

특집논문-00-5-2-04

MPEG-4 Over MPEG-2 TS로부터 MP4 파일로의 포맷 변환기 설계

최재영*, 정제창*

Design of a Format Converter from MPEG-4 Over MPEG-2 TS to MP4

Jaeyoung Choi* and Jechang Jeong*

요약

본 논문에서는 MPEG-2 시스템 층의 하나인 방송 및 전송을 위한 트랜스포트 스트림(TS)상에 MPEG-4 데이터를 구조화하여 MPEG-4 Over MPEG-2 TS 비트 스트림을 만드는 방법과 이를 저장 매체 포맷 중 하나인 MP4 파일로 변환하는 방법에 관한 연구이다. MPEG-4는 객체 단위의 부호화 비트 스트림으로 구성되기 때문에 이를 객체의 속성을 표현하는 객체 기술자, 객체들간의 시공간 관계를 표현하는 장면 기술자가 필요하며 또한 모든 객체들간의 복호화 정보와 객체간의 동기화를 위해 다양한 여러 가지 기술자들이 필요한데 이런 다양한 MPEG-4 비트 스트림을 어떻게 MPEG-2 TS 규격에 맞게 전송하는가에 초점을 두었다. 또한 설계한 MPEG-4 Over MPEG-2 TS 비트 스트림을 저장매체를 대상으로 하는 새로운 파일 규격인 MP4 파일로 변환하는 알고리듬을 제시하고 구현 방법을 소개한다.

Abstract

MPEG-4 is a digital bit stream format and associated protocols for representing multimedia content consisting of natural and synthetic audio, video and object data.

This paper describes an application where multiple audio/visual data stream are combined in MPEG-4 and transported via MPEG-2 transport streams(TS). Also, this paper describes how to convert MPEG-4 Over MPEG-2 TS bit streams into MP4 file which is designed to contain the media information of an MPEG-4 presentation in a flexible, extensible format. MPEG-4 is presented in the form of audio-visual objects that are arranged into an audio-visual scene by means of a scene descriptor and is composed of the audio-visual objects by means of an object descriptor. These descriptor streams are not defined MPEG-2 TS. So, this paper focuses on handling of these descriptors and parsing TS streams to get MPEG-4 data.

The MPEG-4 Over MPEG-2 TS to MP4 format converter is implemented in the demonstrated systems.

I. 서론

디지털 TV의 장점은 무엇보다도 고화질의 영상 서비스와 다양한 멀티미디어, 그리고 양방향 서비스를 가능하게 한다는 점이다. 이러한 디지털화 경향은 TV 방송에만 국

한된 것이 아니라 모든 방송, 통신, 가전 분야에 공통된 것이다. 또한, 콘텐츠가 디지털 영역에서 처리됨에 따라 다양한 형태의 멀티미디어 서비스가 새로이 출현하고 있으며, 이러한 디지털 콘텐츠의 전달 수단 또한 디지털 저장 매체, 디지털 유선 채널, 디지털 지상파 채널, 디지털 위성 채널 등으로 다양화되고 있다.

위의 변화로 말미암아 다양한 멀티미디어를 포괄하는 MPEG-4 기술이 등장하게 되었는데 텍스트, 영상, 음성, 3차원 합성 영상, MIDI 등 예전에 볼 수 없었던 다매체를 통합

* 한양대학교 전자통신공학과
Department of Electronic Communications Engineering, Hanyang University

※ 본 연구는 한국전자부품연구원(KETI)의 지원에 의해 수행되었음.

한 기술이 바로 MPEG-4(ISO/IEC 14496)이다^[1]. MPEG-4의 특징을 살린 각종 애플리케이션이 가능한데 저 비트율과 에러 내성 기능을 이용한 TV전화 등의 실시간 통신, 객체기반의 코딩 특성을 이용한 휴대정보 단말기와 같은 이동 멀티미디어 기기, 인터넷 방송 등 그 응용은 무궁무진하다. 그 중 MPEG-4 멀티미디어 서비스가 디지털 TV에 응용될 수 있는 분야 중에는 전송 및 방송을 위한 MPEG-2 트랜스포트 스트림 상에 MPEG-4 데이터를 실어보내 다양한 부가서비스로 그 장점을 극대화시키는 분야를 들 수 있다. 예를 들면 야구경기를 보다가 특정 선수의 기록을 보고 싶다면 간단한 마우스의 조작으로 그에 대한 세부 정보를 볼 수 있고 드라마를 보다가 마음에 드는 소품을 선택해서 바로 구매할 수도 있다. 이러한 부가 정보를 MPEG-4로 구현하는 것이다. 이러한 MPEG-4 기술로 인해 보다 능동적인 사용자와의 상호작용(User Interaction)이 가능하게 되었다^[2].

본 논문의 구성은 다음과 같다. II장에서는 MPEG-4 비트 스트림에 대해 살펴보고 III장에서는 MPEG-4 데이터를 실어 보낼 방송용 채널인 MPEG-2 트랜스포트 스트림에 대해 설명하였다. IV장에서는 비트 스트림을 저장한 로컬 저장매체중 하나인 MP4 파일에 대해 설명하였다. V장에서는 MPEG-4의 다양한 부호화 비트 스트림을 각각 어떻게 구조화하여 MPEG-4 Over MPEG-2 TS 비트 스트림을 만들 것인가에 대해 설명하였고 또한 MPEG-4 Over MPEG-2 TS 비트 스트림을 MP4 파일로 변환하는 알고리듬 및 이의 설계에 대해 설명한다. 그리고 구현결과를 제시한다. 마지막으로 VI장에서는 결론을 맺는다.

II. MPEG-4 비트 스트림

MPEG-4는 다양한 객체를 통합한 멀티미디어 부호화 표준으로 기존의 MPEG-1, 2와는 다르다. MPEG-4 시스템은 장면기술과 다중화의 2계층으로 구성되고 이 다중화 층 아래에 기존의 전송방식과의 인터페이스를 규정하는 층이 있다. 이것을 MPEG-4 시스템에서는 DMIF(Delivery Multimedia Integration Framework)라고 부른다^[3]. 이러한 여러 객체의 부호화 비트 스트림의 다중화 및 동기뿐만 아니라, 장면기술에 따른 합성을 취급하는 것이 MPEG-1, 2와는 다른 점이다. MPEG-4 표준 문서에도 나와 있는 그림 1은 위에서 설명한 MPEG-4 시스템의 구조를 보여주고 있다^[1]. 다중화 계층(TransMux Layer)에는 MPEG-2 TS, UDP/IP, ATM, PSTN망 등 다양한 전송 매체로 전달되어온 MPEG-4 데이터는 DMIF와의 인터

