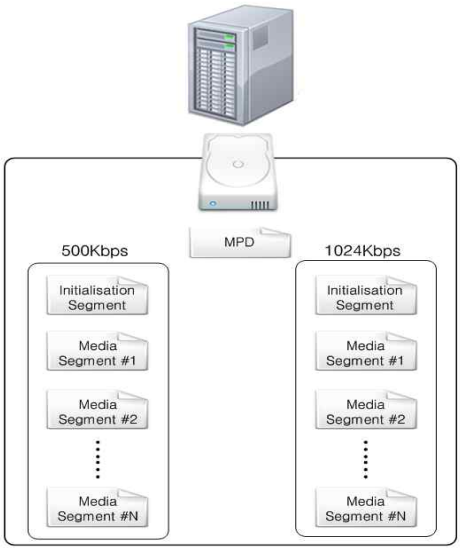


그림 6. DASH 서버의 파일 관리
Fig. 6. File management of DASH server

의 미디어 세그먼트와 이니셜라이세이션 세그먼트의 접근 정보 및 기타 부가정보를 제공하기 위한 MPD 파일이 서버에 저장되어 있는 것을 볼 수 있다. 최초 클라이언트는 서비스를 위해 서버에 저장되어 있는 세그먼트의 접근정보 및 기타 부가정보를 얻기 위해 서버에 MPD를 요청한다. 수신된 MPD의 파싱(Parsing)을 통해 클라이언트는 웹 서버(Web Server)에 저장되어 있는 각 퀄리티의 미디어 세그먼트와 이니셜라이세이션 세그먼트의 접근 주소 및 부가정보를 습득한다. 클라이언트는 습득 된 정보를 통해 자신의 전송채널 해당하는 퀄리티의 미디어 세그먼트를 선택한 다음 해당 미디어 세그먼트의 초기화 정보를 제공하는 이니셜라이세이션 세그먼트를 요청하여 초기화 정보를 얻은 후 미디어 세그먼트를 요청하여 콘텐츠를 소비한다. 만약, 콘텐츠를 소비하던 도중, 전송환경이 변하면 클라이언트는 변경 된 전송채널의 가용 대역폭에 해당하는 이니셜라이세이

표 2. MPD Element & Attribute
Table 2. MPD Element & Attribute

Element(E) or Attribute(A)	Description
MPD	최상위 Element
type	서비스 타입
mediaPresentationDuration	콘텐츠 플레이 시간
ProgramInformation	콘텐츠 일반 정보(ex. 저작권자, 관련 정보 URL)
Period	미디어 시퀀스를 시간 단위로 구분하기 위한 Element
start	미디어 세그먼트의 시작 시간
segmentAlignment	미디어 세그먼트의 시간 단위 정렬 여부
SegmentInfoDefault	Template 형태의 기술방법을 사용하기 위한 Element
Representation	미디어 세그먼트의 특성을 기술을 위한 Element
bandwidth	인코딩 비트레이트(bitrate)
width	가로 해상도
height	세로 해상도
lang	사용 언어
mimetype	파일 타입
group	그룹 넘버(Number)
ContentProtection	세그먼트 보안 정보 기술 Element
SegmentInfo	세그먼트 정보 기술 Element
duration	세그먼트 Duration
baseURL	세그먼트를 요청하기 위한 기본 URL
InitialisationSegmentURL	이니셜라이세이션 세그먼트 정의 Element
sourceURL	이니셜라이세이션 세그먼트를 요청하기 위한 주소
Url	미디어 세그먼트 정의 Element
sourceURL	미디어 세그먼트를 요청하기 위한 주소
UrlTemplate	미디어 세그먼트 Element(Template 형식)
sourceURL	초기화 세그먼트 정의 Element (Template 형식)
id	참조 id
startIndex	참조 시작 index
endIndex	참조 종료 index

션 세그먼트와 미디어 세그먼트를 MPD에 정의된 메타 데이터를 통해 식별하고, 앞서 설명한 바와 같이 다시 이니셜라이세이션 세크먼트를 요청해 초기화 정보를 얻은 후, 해당 미디어 세그먼트를 요청하여 끊김 없이 미디어 시퀀스를 소비한다.

2.1.2 Index File : MPD(Media Presentation Description)

DASH는 클라이언트의 요청과 이에 대한 서버의 응답으로 구성되는 통신방식을 통해, 클라이언트의 전송 대역폭에 해당하는 최적의 미디어 시퀀스를 서버에 요청하여 소비한다. 하지만, 클라이언트의 요청으로 시작되는 DASH 서비스의 클라이언트는 서버에 준비되어 있는 서비스 파일을 요청하기 위한 'URL', 'bandwidth' 등과 같은 부가정보들을 필요로 한다. 따라서, DASH에서는 상기 정보를 클라이언트에게 제공하기 위해, 표 2에 나타나 있는 Element와 Attribute를 통해 기술되는 MPD 파일을 통해 해당 정보를 제공한다. 표 3은 이러한 Element와 Attribute를 통해 512K와 1024K의 비트레이트로 인코딩 된 미디어 세그먼트 및 이니셜라이세이션 세그먼트에 대한 부가정보를 기술한 실제 MPD를 보여준다. 본 MPD에게 가장 주목해야하는 부분은 절대주소(Absolute Address)를 통해 데이터를 요청하는 스태틱 페이지 기반의 DASH 서비스는 서버에 준비되어 있는 각각의 서비스 파일의 접근 주소를 절대주소로 기술한다는 것이다[1][2][5].

2.2 Smooth Streaming(Microsoft)

2.2.1 Smooth Streaming 소개

Smooth Streaming은 Microsoft에서 제공하고 있는 HTTP 적응형 스트리밍 서비스로 기본적인 시스템의 개념은 DASH와 유사한 구조를 가진다. 하지만, DASH와 같이 각각의 일련의 미디어 시퀀스를 미디어 세그먼트를 이용하여 개개의 단일파일로 저장하는 형태가 아닌, 일련의 미디어 시퀀스를 단일 파일로 통합하여 저장 한다. 이러한 파일 저장구조는 서비스 제공자에게 손쉽게 콘텐츠 관리를 할

표 3. MPD 기술의 예제
Table 3. Example of describing MPD

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
type="OnDemand"
minBufferTime="PT3.99S"
mediaPresentationDuration="PT3M49.71S"
xsi:schemaLocation="urn:3GPP;NS:PSS:AdaptiveHTTPStreamingMPD:2009
3GPP-MPD-r1.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
<ProgramInformation moreInformationURL="sky">
<Title>sky</Title>
<Source>sky</Source>
<Copyright>sky</Copyright>
</ProgramInformation>
<Period start="PT0S">
<Representation mimeType="video/3gpp" bandwidth="512000">
<SegmentInfo duration="PT3.99S" baseURL="http://Server/">
<InitialisationSegmentURL sourceURL="Sky-HD_512_init.3gp"/>
<Url sourceURL="Sky-HD_512_1.3gs"/>
<Url sourceURL="Sky-HD_512_2.3gs"/>
<Url sourceURL="Sky-HD_512_3.3gs"/>
</SegmentInfo>
</Representation>
<Representation mimeType="video/3gpp" bandwidth="1024000">
<SegmentInfo duration="PT3.99S" baseURL="http://Server/">
<InitialisationSegmentURL sourceURL="Sky-HD_1024_init.3gp"/>
<Url sourceURL="Sky-HD_1024_1.3gs"/>
<Url sourceURL="Sky-HD_1024_2.3gs"/>
<Url sourceURL="Sky-HD_1024_3.3gs"/>
</SegmentInfo>
</Representation>
</Period>
</MPD>
```

수 있는 방법을 제공하지만, 서버가 추가적인 동작을 통해 파일에 저장되어있는 미디어 시퀀스를 추출하는 서버의 추가동작이 필요하다. 따라서, 이를 위해 Smooth Streaming 서비스는 클라이언트와 서버간의 상호 약속된 메시지를 통해 서버에 추가적인 기능을 요청하는 다이나믹 페이지(Dynamic Page)를 통해 서비스를 제공한다. 또한, 상호간의 약속된 메시지에 포함되어야 하는 데이터를 제공하기 위해 인덱스 파일인 Manifest 파일을 통해 파일에 포함되어 있는 미디어 시퀀스를 요청하기 위한 부가정보를 클라이언트에게 제공한다[3][5].

표 4. Smooth Streaming 서비스 파일 형태
Table 4. Smooth Streaming service file type

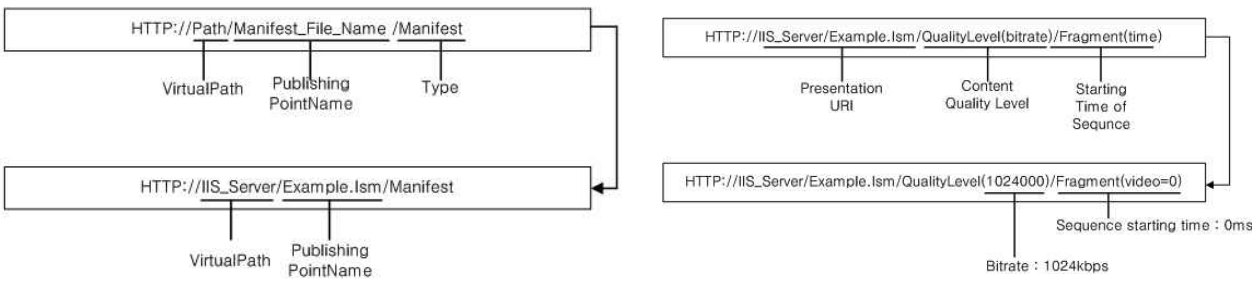
구분	특징
*.ism	서버 저장되어 있는 콘텐츠 관리 파일
*.ismc(manifest)	미디어 시퀀스 요청의 위한 부가정보 제공 파일
*.ismv(media file)	다수의 미디어 시퀀스를 단일파일로 저장한 파일

표 4는 앞서 설명한 Manifest 파일과 같이 Smooth Streaming 서비스에서 제공하고 있는 서비스 파일의 종류 및 특징을 보여주는 표로, 표에 나타난 바와 같이 '*.ismc'는 서비스를 위한 부가정보를 제공하는 Manifest의 확장자이며, '*.ismv'는 다수의 미디어 시퀀스를 통합하여 파일로 저장하는 미디어 파일의 확장자인 것을 볼 수 있다[3]. 표 4에서 가장 주목해야 하는 부분은 DASH 서비스에서는, 미디어 시퀀스를 복호화 하기 위한 정보를 제공하기 위해 이니셜라이세이션 세그먼트를 통해 초기화 정보를 클라이언트에 제공했다. 하지만, Smooth Streaming에서 제공하고 있는 서비스 파일을 보여주는 표 4에서는 미디어 시퀀스의 복호화를 위한 초기화 정보를 제공하는 서비스 파일이 없는 것을 볼 수 있다. 이러한 이유는 Smooth Streaming에서는 미디어 시퀀스를 복호화하기 위한 초기화 정보를 DASH처럼 파일로 제공하지 않는 대신, Manifest 파일에 초기화 정보를 직접 기술해서 제공하기 때문이다.

2.2.2 메시지 요청 형태

다이나믹 페이지를 기반으로 서비스를 제공하는 Smooth Streaming은 그림 7과 같이, 서버와 클라이언트의 상호 약속된 메시지를 통해 서버에 저장되어 있는 Manifest 파일 및 미디어 파일에 포함되어 있는 미디어 시퀀스를 서버에 요청한다.

그림 7은 이러한 서비스 파일을 요청하기 위해 Smooth Streaming에서 사용되고 있는 메시지를 보여 주는 그림이다. 그림 7(a)는 Manifest 요청하기 위한 메시지 형식으로 여기서 'Path'는 서버에 접속하기 위한 최상위 주소를 의미하며, 'Manifest_File_Name'은 파일관리 및 식별을 위해 서버에 저장되어 있는 '*.ism' 파일의 이름을 의미한다. 마지막으로 'Manifest' 필드는 해당 메시지가 Manifest파일을 요청하는 것을 정의하는 필드이다. 그림 7(b)은 미디어 파일에 포함되어 있는 미디어 시퀀스를 요청하기 위한 메시지 형식으로, 이 메시지에서 'IIS_Server'는 서버에 접속하기 위한 최상위 주소를 의미하며, 'Example_Ism'은 파일관리를 위한 서버에 저장되어 있는 미디어 파일을 관리하는 파



(a) Manifest 요청 메시지 형식 (b) 미디스 시퀀스 요청 메시지 형식

그림 7. Smooth Streaming 서비스 파일별 요청 메시지 형식
Fig. 7. Request message types for each service file of Smooth Streaming

일의 이름을 의미한다. 미디어 시퀀스 요청 메시지에서 이 두 필드는 ‘Presentation URI’로 지칭되며, 서버에 준비되어 있는 다양한 퀄리티의 미디어 파일들을 최상위 레벨에서 구분하기 위하여 사용된다. 다음 필드는 'QualityLevel' 과 'Fragment'로서, 각각의 필드는 'bitrate', 'time' 인수를 가지고 있다. 여기서 'bitrate'는 서버에 준비되어 있는 미디어 파일의 퀄리티를 선택 할 수 있는 인수이며, 'time' 은 요청하는 미디어 시퀀스의 시작 시간 값의 나타내는 인수이다[4].

2.2.3 Index File : Manifest

Smooth Streaming은 클라이언트에게 다양한 퀄리티의 미디어 파일들에 포함되어 있는 각각의 미디어 시퀀스들을 요청하기 위한 부가정보를 제공하는 파일로 Manifest 파일을 정의하고 있다. Manifest 파일은 다이나믹 페이지를 통해 서비스를 제공하는 Smooth Streaming 서비스를 위해 필수적으로 요구되는 메시지 요청 방법 및 요청을 위해 필요한 추가 정보, 미디어 시퀀스의 초기화 정보 등을 제공한다. 다음 표 5는 이러한 정보를 Manifest에 기술하기 위해 정의된 주요 Element 및 Attribute를 보여준다.

Smooth Streaming 서비스는 표 5에 정의되어 있는 Element 및 Attribute를 통하여 클라이언트에게 서로 다른 퀄리티의 미디어 시퀀스에 대한 속성 정보 및 기타 서비스를 위한 부가정보를 Manifest 파일에 기술하여 클라이언트에게 제공한다. 실례로, 다음 표 6은 표 5의 Element와 Attribute를 사용해서, 10Mbps와 3Mbps의 퀄리티로 Smooth Streaming을 제공하기 위해 사용된 실제 Manifest 파일을 보여준다. 본 표에서 볼 수 있듯이, 'StreamIndex'의 'Qualtiy Levels'를 통해 2개의 퀄리티의 미디어 파일이 있다는 것을 알 수 있으며, 'Chunk'를 통하여 서버에 준비되어 있는 각각의 미디어 파일이 115개의 미디어 시퀀스를 포함하고 있다

표 5. Manifest Element & Attribute
Table 5. Manifest's Element & Attribute

Element(E) or Attribute(A)	Description
SmoothStreamingMedia(E)	최상위 Element
MajorVersion(A)	데이터 정의를 위한 상위 버전 속성 (Default : 2)
MinorVersion(A)	데이터 정의를 위한 하위 버전 속성 (Default : 1)
Duration(A)	해당 미디어 콘텐츠의 총 재생 시간
StreamIndex(E)(E)	각각의 미디어 시퀀스의 속성 정보 정의
Type	미디어 시퀀스의 데이터 속성 ("video", "audio", "etc")
Chunks	미디어 시퀀스의 총 수
QualityLevels	클라이언트가 선택할 수 있는 퀄리티의 레벨의 총 수
MaxWidth	최대 가로 해상도
MaxHeight	최대 세로 해상도
Url	미디어 시퀀스를 요청하는 형식 기술
QualtiyLevel(E)(E)(E)	퀄리티 별 미디어 파일에 대한 세부 정보 기술
Index	Index
Bitrate	인코딩 레벨 (Bitrate)
FourCC	압축 형식
MaxWidth	최고 가로 해상도
MaxHeight	최고 세로 해상도
CodecPrivateData	미디어 시퀀스를 복호화 하기 위한 디코더 초기화 정보
c(E)(E)(E)	미디어 시퀀스 요청을 위한 시간 데이터 기술
n	Index
d	각각의 미디어 시퀀스에 대한 Duration

표 6. Manifest 기술 예제
Table 6. Example of describing Manifest