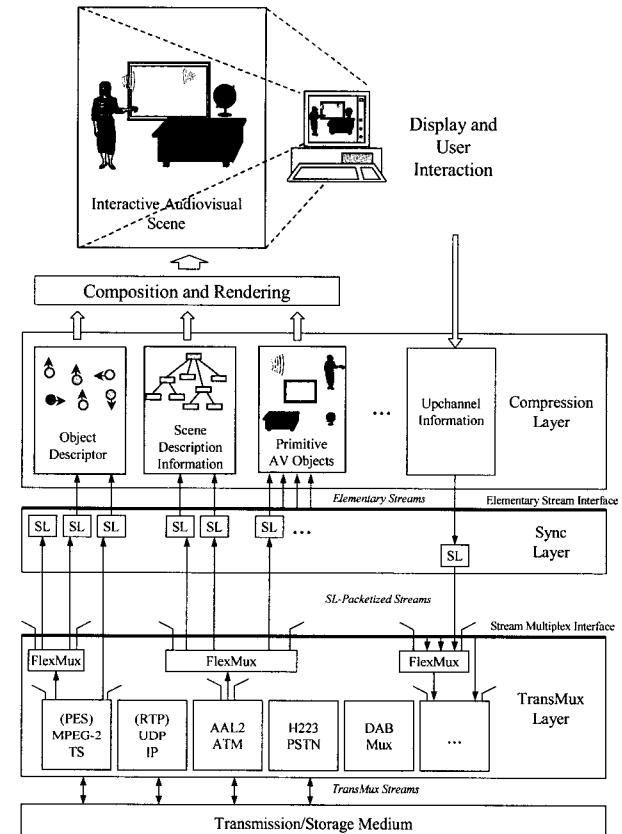


그림 1. MPEG-4 시스템 계층도
Fig. 1. MPEG-4 terminal architecture

페이스를 거쳐 미디어간의 동기화를 담당하는 동기 계층(Sync Layer)에 전달된다.

동기 계층에서는 각 스트림 별로 동기화 정보가 들어간 SL 패킷(SL packet)별로 나누어지고 기초 스트림 인터페이스(Elementary Stream Interface)는 각 객체별로 압축 계층(Compression Layer)에 전달하는 역할을 담당하고 있다. 압축 계층에 전달된 각각의 객체들, 즉 객체의 시공간 관계를 기술한 장면기술자, 장면기술과 미디어 객체의 속성을 기술한 객체기술자, 비주얼 및 오디오 데이터들이 복호화 된다. 복호화된 스트림들은 장면기술자에 의해서 합성(Composition)되고 프리젠테이션이 되며, 이러한 기술자들에 의하여 사용자와의 상호작용이 가능하게 된다.

1. 장면 기술(Scene Description)

MPEG-1, 2에서는 정형화된 비디오 부호화만을 취급해 왔기 때문에 장면이라는 개념이 없었다. 그러나 MPEG-4

에서는 임의의 객체를 부호화 할 수 있기 때문에 장면을 기술하는 요소가 필요하다. 여기서 장면이란 한 화면으로 보면 되는데 이 한 화면 안에는 정지영상, 텍스트, 동영상, 오디오 그 밖의 여러 미디어 객체가 포함되어 있다. 그래서 이러한 객체들 간의 공간적 위치, 시간적인 관계를 표현하는 장면 기술자(Scene Descriptor)가 필요하다. 이를 MPEG-4에서는 BIFS(Binary Format for Scene)로 규격화하고 있다.

BIFS는 VRML을 기초로 하고 있다^[4]. BIFS와 VRML은 둘 다 계층구조로 정렬된 노드들의 집합으로 이루어져 있다. 각각의 노드들은 장면(Scene)의 객체를 표현하고 집합으로 한데 묶고 또는 변형시킨다. 이러한 노드들은 노드의 특별한 동작을 정의하는 필드(Fields)로 구성된다. 또한 'DEF'와 'USE'를 사용하여 노드들을 재 사용하여 비트 발생률을 줄이고 있고 또한 장면 구성요소들(Scene elements) 사이에 이벤트를 전파시키기 위해 'ROUTE'를 이용한다.

(1) 장면 간신

MPEG-4는 방송용 애플리케이션, 일대일 통신 애플리케이션에 이용할 수 있도록 고안되었기 때문에 이런 충족 요건을 만족시키기 위해서는 애플리케이션 자체가 일시적 스트림으로 간주될 수 있어야 한다는 것이다. 이것의 의미는 BIFS 프리젠테이션이 시간에 따라 변할 수 있음을 뜻 한다. 이를 실현하기 위해서 장면을 변화시킬 수 있는 기본 스트림을 구조화해서 보내는 규격이 존재하는데 이를 BIFS Command Frame이라 칭한다. 이는 장면을 구성하는 노드 및 노드의 구성요소, 이를테면 필드 값, ROUTE들을 간신하는 것이다.

BIFS Command Frame은 작은 애니메이션에 적합하다. 반면 애니메이션이 연속적으로 일어나거나 좀더 효율적인 압축이 필요하다면 또 다른 방법인 BIFS Animation Frame을 이용한다. BIFS Command Frame은 주로 정적인 장면의 간신인데 반하여 BIFS Animation Frame은 비디오 및 오디오 스트림처럼 스트리밍 데이터로서 전송하고 있다는 것을 가정하고 있는 점이 다르다. 방송 애플리케이션에서 BIFS Command Frame은 동일 내용이 전송되는 것을 생각할 수 있는데, BIFS Animation Frame은 중복하여 전송하지 않는 것이 일반적이다.

또 위에서 기술한 장면의 간신은 사용자 조작에 의한 노드나 필드 값의 변화를 의미하지 않는다. 예를 들면 사용자가 어떤 객체를 마우스로 클릭하여 이동시킨 경우 좌표를 지정하는 'Transform' 노드의 'translation' 필드 값이

변화하지만, 이것은 미리 장면에 포함시킨 동작이다. BIFS Command Frame이나 BIFS Animation Frame은 콘텐츠 작성자의 의도대로 장면을 간신하기 위하여 사용된다^[2].

(2) 프로파일(Profile)

애플리케이션에 따라서는 반드시 필요로 하지 않는 노드와 기능이 있다. 예를 들면, 오디오만을 사용하는 애플리케이션에 있어서, 화면 변화에 관련된 노드까지 포함하는 것은 합리적이지 않다. 그래서 MPEG-4의 다른 부분과 마찬가지로, 장면기술에 대해서도 프로파일의 개념이 도입되었다. 프로파일은 이후 개선될 가능성도 있지만, 현재는 표 1과 같이 설정되어 있다. 표의 노드란에서 사용되는 오디오 및 비디오와 관련된 노드는 기술되어 있지 않다. 이들 노드는 MPEG-4의 오디오와 비디오 부분에서 정의되는 프로파일에 의존한다. 단순 프로파일은 구형형상의 비디오와 오디오를 다루는 MPEG-2와 마찬가지로 단순한 장면을 취급하는 애플리케이션을 위한 것이다.