```
<?xml version="1.0" encoding="utf-16"?>
<SmoothStreamingMedia
  MajorVersion="2" MinorVersion="0" Duration="2297161411">
  <StreamIndex
    Type="video"
    Chunks="115"
    QualityLevels="2"
    MaxWidth="1920"
    MaxHeight="1080"
    Url="QualityLevels({bitrate})/Fragments(video={start time})">
    <QualityLevel
      Index="0"
      Bitrate="10240000"
      FourCC="H264"
      MaxWidth="1920"
      MaxHeight="1080"
CodecPrivateData="0000000016742C028965403C0112F2A0000000168CE3C80"
/>
    <QualityLevel
      Index="1"
      Bitrate="3072000"
      FourCC="H264"
      MaxWidth="1280"
      MaxHeight="720"
CodecPrivateData="0000000016742C01F965402802D8800000000168CE3C80" />

  <c
    n="0"
    d="26454200" />
  <c
    n="1"
    d="20020000" />
  <c
    n="2"
    d="20020000" />
      .
      .
  <c
    n="113"
    d="19969161" />
  <c
    n="114"
    d="15092970" />
  </StreamIndex>
</SmoothStreamingMedia>
```

는 것을 알 수 있다. 또한, 각 미디어 파일의 세부 정보를 제공하는 'QualityLevel'의 Attribute에 기술된 정보를 통해 미디어 파일의 비트레이트, 사용 코덱, 해상도, 미디어 시퀀스를 복호화 하기 위한 초기화 정보를 알 수 있다[3].

그림 8에서 나타난 바와 같이, 클라이언트는 서비스를 이용하기 위해서 표 6과 같이 기술된 Manifest 파일을 서버에 요청하여 서비스를 위한 부가정보를 습득하고, 이를 기반으로 자신의 전송채널에 해당하는 미디어 시퀀스를 서버에

실시간으로 요청하여 콘텐츠를 소비한다.

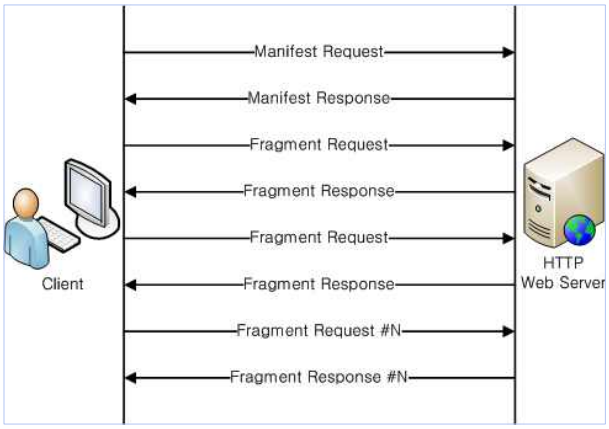


그림 8. Smooth Streaming 서비스 흐름도
Fig. 8. Service flow of Smooth Streaming

그림 9는 실제 Smooth Streaming 서비스를 통해 클라이언트가 콘텐츠를 렌더링(Rendering)하는 화면을 보여준다.

3. HTTP Adaptive Streaming 위한 전송 및 파일 저장 포맷

HTTP 적응형 스트리밍은 미디어 시퀀스의 전송 및 저장을 위한 포맷(Format)으로 현재 가장 널리 사용되고 있는 ISO Base Media File Format에 정의 되어 있는 프레그먼트(Fragment) 파일 구조를 통해 일련의 GOP(Group of Pictures)를 시간단위로 확장하여 저장하는 구조를 가진다[6]. 이러한 구조는 다양한 비트레이트(bitrate)로 인코딩 되는 미디어 시퀀스들을 시간 단위로 구분 할 수 있는 방법을 제공함으로써, 동일한 영상 데이터를 포함하는 서로 다른 비트레이트의 미디어 시퀀스간의 일관된 참조 방법을 제공한다.

그림 10은 프레그먼트가 일련의 GOP를 저장하는 구조를 나타내는 그림으로, 본 구조의 'mdat' box는 일련의 GOP를 연속적인 시간단위로 저장하며, 'moof' box는 'mdat' box에 저장되어 있는 영상 데이터의 접근정보와 같은 부가정보를 제공한다. 본 구조와 같이 GOP를 시간단위로 확장하

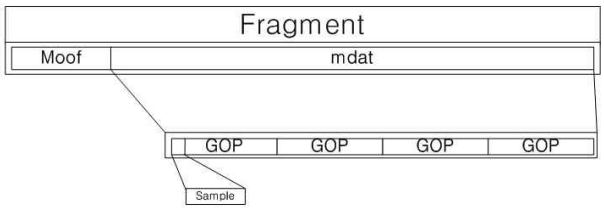


그림 10. 프레그먼트 GOP 저장 구조
Fig. 10. Storage structure of fragment for containing GOP

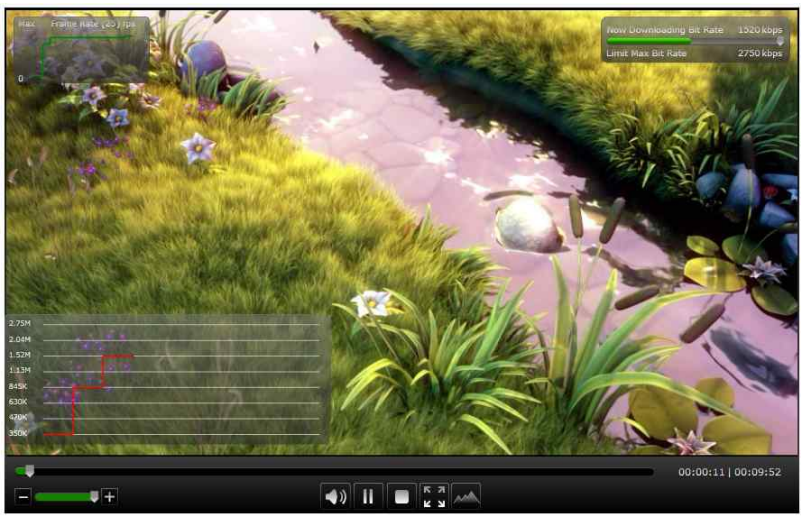


그림 9. IIS Smooth Streaming 서비스
Fig. 9. IIS Smooth Streaming service

여 저장하는 형태는 HTTP 적응형 스트리밍 서비스에서 서로 다른 비트레이트로 부호화된 미디어 시퀀스들을 동일한 시간 값을 통해 참조 할 수 있는 방법을 제공함으로써, 변화하는 전송채널 따라 미디어 시퀀스를 요청해야 하는 HTTP 적응형 스트리밍 서비스를 위한 기본 메카니즘을 제공한다.

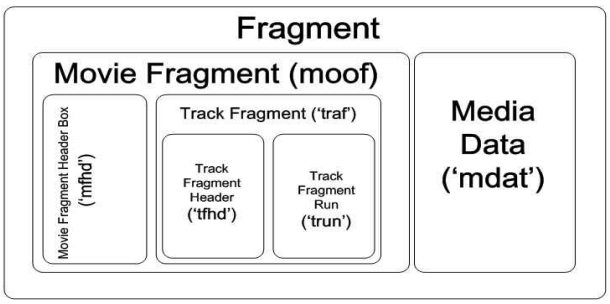


그림 11. 프레그먼트 구조
Fig. 11. Fragment structure

그림 11은 상기 설명한 GOP를 시간단위로 확장하여 저장하는 프레그먼트의 세부 구조를 보여주는 그림으로, HTTP 적응형 스트리밍 서비스에서는 본 구조를 사용해 미디어 시퀀스를 저장한다. 본 그림의 구조는 미디어 데이터에 대한 정보를 제공하는 무비 프레그먼트 박스('moof')와 실제 영상 데이터를 담고 있는 미디어 데이터 박스('mdat')로 나누어지며, 본 구조를 프레그먼트(Fragment)로 지칭한다. 'moof' box의 하위 박스로는 각 프레스먼트의 순서 정보를 제공하는 'mhfd' box와 'mdat' box 저장되어 있는 각 트랙('track', ex. Video, Audio, and etc)의 속성 및 샘플(Sample)의 접근 정보 등을 제공하는 'traf' box로 구분된다. 'traf'의 하위 박스로는 해당 트랙의 샘플 접근방법이나 기본 시간 정보 등을 기술하는 'tfhd' box와 'mdat' box에 시간단위로 확장 되어 저장된 미디어 시퀀스의 시간, 크기 등의 정보를 제공하는 'trun' box로 나누어진다. 이러한 일련의 GOP를 시간단위로 확장하여 저장하는 프레그먼트의 구조는 가변적인 전송환경에서 다양한 비트레이트의 미디어 시퀀스를 클라이언트가 자연스럽게 스위칭 할 수 있는 기본 메커니즘을 제공한다[3][6].

그림 12는 ISO Base Media File Format에서 권고하는 다수의 프레그먼트를 단일 파일로 저장하는 구조를 나타내는 그림으로, 본 구조는 미디어 콘텐츠의 파일의 타입을 명시하는 'ftyp'(File Type) box를 시작으로, 저장된 미디어 콘텐츠의 독립성을 유지하면서 파일에 포함되어 있는, 각각의 트랙('trak')에 대한 메타데이터를 제공하는 'moov'(Movie Metadata) box, 일련의 GOP를 시간단위로 그룹화한 미디어 데이터를 저장하는 프레그먼트, 각각의 프레그먼트의 접근을 위한 정보를 제공하는 'mfra'(Movie Fragment Random Access) box 구성되는 구조를 가지고 있다. 그림 12에서는 현재 두 개의 프레그먼트가 파일에 포함되어 있지만, 실제 서비스에서는 다수의 프레그먼트가 존재 할 수 있다. 이러한 프레그먼트를 단일 파일에 저장하는 구조는

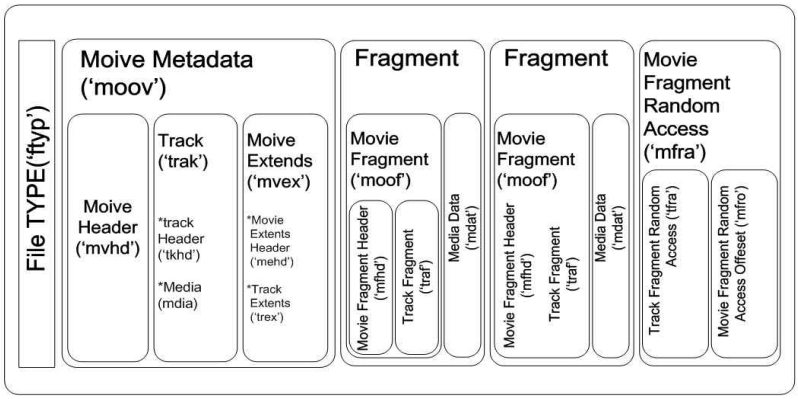


그림 12. 다수의 Fragment를 단일 파일에 저장하는 저장 구조
Fig. 12. The single storage structure for containing a number fragments

미디어 데이터의 구조적 독립성을 유지하면서, 쉽게 미디어 데이터를 접근 할 수 있는 방법을 제공한다. 본 구조는 Smooth Streaming 서비스에서 일련의 미디어 시퀀스를 단일 파일로 저장하기 위해 사용하는 미디어 파일의 구조로, 서버 관점에서 콘텐츠의 관리가 용이하다는 장점이 있다. 다음 표 7은, ISO Base Media File Format의 프레그먼트에 정의된 box의 형태를 보여준다[3][6].

표 7. 프레그먼트 박스 타입
Table 7. Fragment box type

moof			movie fragment
	mfhd		movie fragment header
	traf		track fragment
		tfhd	track fragment header
		trun	track fragment run
		sdtp	independent and disposable samples
		sbgp	sample-to-group
		subs	sub-sample information
mfra			movie fragment random access
	tfra		track fragment random access
	mfro		movie fragment random access offset
mdat			media data container

III. 유저 프로파일 이용한 단말간의 연속된 콘텐츠 소비환경 제공 방법

본 논문은, HTTP 스트리밍 환경에서 사용자가 단말기 변경하면서 동일한 콘텐츠를 소비하는 경우, 기존의 HTTP 적응형 서비스에서는 단말기 사이에 연속된 콘텐츠 소비환경을 적용할 수 없다는 문제점을 해결하기 위해, HTTP 적응형 스트리밍 환경에서 단말기 간의 연속된 콘텐츠 소비환경을 제공하기 위한 유저 프로파일 정의한다. 유저 프로파일은 단말간의 동일한 소비환경 제공하기 위한 정보를 포함하며, 본 정보는 표 8의 나타난 Element와 Attribute을 통해 유저 프로파일에 기술된다.

유저 프로파일은 표 8에서 나타난 바와 같이, 사용자의 유저 프로파일 접근 가능여부 식별하는 'Profile'과 사용자의 콘텐츠 소비정보를 기술하는 'UserContentData'로 구분되며, 아래 표 9와 같은 방법을 통하여 기술된다.

표 9. User Profile 기술 방법
Table 9. Method for describing User Profile

```
<? xml version="1.0" encoding="UTF-8">
<UserProfile>
<Profile UserID="$userID" PinNumber="$password"/>
<UserContentData
        Title="$title"
        IndexURL="$URL"
        UserRequestTimeOrUrl="$Time or $URL"
        UserPlayStopTime="$Time"
</UserProfile>

$userID : 사용자 식별자
&$password : 사용자 비밀번호
$Title : 콘텐츠 제목
$time : 시간 값
$URL : URL
```

표 10은 앞에 설명한 작성방법을 통해 실제 유저 프로파일을 작성한 예로, 표에 나타난 바와 같이, 사용자의 유저

표 8. 유저 프로파일 Element & Attribute
Table 8. User Profile Element & Attribute

Element (root)	Element (subset)	Attribute	설명
UserProfile			
	Profile		사용자의 접근 정보를 제공하는 Element
		UserID	사용자 ID
		PinNumber	비밀 번호
	UserContentData		연속적인 콘텐츠 소비환경을 위한 정보를 제공하는 Element
		Title	재생 도중 종료 된 콘텐츠의 제목
		IndexURL	재생 도중 종료 된 콘텐츠의 인덱스 파일 요청 방법
		UserRequestTimeOrUrl	재생 도중 종료 된 시점의 영상의 샘플을 포함하고 있는 미디어 시퀀스의 요청 시간 값 또는 URL
		UserPlayStopTime	사용자가 콘텐츠를 종료 한 시간 값

표 10. 유저 프로파일 기술 예제
Table 10. Example of describing User Profile