표 1 장면기술 프로파일(Profile)
Table. 1. Scene description profiles

프로파일명	노드	루트	BIFS Command	BIFS Animation
Simple	Layer 2D, Transform 2D	x	o	x
2D	Layer 2D, 2D Node	o	o	o
VRML	VRML Node	o	o	o
Audio	Audio Node	o	o	x
Complete	All	o	o	o

2. 객체 기술자(Object Descriptor)

장면 기술자는 장면에서 미디어 객체들 간의 시공간적인 관계를 기술한다. 그런데 객체를 기술함에 있어서 오디오, 비디오 객체를 운송하는 기초 스트림을 직접 가리키는 것이 아니라 객체 기술자라는 개념을 사용한다. 즉 객체 기술자는 객체 속성 자체를 표시하는 기술자이다. 이 개념은 장면구조, 미디어 데이터, 그리고 전송간의 분리를 용이하게 하여, 다른 요소에 영향을 미치지 않고도 이들 요소의 변경을 쉽게 할 수 있게 한다. 객체 기술자 사용 목적은 장면 기술에서 사용되는 미디어 객체와 이와 관련된 기초 스트림을 확인하여 적절히 연결시켜 주기 위함이다. 이때 미디어 객체는 객체 기술자 ID라는 숫자 확인자를 사용하여 객체 기술자를 가리킨다. 그림 2는 객체 기술자

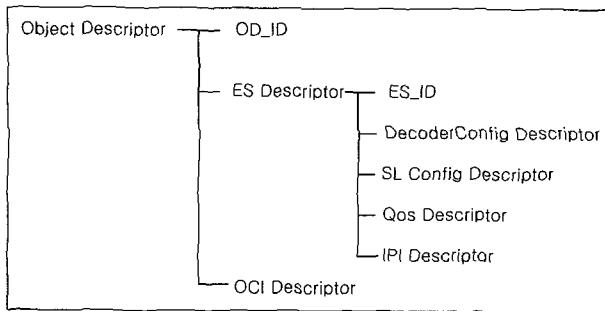


그림 2. 객체 기술자의 구성

Fig. 2. Structure of object descriptor

의 구성을 보여준다. 객체 기술자 ID는 하나의 MPEG-4 세션 내에서 중복되어서는 안 된다^[1]

(1) 기초 스트림(Elementary Stream) 기술자

기초 스트림 기술자는 비디오 및 오디오 스트림이 스케일러를 부호화되어 복수의 스트림으로 나뉘어져 있는 경우, 각각의 스트림에 대해서 필요하게 된다. 각각의 기초 스트림의 기술자는 ES_ID에 의하여 식별된다. ES_ID는 16비트로 할당된다. 나중에 설명할 MPEG-2 트랜스포트 스트림에 실린 MPEG-4 데이터를 분리해낼 때, 그리고 각각의 객체를 구분할 때 이 ES_ID가 중요한 역할을 한다. 기초 스트림 기술자는 스트림의 종류를 판별하기 위하여 스트림 타입, 프로파일, 복호화에 필요한 버퍼 크기, 스트림의 최대/평균 전송률 등을 기술하는 복호화 설정 기술자를 포함하고 있으며 동기화 헤더 구성정보를 가진 SL_Config 기술자도 포함한다. 그리고 콘텐츠 정보를 기술하는 OCI 기술자, 저작권과 관련된 정보를 기술하는 IPMP 기술자도 기초 스트림 기술자 하층구조로 포함되어 있다.

(2) 객체 기술자 스트림

객체 기술자 및 기초 스트림 기술자는 MPEG-4 프리젠테이션 개시에 필요할 뿐만 아니라, 프리젠테이션 도중에 스트

림을 추가, 삭제, 변경 등을 할 때도 필요하다. 객체 기술자도 한 개의 기본 스트림으로 구분한다. 객체 기술자와 그 제어 명령을 전송하는 스트림을 객체 기술자 스트림이라고 부른다. ObjectDescriptorUpdate, ObjectDescriptorRemove는 세션의 개시와 장면에 새로운 스트림이 추가된 경우에 사용된다.

ES_DescriptorUpdate와 ES_DescriptorRemove는 객체 기술자중의 일부 기초 스트림 기술자를 갱신할 때 사용한다.

3. 초기 객체 기술자(Initial Object Descriptor)

MPEG-4 세션에서 최초에 전송되는 데이터로써 초기 액세스 포인트, 즉 장면 기술자 스트림이나, 객체 기술자 스트림에 대한 정보를 가진 기술자이다. 초기 객체 기술자 구조는 기본적으로 객체 기술자와 비슷하다. 즉 복호화에 필요한 객체 정보를 가진 객체 기술자와 장면기술자를 가리키는 또 다른 기술자로 작용한다.

4. FlexMux

FlexMux 옵션으로서 다수의 스트림을 동시에 다중화할 때 전송 스트림의 오버헤드를 줄이기 위하여 사용되거나 또는 논리채널의 부족을 보완하는 경우에 사용된다.

FlexMux 패킷은 단순(Simple) 모드와 다중화 코드(MuxCode) 모드 두 가지가 있다. 그림 3과 4에서 알 수

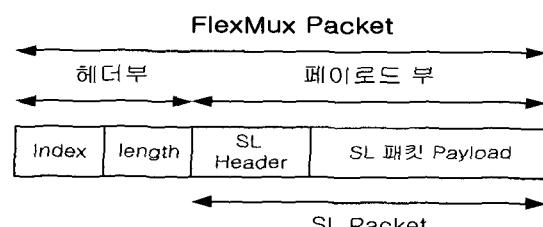


그림 3. 단순 모드의 FlexMux 패킷
Fig. 3. A Flexmux packet in simple mode

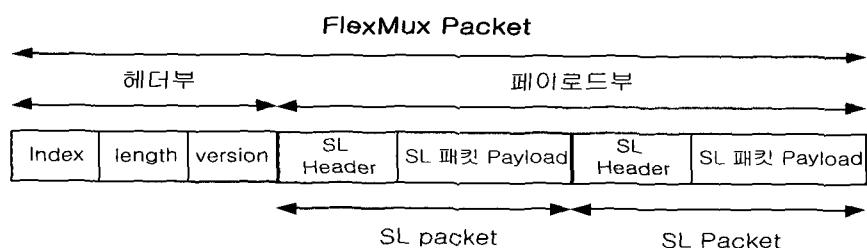


그림 4. 다중화 코드 모드의 FlexMux 패킷
Fig. 4. A Flexmux packet in MuxCode mode

있듯이 다중화 코드 모드는 복수의 SL 패킷을 그것의 패이로드로 갖는반면 단순 모드에서는 하나의 SL 패킷을 가진다. FlexMux 패킷은 8비트의 index와 length 필드로 구성된다. 여기서 length 필드는 페이로드의 길이를 나타내고 index는 해당 채널에 해당한다. 나중에 이 값을 가지고 원하는 SL 패킷을 찾아낼 수 있다. 여기서 index 필드가 8비트로 표현되기 때문에 완전한 255바이트 내의 SL 패킷만을 가져야 한다. index가 0~239의 경우는 단순 모드가 되며 240 이상에서는 다중화 코드 모드가 된다. 다중화 코드 모드인 경우 별도로 다중화 코드 테이블(Muxcode Mode Table)이 전송되어 다중화 패턴을 결정한다^[1].

III. MPEG-4 Over MPEG-2 TS 비트 스트림

MPEG-2 스트림은 독립적인 MPEG-4 비디오나 MPEG-4 오디오뿐만 아니라 이와 관련된 ISO/IEC 14496-1 오디오-비주얼 장면(audio-visual scene)도 전송할 수 있다. MPEG-2 시스템에는 저장미디어를 대상으로 하는 프로그램 스트림(Program Stream) PS와 HDTV 방송망 등 전송을 위한 트랜스포트 스트림이 존재한다^[5]. MPEG-4 콘텐츠는 초기 객체 기술자와 다양한 여러 가지 스트림, 예를 들면 객체 기술자, 장면 기술자 스트림, IPMP, OCI 스트림, 그리고 오디오-비주얼 스트림으로 구성된다. 이러한 모든 스트림은 시간정보를 포함하는 SL 패킷 혹은 FlexMux 패킷으로 만든 후 MPEG-2 트랜스포트 스트림에 실리게 된다. 이번 절에서는 MPEG-4의 다양한 스트림을 어떻게 트랜스포트 스트림에 구조화하여 전송할 것인가에 대한 방법을 기술한다.

1. SL 패킷과 FlexMux 패킷

MPEG-4에서의 SL 패킷은 동기화를 이루는 기본 단위로 사용되며 헤더와 페이로드로 구성된다. 헤더는 타임스탬프와 연관된 정보들이 들어 있고 기초 스트림 기술자 안에 있는 SLConfig 기술자 값에 따라 달라진다. SL 패킷은 길이정보를 포함하고 있지 않기 때문에 FlexMux 툴을 이용하여 FlexMux 패킷화한다. 또한 각각의 SL 패킷 스트림을 하나의 FlexMux 채널에 대응시킴으로써 나중에 추가 정보 없이 하나의 데이터로 쉽게 구분할 수 있다. 본 논문은 단순 모드의 FlexMux 패킷을 사용하였다.

2. SL 비트 스트림의 PES 혹은 Private 섹션에 의한 전송

객체 기술자 스트림, 장면 기술자 스트림을 제외한 모

든 스트림은 PES 패킷에 구조화한다. SL 패킷화된 스트림을 PES 스트림에 실어 전송할 때는 SL 기술자를 이용하여 트랜스포트 스트림의 elementary_PID값과 SL 패킷화된 스트림의 ES_ID값을 연결시켜줘야 한다. 이 SL 기술자는 트랜스포트 스트림의 PMT(Program Map Table)의 기술자 루프(descriptor loop)에 기초 스트림과 함께 전송한다^[6]. 그림 5는 SL 기술자의 구조를 나타낸다. 아래 그림에서 descriptor_tag는 여러 기술자중 SL 기술자를 가리키는 값으로 표준에서 정의한 값 30으로 할당된다.

```
SL_descriptor () {
    descriptor_tag 8 bit
    descriptor_length 8 bit
    ES_ID          16 bit
}
```

그림 5. SL 기술자의 구조
Fig. 5. Structure of SL descriptor

FlexMux 툴을 이용하여 SL 패킷화된 스트림을 FlexMux 스트림으로 다중화한 경우는 마찬가지로 FMC_descriptor가 요구된다. 이는 PES에 실어 보낼 때뿐만 아니라 MPEG-4 섹션에 실어 전송할 때도 마찬가지이다. 이 FMC_descriptor는 FlexMux 채널과 SL 패킷화된 스트림의 ES_ID값을 FlexMux 스트림내에서 연계시켜 주는 역할을 한다. SL 기술자와 마찬가지로 PMT의 기술자 루프에 전송한다. 그림 6은 FMC_descriptor의 구조를 나타낸다.

```
FMC_descriptor () {
    descriptor_tag 8 bit
    descriptor_length 8 bit
    for(i=0; i<descriptor_length; i+=3){
        ES_ID          16 bit
        FlexMuxChannel 8 bit
    }
}
```

그림 6. FMC_descriptor의 구조
Fig. 6. Structure of FMC_descriptor

위 그림에서 ES_ID 값은 SL 패킷화된 스트림이 가지고 있는 기초 스트림의 ID값이고 FlexMux의 index값에 해당하는 값이 FlexMuxChannel 값이다. 이는 SL 패킷화된 스트림의 ES_ID 값과 FlexMux 채널을 대응시키는 방법이다. 본 논문에서는 모든 스트림은 SL 패킷화한 후

이를 다시 FlexMux 패킷 스트리밍하여 PES 패킷에 전송하였다. 다중화 코드 모드의 FlexMux 스트림을 전송할 경우 MuxCodeTableEntry가 필요한데 이때는 Muxcode 기술자가 필요하다. 본 논문의 모든 FlexMux 패킷 스트림은 단순 모드를 이용하였다.

반면 객체 기술자 스트림이나 장면 기술자 스트림은 PES 패킷에 구조화할 수도 있지만 본 논문에서는 특별히 이 두 가지 스트림에 대해서 MPEG-4 섹션에 실어 TS 패킷화하였다. 이는 BIFS Command 스트림과 OD 기술자 스트림은 다른 콘텐츠를 기술하는 정보이지 시간정보와 관련이 없기 때문에 굳이 PES 패킷에 실을 필요가 없는 것이다. 즉 섹션에 실을 수 있는 것은 BIFS Command 스트림과 객체 기술자 스트림으로 제한되어 있다.