```
<? xml version="1.0" encoding="UTF-8">
<UserProfile>
<Profile UserID="User" PinNumber="1234"/>
<UserContentData
 Title="Avarta"
 IndexURL="http://Server/example.ism/manifest"
 UserRequestTimeOrUrl="2000000"
 UserPlayStopTime="3000000"/>
</UserProfile>
```

프로파일 접근가능 여부를 식별하기 위한 정보를 기술하는 'Profile'의 Attribute을 통해 아이디(ID)와 비밀번호(Pin number)가 기술되어 있는 것을 볼 수 있으며, 콘텐츠의 소비 정보를 기술하기 위해 'UserContentData'의 Attribute을 이용하여 소비하던 콘텐츠 제목, 인덱스 파일 접근 방법, 콘텐츠 종료 시점을 영상이 포함되어 있는 미디어 시퀀스의 요청 시간 값, 콘텐츠의 종료 시간 값이 유저 프로파일에 기술된 것을 볼 수 있다.

본 논문에서는 제안한 유저 프로파일을 통해 단말간의 연속한 콘텐츠 소비환경을 제공하는 단계를 Smooth Stream-

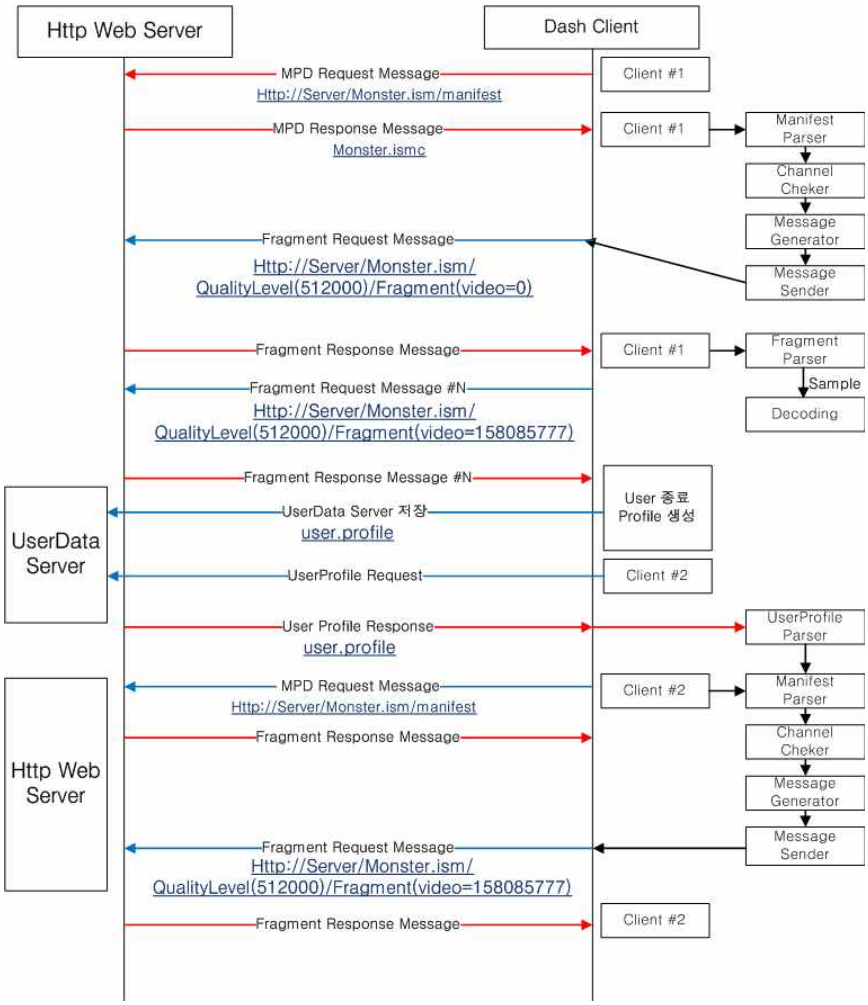


그림 13. 단말기 사이의 연속된 콘텐츠 소비 환경을 제공하기 위한 서비스 흐름도
Fig. 13. Service flow of offering the connected content consume environment between the terminals

ing 기반의 서비스를 통해 그림 13에서 보여준다. 본 그림에서 Client #1은 사용자가 처음 콘텐츠를 소비하기 위해 사용한 단말기를 의미하며, Client #2는 변경된 사용자의 단말기를 의미한다. Client #1은 최초 서비스를 위해 Manifest파일을 요청하고, 수신된 Manifest의 파싱(Parsing)을 통해 서버에 저장되어 있는 다양한 퀄리티의 미디어 시퀀스를 요청하기 위한 부가정보를 습득한다. 습득된 부가정보를 통해 Client #1은 자신의 가용 대역폭에 해당하는 미디어 시퀀스를 요청/수신하여 콘텐츠를 소비한다. 콘텐츠가 소비되던 도중 종료되면, 클라이언트는 유저 프로파일을 생성하고, 생성된 파일을 사용자의 다른 단말기와 공유하기 위하여 사용자 데이터 서버로 전송한다. 사용자가 클라이언트 #2로 단말기를 변경하고, 전 단말기(Client #1) 콘텐츠 종료 시점부터 이어서 소비하고자 하는 경우, 최초 서비스 시작 시 Client #2는 사용자 데이터 서버에 저장되어 있는 유저 프로파일을 요청한다. 클라이언트는 수신된 유저 프로파일 포함되어 있는 정보를 통해, Client #2는 전 단말에서 종료된 시점의 미디어 시퀀스부터 요청을 하여, 사용자의 추가 조작 없이 Client #1의 콘텐츠의 종료 시점부터 Client #2에서 이어서 콘텐츠 소비를 할 수 있다. 본 그림에서 주목해야 하는 점을 앞절에 설명한 DASH는 미디어 시퀀스를 복호화 하는 정보를 이니셜라이세이션 파일을 통해 하나의 서비스 파일로 제공하지만, Smooth Streaming에서는 초기화정보를 Manifest에 직접 기술하여 해당 정보를 제공한다. 이러한 서비스 방식은 DASH와 달리 추가적인 복호와 정보를 위해 서버에 해당 파일을 요청하지 않고 Manifest를 통하여 서버에 준비되어 있는 모든 미디어 파일을 복호화 할 수 있다는 장점이 있다.

Ⅳ. 실험 및 검증

본 논문에서는 HTTP 적응형 스트리밍 환경에서 콘텐츠를 소비하는 단말간의 소비환경 제공하는 방법에 대해 제안하였다. 그림 14는 제안한 방법을 실험하기 위한 전체 시스템 구조도로서, Microsoft의 Smooth Streaming 기반으로 하고 있다.

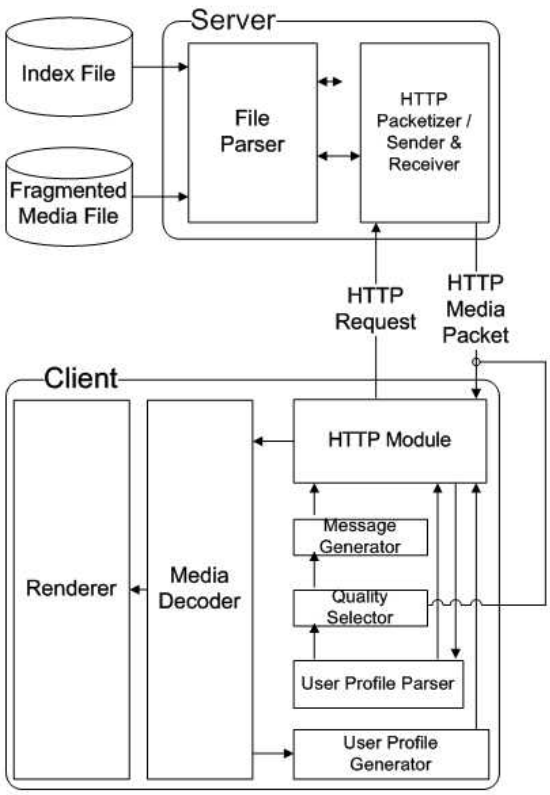


그림 14. 전체 시스템 구조도
Fig. 14. The whole system structure

본 실험에서는 유저 프로파일을 이용한 단말간의 연속된 콘텐츠 소비 여부를 확인하기 위하여Smooth Streaming 서

표 11. 클라이언트 플레이어 모듈 및 기능
Table 11. Client player's module & function

구 분	기 능