3. MPEG-4 비주얼 데이터 및 오디오 데이터의 전송

독립적인 MPEG-4 오디오-비주얼 스트림은 PES 패킷

페이지로 실어 전송된다. 이럴 경우 동기를 맞추기 위해 PTS(Presentation Time Stamp)와 DTS(Decoding Time Stamp)가 PES 패킷헤더에 적절히 인코딩되어야 한다. 이럴 경우 MPEG-4_video_descriptor와 MPEG-4_audio_descriptor에 의해 정보를 기술해줘야 하는데 위에서 설명한 다른 기술자와 마찬가지로 PMT의 기술자 루프에 전송된다. 이들 기술자의 역할은 각각 비디오 프로파일과 레벨, 오디오 프로파일과 레벨을 정의 해주는 역할을 한다.

지금까지 설명한 MPEG-4의 각각의 콘텐츠를 TS 패킷화하는 방법을 그림으로 나타내면 아래와 같다. 그림 7은 FlexMux 패킷화한 후 PES 패킷에 실은 다음 이것을 다시 188 바이트의 TS 패킷으로 분할하는 과정을 보여주고 있다.

그림 8은 FlexMux 패킷화한 후 이를 섹션에 실어 TS 패킷으로 분할하는 과정을 보여주고 있다. 단 BIFS Command 스트림과 객체 기술자 스트림에 한해서이다.

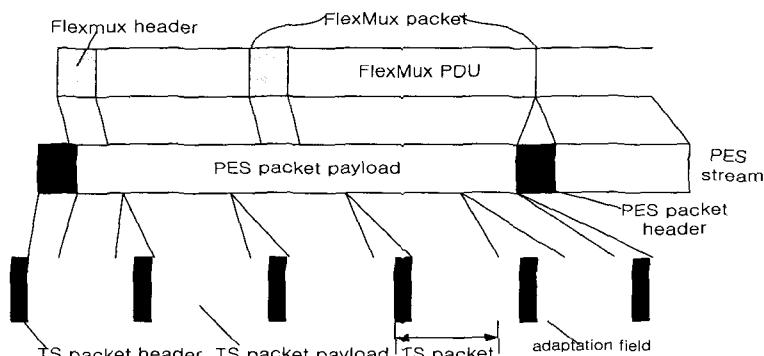


그림 7. PES 스트림의 TS 패킷화
Fig. 7. TS packetization of PES stream

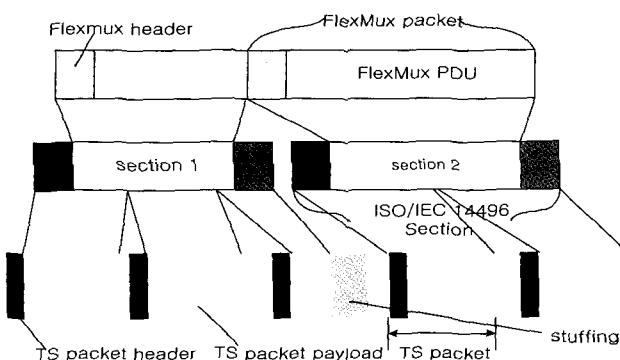


그림 8. 섹션의 TS 패킷화
Fig. 8. TS packetization of section

4. 초기 객체 기술자의 전송

초기 객체 기술자는 MPEG-4의 다른 모든 스트림을 구분하기 위해 필요한 초기 액세스 포인트다. MPEG-2 TS상에 비트 스트림을 전송하는 데 있어서도 마찬가지로 이 초기 객체기술자를 맨 먼저 전송해야 하는데 트랜스포트 스트림의 PMT의 첫 번째 기술자 루프에 IOD_descriptor를 이용하여 초기 객체 기술자를 전송한다. 이를 통해서 장면 기술자 및 객체 기술자 스트림의 ES_ID를 찾아낸다.

IV. MP4 파일의 설계

MP4 파일 포맷은 ISO/IEC 14496-1에서 규정하고 있는 오디오-비주얼 장면을 효율적으로 저장하기 위한 목적으로 설계되었다^[7]. 이러한 MP4 파일 포맷은 단순한 데이터의 저장을 위한 목적 외에 보다 유동적이고 확장이 용이한 구조를 제공하고 있다. MP4 파일의 구조는 Apple사의 Quicktime 파일 포맷에 기반을 두고 있는데 기본 데이터 저장구조인 atom은 주로 Quicktime에서 사용되고 있는 것들이 대부분이며 MPEG-4 프리젠테이션을 저장하기 위해 몇 가지 atom이 추가되었다^[8]. 또한 교환이나 배포 목적이라면 여러 미디어 객체를 하나의 파일로 저장할 수도 있고 미디어 객체를 재편집하거나 변경해서 사용할 경우라면 각각 독립적인 파일로 저장할 수도 있다. 그뿐만 아니라 모든 미디어 객체를 파일로 만들 수 없는 경우에는 네트워크로 전송하여 재생할 수 있고, 또한 이 파일 포맷은 어떠한 전송 프로토콜에 독립적으로 설계되었다. 본 논문에서는 여러 미디어 객체를 하나의 파일로 구현하였다. 이번 장에서는 MP4 구성요소 및 설계 방법을 간략히 설명하고자 한다.

1. 기본 구조

MPEG-4 프리젠테이션에 의해 재생되는 장면은 수많은 개체들이 합성된 장면이다. 각 개체별로 최소한 하나씩의 기초 스트림이 필요하다고 가정할 때, 이를 저장하는 MP4 파일은 수많은 종류의 미디어 데이터를 포함하는 구조가 된다. 결과적으로 MP4 파일을 구성할 때, 미디어 데이터를 어떻게 배치하는가는 매우 중요한 고려사항이 된다. MP4 파일 포맷에서는 미디어 데이터의 배치방법을 크게 두 가지로 분류하고 있다.

첫 번째 방법은 하나의 파일에 모든 미디어 데이터를 수록하는 방법이고, 두 번째 방법은 미디어 데이터를 각각 별도의 파일에 저장하는 방법이다. 첫 번째 방법으로 구성한 경우에는 하나의 파일에 모든 프리젠테이션 데이터가 저장된다는 장점을 가지고 있지만, 재생을 위해서 장면을 구성하려면 파일의 여러 부분에 나뉘어 저장되어 있는 각각의 미디어 데이터를 추출해 내기 위해서, 파일 전체를 메모리에 저장해야 할 필요가 있다. 이와 비교해서 두 번째 방법으로 구성한 경우에는 시스템이 재생을 위해 많은 파일을 동시에 읽어야 한다는 단점이 있지만, 각각의 미디어 데이터를 추출하거나 재사용하기에 편리하다는 장점이 있다.

2. Meta 데이터와 Media 데이터

Meta 데이터는 미디어 데이터에 대한 세부정보를 가지고 있는데 미디어의 타입과 속성, 재생시간, 길이정보 등을 포함한다. Media 데이터는 부호화된 오디오 비주얼 등의 데이터로 MPEG-4에서 정의하고 있는 기초 스트림 개념과 유사하다. MP4 파일은 크게 메타 데이터인 moov와 미디어 데이터인 mdat로 구분할 수 있으며 moov에는 그림 9에서 볼 수 있듯이 여러 개의 트랙(track)으로 이루어져 있다. 각각의 트랙은 대응되는 하나의 미디어 데이터에 관한 정보를 포함하고 있어서 mdat의 비디오 데이터를 재생하려고 하면 반드시 video 트랙에서 그 정보를 가지고 재생하게 된다. 또한 미디어 데이터에 적용되는 트랙 외에 BIFS 트랙과 객체 기술자 트랙(OD Track)이 존재해서 MPEG-4의 오디오-비주얼 장면간의 시공간적 관계를 나타낸다. 물론 객체 기술자 스트림과 BIFS 스트림도 하나의 미디어 객체로 간주해서 mdat에 위치한다.

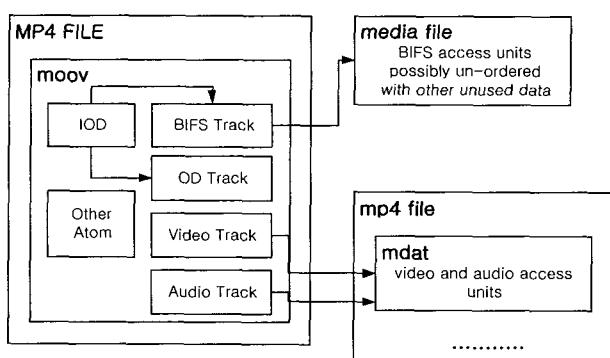


그림 9. moov와 mdat의 관계

Fig. 9. Relation between moov and mdat

MP4 파일 포맷은 스트리밍 포맷이 아니라 스트리밍이 가능한 포맷이다. 이것은 on-the-wire 프로토콜을 정의하고 있지 않다는 뜻이다. 그리고 사실상 전송매체에 스트리밍되지 않는다. 대신에 메타데이터 내의 'hint tracks'이라 불리는 정보가 어떻게 특정 전송 프로토콜에 미디어 데이터를 전송할 것인가에 관해서 서버 애플리케이션에 그 방법을 알려주고 있다. 하나의 프리젠테이션 내에 여러 전송 채널에 관한 정보를 갖는 여러 개의 hint track이 존재할 수도 있다. 이런 방법으로 직접적으로 스트리밍 하지 않고 스트리밍 기술을 구현한다^[7]

3. 객체 기술자 스트림

그림 9의 mdat 부분에 객체 기술자 스트림은 MPEG-4 시스템에서 정의하고 있는 객체 기술자 스트림과 약간 다르고 그 구조가 매우 단순하다. 본 논문에서는 MP4 파일 내에서 정의하고 있는 객체 기술자 스트림의 구조를 따랐다. 예를 들면 기초 스트림 두개를 가진 객체 기술자 스트림은 다음과 같은 값을 가진다.

```
00 00 00 00 00 00 00 1C 01 80 80 0A 01 80 80 06
00 DF 0F 02 00 01 01 80 80 0A 01 80 80 06 01 1F
0F 02 00 02
```

실제 MP4 파일 내에는 00-1C의 8바이트는 없고, 01~02의 28바이트만 존재한다.

- ▶ 첫 8번째 바이트 1C라는 것은 나머지 객체 기술자 스트림의 크기 ($0x1C = 28$)
- ▶ 나머지 28 바이트는 한 트랙 또는 객체별로 14바이트씩 할당된다. 두 개의 기초 스트림이 있으므로 $14*2=28$ 바이트가 된다.
- ▶ 첫번째 01: OD Update tag
- ▶ 다음 80 80 0A 는 다음 크기 ($14-4=10=0x0A$)
- ▶ 01: OD tag
- ▶ 80 80 06: 크기 정보 6바이트
- ▶ 00 DF: 바이너리로 값으로 살펴보면,
"0000 0000 1101 1111" : 첫 10 비트 ("0000 0000
11") → OD_ID = 3
다음 0 은 priority 마지막 "1 1111" 은 reserved 값
- ▶ 0F 는 ES_ID_RefTag
- ▶ 다음은 크기 02
- ▶ 마지막 2바이트 00 01 은 ES_ID(1번) (16비트)

ES_ID 1부터 시작한다.

- ▶ 두 번째 객체 기술자 역시 14바이트로 나머지부분은 같고 OD_ID와 마지막의 ES_ID만 증가했다.

V. MP4 파일의 변환기 설계

1. MPEG-4 비트 스트림 분리과정

MPEG-4 Over MPEG-2 트랜스포트 스트림을 MP4 파일로 변환하기 위해서는 먼저 원하는 MPEG-4 프로그램 컴포넌트를 트랜스포트 스트림상에서 분리해 내어야 한다. 이 분리과정 순서가 그림 10에 나타나 있다.

MPEG-2 TS의 PSI 정보는 여러 개의 프로그램에 전송되어 올 때 하나의 프로그램을 구분해 낼 수 있는 정보이다. PSI 정보에는 PAT(Program Access Table), PMT(Program Map Table), CAT(Conditional Access Table), NIT(Network Information Table) 등이 있다^[9]. 이 중 PAT는 다수의 프로그램 중 하나를 선택할 수 있는 정보를 가지고 있으며 PMT는 한 프로그램을 구성하는 기초 스트림들에 대한 정보들을 가지고 있다. CAT는 프로그램을 시청할 권리를 가진 사람만이 볼 수 있도록 암호화와 연관된 정보를 가지고 있으며 NIT는 시스템 복호기에서 사용하는 정보가 아니라 채널에 관련된 레이어(layer)에서 사용하는 정보이다.

PAT에는 각 프로그램에 대한 PMT 테이블의 PID값을 갖고 있다. 따라서 첫째로 원하는 프로그램에 해당하는 Program Map Table(PMT)을 PAT에서 획득한다. 그림 10에서는 program_number = 1인 값에 해당하는 PMT의 PID 100 번을 얻었다. 둘째로 PID 100인 트랜스포트 스트림 패킷에서 PMT정보를 분석한다. PMT의 첫 번째 기술자 루프에는 초기 객체 기술자 IOD_descriptor 정보가 있는데 ES_ID가 10인 경우는 Stream_type이 BIFS Command 스트림인 것과 ES_ID가 11인 경우는 객체 기술자 스트림이란 정보를 알아낸다. 역시 PMT의 두 번째 기술자 루프에서 해당 트랜스포트 스트림의 PID 값과 FMC_descriptor를 이용하여 ES_ID와 그에 해당하는 채널의 정보를 얻는다. 또한 그에 해당하는 트랜스포트 스트림의 PID를 함께 알아낸다. 세 번째로 PID 111번을 분석하여 장면 기술자 스트림을 획득하고 마찬가지로 112번을 분석하여 객체 기술자 스트림을 획득한다. ES_ID가 20번인 경우 비주얼 스트림인 것을 알고 그것의 PID는 PMT의