HTTP Module	미디어 시퀀스를 요청/수신하는 모듈
Message Generator	서비스 파일을 요청하기 위한 메시지를 생성하는 모듈
Quality Selector	클라이언트의 전송속도를 측정하는 모듈 (본 실험에서는 사용자가 직접 유저 인터페이스(User Interface)를 통해 전송속도를 설정 수 있도록 함)
User Profile Parser	유저 프로파일을 파싱(Parsing)하는 모듈
User Profile Generator	유저 프로파일을 생성하는 모듈
Media Decoder	수신된 미디어 시퀀스를 복호화 하는 모듈
Renderer	복호화 된 미디어 데이터를 화면에 디스플레이 하는 모듈

비스 환경에서 미디어 시퀀스를 재생할 수 있는 클라이언트 플레이어를 구현하여 실험을 진행하였다. 구현된 클라이언트 플레이어는 표 11에 나타난 바와 같이 총 7개의 모듈로 구성되어 있으며, 각 모듈의 기능은 표에 나타난 바와 같다.

그림 15는 구현된 HTTP 모듈을 통해 최초 Manifest 파일의 요청하는 그림으로, 여기서 빨간색으로 표시된 부분은 Manifest를 요청하기 위한 메시지로, 그림에서 나타난 바와 같이 HTTP에서 데이터 요청을 위해 사용되는 'Get' 메소드(Method)를 통해 Manifest 파일을 요청하는 것을 볼 수 있다.

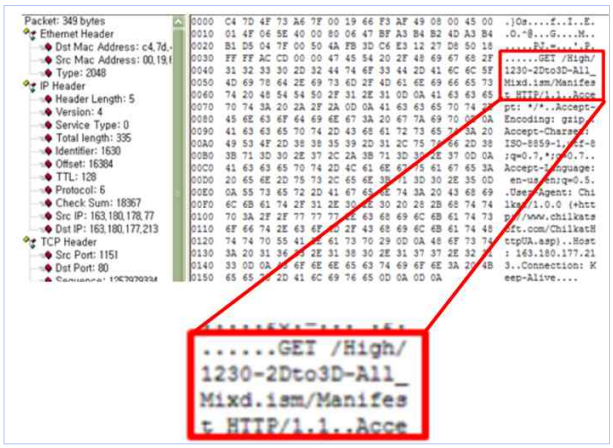


그림 15 Manifest 파일 요청
Fig. 15. Manifest file request

또한, 그림 15의 요청 메시지의 응답으로 그림 16에 나타난 바와 같이 Smooth Streaming에서 Manifest에 부가정보를 기술하기 위해 정의된 Element와 Attribute를 통해 기술된 Manifest 파일이 응답된 것을 그림에 빨간색으로 표시된 부분을 통해서 볼 수 있다.

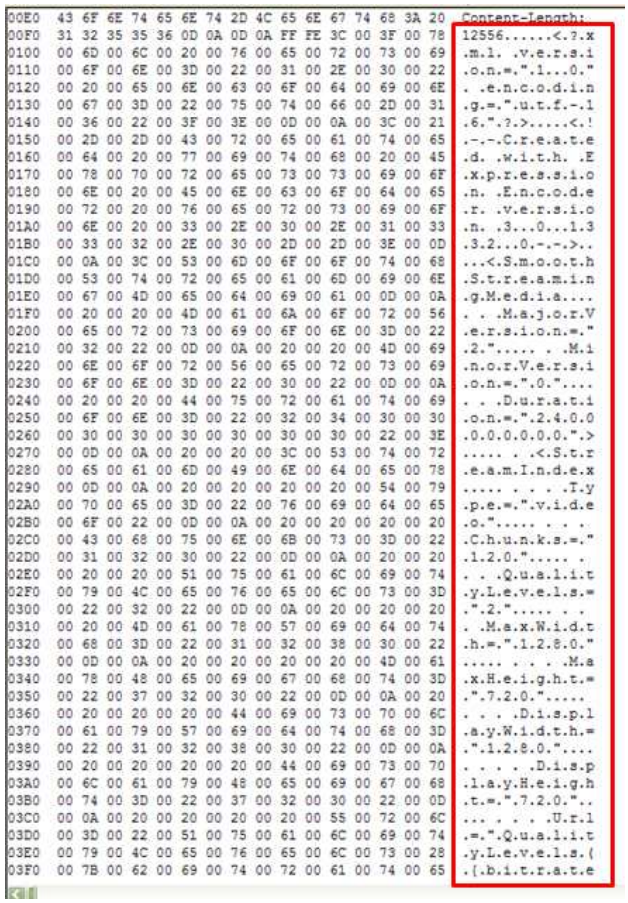


그림 16 수신된 Manifest 파일
Fig. 16. Received Manifest file

이러한 Manifest 요청과 이에 대한 응답으로 수신 된 Manifest에 기술되어 있는 정보를 이용하여, 클라이언트는 자신에 전송채널에 적합한 미디어 시퀀스를 그림 17의 메세지와 같이 서버에 요청한다. 여기서 주목해야 하는 부분은 앞서 설명한 바와 같이 다이나믹 페이지 기반의 서비스를 제공하는 Smooth Streaming에서는 클라이언트와 서버

그림 17 미디어 시퀀스 요청
Fig. 17. Media sequence request

간의 상호 정의 된 메시지와 데이터를 식별하기 위해 약속된 필드와 인수를 통해 데이터를 요청한다는 것이다.

그림 18은 그림 17의 미디어 시퀀스 요청 메시지에 대한 서버의 응답으로 클라이언트에게 수신된 미디어 시퀀스를 보여준다. 그림 에서 빨간색으로 표시된 부분에서 볼 수 있듯이, 'moof' box로 시작되는 프레그먼트의 구조를 가지는 미디어 시퀀스가 클라이언트에 수신된 것을 볼 수 있다.

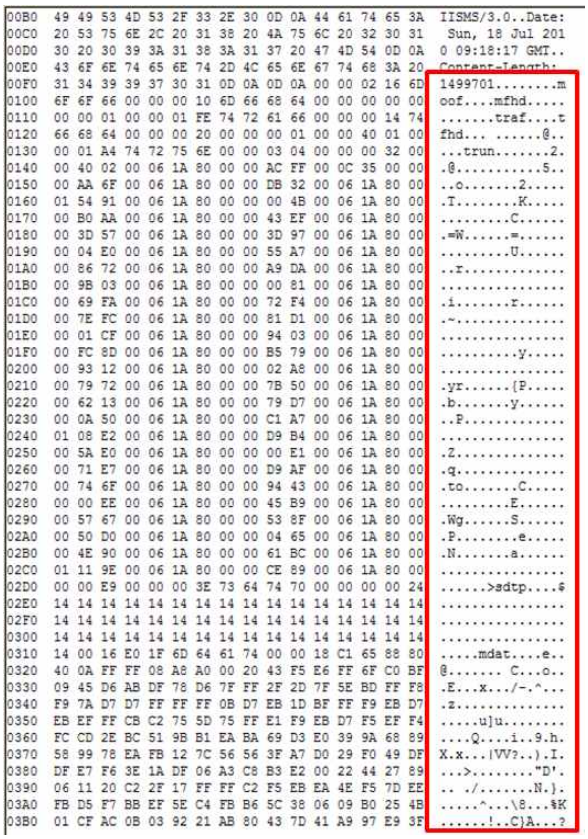
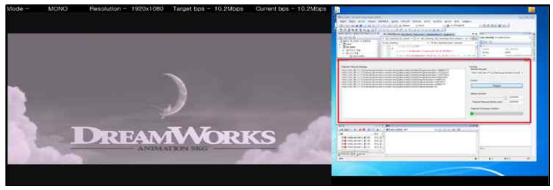


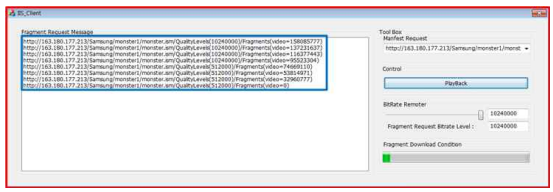
그림 18. 수신된 미디어 시퀀스
Fig. 18. Received media sequence

그림 19는 단말기 사이의 연속된 콘텐츠 소비를 확인하기 위한 실험에서, 사용자가 최초 단말기를 통해 콘텐츠를 소비하는 화면을 보여준다.

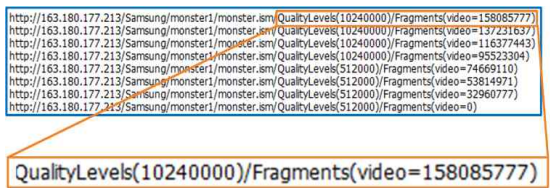
본 그림 19(a)는 Smooth Streaming 환경에서 최초 단말기를 통해 콘텐츠를 재생하는 화면을 보여주고 있으며, 19(b)는 클라이언트 플레이어의 유저 인터페이스를 보여주



(a) 최초 단말기에서 콘텐츠를 재생하는 화면



(b) 단말기 유저 인터페이스(User Interface)



(c) 최초 단말기의 콘텐츠 종료 시점의 미디어 시퀀스 요청 메세지

그림 19. 최초 단말을 통한 콘텐츠 재생 화면
Fig. 19. The screen playing the content through the first terminal

는 그림으로, 파랑색으로 표시된 부분은 클라이언트가 실시간으로 서버에게 보내는 미디어 시퀀스 요청 메시지를 유저 인터페이스를 통해 보여주는 그림이다. 19(c)는 19(b)의 미디어 시퀀스를 요청하는 메시지를 확대한 그림으로, 여기서 주황색으로 표시된 부분은 사용자가 콘텐츠를 종료한 시점의 미디어 프레그먼트 요청메시지를 보여준다. 본 실험에서, 클라이언트는 사용자가 소비하던 콘텐츠가 종료하면 자동으로 생성되는 유저 프로파일에 'time' 인수인 "158085777"을 저장 하고, 단말기 변경 시 이 시간 값에 해당하는 미디어 시퀀스부터 요청함으로써, 단말간의 연속된 콘텐츠 소비환경을 제공할 수 있다.

그림 20은 그림 19에서 사용자의 콘텐츠 소비 종료로 생성되어 사용자 데이터 서버에 저장된 유저프로파일을 변경된 단말기에서 다운로드 받아 파싱한 결과를 보여주는 그림으로, 본 그림에 나타난 바와 같이 Smooth Streaming 서비스 환경에서 단말간의 연속된 콘텐츠 소비 환경을 제공

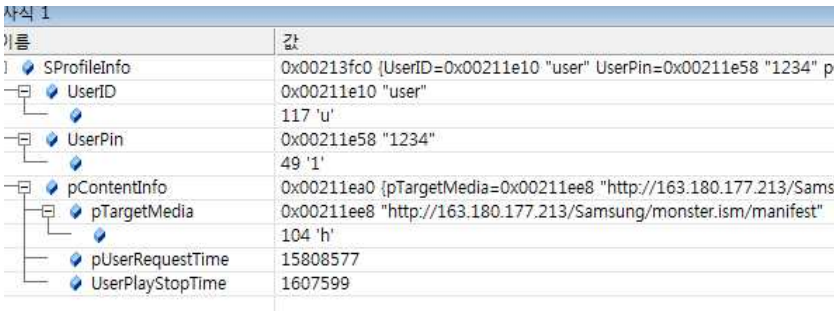
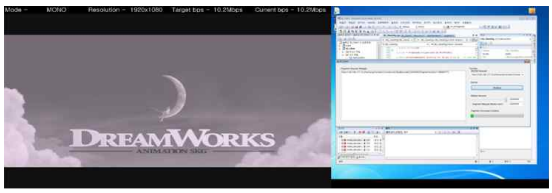


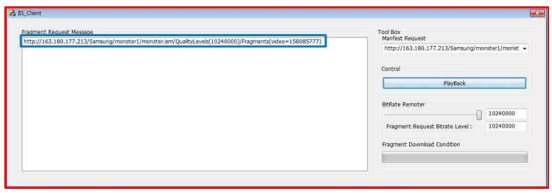
그림 20. 유저 프로파일 파싱(Parsing) 결과
Fig. 20. The result of parsing the User Profile

하기 위해 필요한 데이터인 Manifest의 접근 방법 "http://163.180.177.213/Samsung/monster.sim/manifest"과 사용자가 종료한 시점의 영상 데이터를 포함하고 있는 미디어 시퀀스를 요청하기 위한 시간 값 "15808577", 종료한 영상에 해당하는 종료시간 값 "1607599"가 유저 프로파일에 기술되어 있는 것을 볼 수 있다.

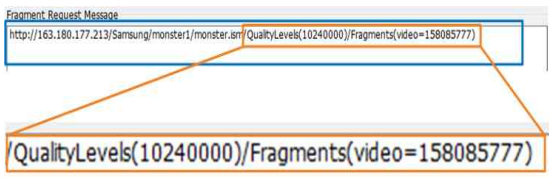
그림 21은 최초 단말에서 콘텐츠 소비 종료로 생성된 유저 프로파일을 변경된 단말기에 적용하여, 최초 단말의 콘텐츠 종료 시점부터, 변경된 된 단말에서 연속해서 콘텐츠를 소비하는 것을 보여준다.



(a) 최초 단말기의 콘텐츠 종료 시점부터, 변경된 단말기에서 연속해서 콘텐츠를 재생하는 화면



(b) 변경된 단말기의 유저 인터페이스(User Interface)



(c) 변경된 단말의 최초 미디어 시퀀스 요청 메세지

그림 21. 변경된 단말기에 유저 프로파일 적용 후, 콘텐츠 재생 화면
Fig. 21. The screen playing the content through changed terminal after applying User Profile

그림 21에서 가장 주목해야 하는 부분은 논문에서 제안한 유저 프로파일을 통해 단말기 변경 시, 클라이언트가 사용자 데이터 서버에 저장되어 있는 유저 프로파일을 요청/수신하고, 유저 프로파일에 포함되어 있는 정보를 변경된 단말기에 적용하여, 변경된 단말기에서 최초 단말기의 콘텐츠 종료 시점의 미디어 시퀀스부터 요청함으로써, 사용자가 직접 소비환경을 구성하지 않고, 변경된 단말기에서 연속해서 콘텐츠 소비한다는 것이다. 그림 21(c)에서 주황색으로 표시된 부분에서 볼 수 있듯이, 변경 전 단말기에서 마지막으로 요청한 미디어 시퀀스의 시간 값과 변경된 단말기에서 최초로 요청하는 미디어 시퀀스의 시간 값이 동일하다는 것을 볼 수 있다.

V. 결 론

과거 고정용 디스플레이 장비와 전용채널을 통해 이루어지던 콘텐츠 소비형태는 최근 방송시스템의 발달과 다양한 기능을 제공하는 고사양의 휴대용 단말기의 등장으로 그 소비형태가 변화하고 있다. 과거 제한된 시간, 장소에서만 콘텐츠를 소비 할 수 있었던 사용자는 언제, 어디서나 자신의 선호하는 콘텐츠를 IP망에 접속할 수 있는 휴대용 단말기를 통해 선택하여 소비 할 수 있게 되었다. 또한, 유동적인 IP망에서도 사용자에 전송채널 적응적으로 미디어