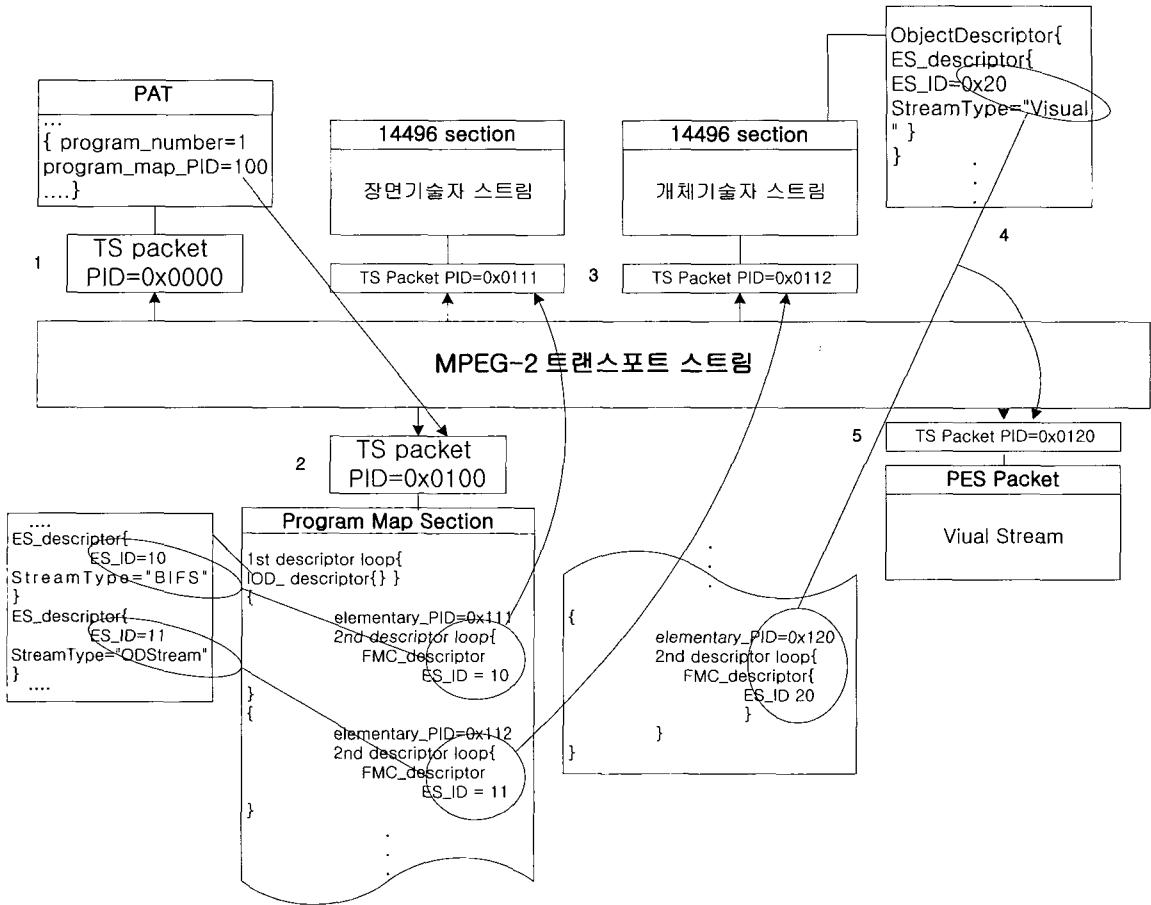


그림 10. TS에서 MPEG-4 스트림 분리과정
Fig. 10. Demultiplexing MPEG-4 stream from TS

표 2. Stream Map Table
Table 2. Stream Map Table

TS packet		IOD, OD, FlexMux descriptor	
PID	ES_ID	FlexMux channel	Stream Type
111	10	01	장면기술자 스트림
112	11	02	개체기술자 스트림
120	20	10	visual 스트림
120	20	11	audio 스트림

두 번째 기술자 루프에서 120인 것을 알 수 있다. 지금 까지 분석한 모듈로 트랜스포트 스트림의 PID, ES_ID, FlexMux 채널을 연계시키는 Stream Map Table을 표 2와 같이 만들 수 있다.

이 Stream Map Table과 객체 기술자 스트림을 이용하여 다른 여러 가지 스트림을 분리해 낼 수 있는데 이렇게

분리해낸 MPEG-4 스트림이 MP4 파일 인코더의 입력 값에 해당한다.

2. 변환기 설계

추출해낸 IOD와 장면기술자, 객체기술자 스트림은 각각 meta 데이터를 구성하는데 이용되고 기초 스트림의 개수 만큼 트랙을 구성하는데 총 기초 스트림 개수에 BIFS 트랙과 객체 기술자 트랙을 더한 개수를 생성한다. 또한 meta 데이터에서 빠지지 말아야 할 것이 'Track Reference' 트랙이다. 이를 이용하여 기초 스트림의 모든 트랙과 객체 기술자 트랙과 연계시켜 줘야만 한다^[10].

meta 데이터 구성이 끝나면 순서대로 기초 스트림을 이용하여 mdat를 만든다. mdat에는 먼저 BIFS 스트림이 위치하고 그 다음으로는 객체 기술자 스트림, 그리고 순서

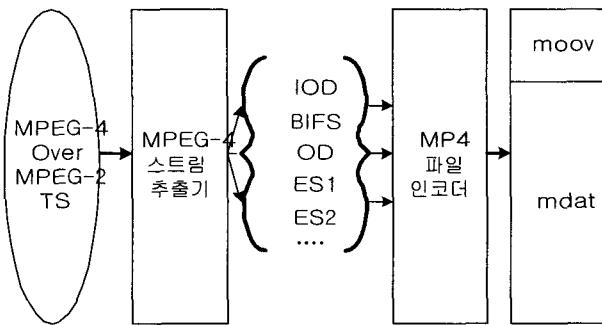


그림 11. 전체 변환기 블록도

Fig. 11. Structure of format converter

대로 기초 스트림이 위치한다. 전체 MP4 변환기 블록도는 그림 11과 같다.

여기서 MP4 파일내의 mdat를 이루는 것중 하나인 객체기술자 스트림은 그림 10과 같이 분리해낸 다음 ES_ID 값과 개수를 알아낸 후 IV장의 3절에 설명한대로 그 내용을 변경하여 MP4 파일 부호화기의 입력 값으로 할당한다.