시퀀스를 전송하는 HTTP 기반의 적응형 스트리밍 서비스가 제공되고 있으며, 이를 표준화하기 위해 대표적인 멀티미디어 단체인 MPEG에서는 HTTP 적응형 스트리밍의 규격에 대한 표준화가 활발히 진행 중이다. 이러한 소비형태에서, 사용자는 다수의 단말기를 보유하게 되었고, 상황에 따라 자신이 원하는 단말기를 변경하면서 콘텐츠를 소비하게 되었다. 하지만, 이러한 소비형태에서 사용자가 단말기를 변경하여 동일한 콘텐츠 소비하는 경우, 전 단말기와 동일한 소비환경을 유지하기 위해 사용자가 직접 변경된 단말기에 소비 환경을 적용해야 한다는 문제점 있다. 본 논문에서는 이러한 문제점을 착안하여, HTTP 적응형 스트리밍 서비스 기술과 제안된 단말기 간의 콘텐츠 소비 환경 유지 방법을 적용하여, 사용자의 단말기 간의 콘텐츠 소비환경 동일하게 유지하는 방법을 제안하였다. 논문에서 제안한 방법을 통해, 사용자가 보유한 다양한 이동 단말기간의 콘텐츠 소비환경 동일하게 유지하게 함으로써, 향후 멀티미디어의 서비스의 활용의 증대 및 콘텐츠의 소비증대를 예상할 수 있다.

참 고 문 헌

[1] 3GPP TS 26.234: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Transparent end-to-end Packet-switched Streaming Services(PSS); Protocols and codecs(Release 9).

[2] 3GPP TS 26.244: "Transparent end-to-end packet switched streaming service (PSS); 3GPP file format (3GP)".

[3] IIS Smooth Streaming Technical Overview: "Author Alex Zambelli, Media Technology Evangelist, Microsoft Corporation - March 2009"

[4] MS-SMTH IIS Smooth Streaming Transport Protocol : "v20090908 IIS Smooth Streaming Transport Protocol Copyright@2009 Microsoft Coperation, Release : Tuesday. Septempber 8, 2009"

[5] "Draft CD as submitted to Secretary of ISO/IEC JTC 1/SC 29 : Text of ISO/IEC 23001-6: Dynamic adaptive streaming over HTTP (DASH)", ISO/IEC JTC1/SC29/WG11, Guangzhou, China, Oct 2010

[6] "Information technology - Coding of audio-visual objects - Part 12 : ISO base media file format", ISO JTC 1/SC 29/WG 11/ MPEG2008/ N9678, Antaly, Turkey, Jan, 2008.

저 자 소 개

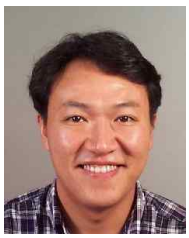
김 정 한

- 2009.2 ~ 현재 : 경희대학교 전자전파공학과 석사과정
- 주관심분야 : 멀티미디어 서비스, 디지털 방송

이 장 원

- 2007년 2월 : 경희대학교 전자공학과 졸업(공학사)
- 2007년 3월 ~ 현재 : 경희대학교 전자전파공학과 석박사과정
- 주관심분야 : 멀티미디어 서비스, 디지털 방송

저 자 소 개



김 규 헌

- 1989년 2월 : 한양대학교 전자공학과 공학사
- 1992년 9월 : 영국 University of Newcastle upon Tyne 전기전자공학과 석사
- 1996년 7월 : 영국 University of Newcastle upon Tyne 전기전자공학과 박사
- 1996년 8월 ~ 1997년 7월 : 영국 University of Sheffield, Research Fellow
- 1997년 9월 ~ 2006년 2월 : 한국전자통신연구원 대화형미디어 연구팀장
- 2006년 2월 ~ 현재 : 경희대학교 전자정보대학 부교수
- 주관심분야 : 영상처리, 멀티미디어통신, 디지털 대화형 방송

서 덕 영

- 1980년 2월 : 서울대학교 원자핵공학과 학사
- 1985년 6월 : Georgia Tech 핵공학과 석사
- 1990년 6월 : Georgia Tech 전자공학과 박사
- 1990년 9월 ~ 1992년 3월 : 상공부 생산기술연구원 HDTV 연구개발단 선임연구원
- 2002년 2월 ~ 2003년 2월 : 미국 North Carolina State Univ. 방문교수
- 1992년 3월 ~ 현재 : 경희대학교 전자공학과 교수
- 주관심분야 : Networked Video