3. 구현한 MP4 파일

실험과정은 먼저 MPEG-2 인코더로 트랜스포트 스트림을 만든 후에 직접 MPEG-4 스트림을 끼워 넣거나, MPEG-4를 명시하기 위해 트랜스포트 스트림 패킷 일부를 수정하였다. MPEG-4 스트림의 경우 IM1의 BIFS 부호화기로 BIFS 스트림을 작성하였고 나머지 비주얼 스트림인 JPEG파일을 그대로 이용하였다^[10]. H.263 동영상 스트림인 경우 매 프레임을 SL 패킷화한 후 이를 PES에 실어 TS 패킷화 하였다.

MP4 파일 재생에는 IM-2D 플레이어를 이용하여 이를 검증하였다^[11]. 그림 12 (a)는 JPEG파일 13개로 이루어진 MP4 파일이다. 그림 12 (b)는 그림 12 (a)에서 오른쪽 4개의 그림 중 하나를 선택했을 때 바뀌어지는 화면을 나타냈는데 이는 BIFS를 통해서 구현된 결과이다.

그림 12 (c)는 JPEG 파일뿐만 아니라 H.263동영상도 함께 재생되는 MP4 파일이다. 가운데 마지막 화면이 H.263 동영상 화면이다.



그림 12. 구현한 MP4 파일 (a) JPEG 이미지 13개로 이루어진 MP4 파일 (b) (a)파일의 Interaction 장면 (c) H.263 동영상과 JPEG 이미지로 이루어진 MP4 파일

Fig. 12. Examples of MP4 files (a) MP4 file for 13 JPEG images (b) Interaction of (a) (c) MP4 file for H.263 videos and JPEG images

VI. 결 론

MPEG-4는 단순히 영상 및 음성의 전송, 배포 용도뿐만 아니라, MPEG-4가 제공하는 상호 작용성등의 특징을 살려서 다양한 분야에 적용하는 것이 가능하다. 그 중에서 디지털 방송망으로 이용되는 MPEG-2 트랜스포트 스트림에 MPEG-4를 부가 서비스 방식으로 이용하는 것을 들 수 있다^[12].

MPEG-4는 기존의 MPEG-1,2와는 다른 시스템을 갖는다. 또한 MPEG-2 트랜스포트 스트림이 표준화되고 나서 그 이후에 표준화 작업이 진행되었기 때문에 MPEG-2 트랜스포트 스트림에 MPEG-4를 실기 위해 MPEG-2 시스템의 표준이 수정, 추가되었다. 본 논문은 이것을 바탕으로 MPEG-4 스트림에 대한 연구와 이를 바탕으로 MPEG-2 트랜스포트 스트림에 MPEG-4 비트 스트림을 실어서 이를 저장미디어 대상 파일 방식인 MP4로 변환하는 알고리듬 및 변환기를 설계해 보았다.

MPEG-4 Over MPEG-2 트랜스포트 스트림을 만들 때 기존의 MPEG-2 트랜스포트 스트림 부호화기를 사용하여 만들었고 MPEG-4 비트 스트림을 올릴 때는 트랜스포트 스트림 일부를 수정하여 작성하였다. 개선해야 할 사항으로는 차후 자동적으로 MPEG-4를 MPEG-2 트랜스포트 스트림에 실는 모듈을 개발하고 비디오뿐만 아니라 오디오 또한 포함시켜 MP4 파일을 구성하도록 하고, 또한 현재는 수신 비트 스트림을 다 받고 나서야 MP4 파일을 작성할 수 있지만 차후 이를 실시간으로 변환할 수 있도록 하는 점이다.

본 논문의 MPEG-4 Over MPEG-2 TS 비트 스트림의 MP4 변환기는 향후 MPEG-4의 응용분야에 효과적인 모델이 될 것이다.

참 고 문 헌

- [1] ISO/IEC 14496-1, "Information technology-Coding of Audio-Visual Objects Part I: Systems," *International Standard*, 1999.
- [2] 고성제, 김종욱 역, *MPEG-4의 세계*, 영풍문고, 2000
- [3] ISO/IEC 14496-1, "Information Technology-Coding of Audio-Visual Objects Part VI: DMIF," *International Standard*, 1999.
- [4] ISO/IEC 14772-1, "International Standard, Virtual Reality Modeling Language," 1997.
- [5] 유시룡, 장규환, 이병욱, 김종일, 정해묵, "MPEG 시스템", 대영사, 1997.
- [6] ISO/IEC 13818-1, "Information Technology-Generic Coding of Moving Pictures and Associated Audio:Systems Amendment 7: Transport of ISO/IEC 14496 data over ISO/IEC 13818-1," *International Standard*, 1999.
- [7] David Singer, William Belknap, ISO/IEC ISO/IEC subpart4 "Text for ISO/IEC 14496-1/PDAM1 (MPEG-4 version 2 Intermedia Format-MP4)," *International standard*, 1999.
- [8] Apple Computer, Inc, "Quicktime File Format Specification, Developer Press," 1996.
- [9] ISO/IEC 13818-1, "Information Technology-Generic Coding of Moving Pictures and Associated Audio Information : Systems," *International Standard*, 1994.
- [10] "MPEG-4 Reference Software Guide," *Apple Computer, Inc* 1999.
- [11] <ftp://ftp.fzj.de>
- [12] 정제창 역, 그림으로 보는 최신MPEG, 교보문고, 1997.

저 자 소 개



최 재 영

1999년 2월 : 한양대학교 공과대학 전자통신전파 공학과 졸업 (공학사)
 1999년 7월 ~ 1999년 12월 : 한국전자부품연구원(KETI) System IC 위촉 연구원
 2001년 2월 : 한양대학교 대학원 전자통신공학과 졸업예정 (공학석사)
 주관심분야 : 영상 통신 및 압축, 영상처리, 디지털 신호처리, MPEG-4 시스템 등



정제창

1980년 2월 : 서울대학교 공과대학 전자공학과 졸업 (공학사)
1982년 2월 : 한국과학기술원 전기전자공학과 졸업 (공학석사)
1990년 8월 : 미시간대학교(앤아버) 전기공학과 졸업 (공학박사)
1982년 2월 ~ 1986년 7월 : 한국방송공사 기술연구소 연구원 (뉴미디어 연구개발)
1990년 9월 ~ 1991년 1월 : 미시간대학교(앤아버) Post-dotorial Research Fellow
1991년 2월 ~ 1995년 2월 : 삼성전자 멀티미디어 연구센터 신호처리연구소 수석연구원
(HDTV 및 멀티미디어 연구개발)
1995년 3월 ~ 현재 : 한양대학교 전자통신공학과 교수
주관심분야 : 영상 및 음성 압축, 영상처리, 디지털 신호처리, 디지털 통신, VLSI 설계 등