DMP 기반의 Gstreamer를 이용한 DRM시스템

박기철* · 이주영** · 안상우** · 남제호** · 정회경*

*배재대학교 컴퓨터공학과, **한국전자통신연구원

DRM System using Gstreamer based on DMP

Ki-Chul Park* · Joo-Young Lee** · Sang-Woo Ahn** · Je-Ho Nam** · Hoe-Kyung Jung*

*Dept. of Computer Engineering, Paichai University · **ETRI

E-mail : *{kcpark, hkjung}@pcu.ac.kr, **{leejy1003, asw, namjeho}@etri.re.kr

본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업의 일환으로 수행하였음.
[2007-S-003-01, 지상파 DTV 방송프로그램 보호 기술개발]

요 약

동화상 압축 기술과 네트워크의 발달로 빠르게 보급된 디지털 콘텐츠는 불법복제와 인터넷을 통한 유포의 피해로 지적 재산권에 대한 보호의 필요성이 요구되었다. 이를 배경으로 지적재산권 보호를 위해 여러 표준화 단체들과 콘텐츠 공급자들이 DRM(Digital Rights Management) 시스템을 개발하였다. 그러나 서로 독자적인 DRM 시스템 개발로 인해, 이기종 DRM 시스템간의 상호운용성이 결여되는 문제점을 낳았다. 이런 문제를 해결하기 위해 MPEG(Moving Picture Experts Group)에서는 IPMP(Intellectual Property Management and Protection) 표준을 제안 하였으나 흡족한 기능을 제시하지 못하여 이를 보완하는 DMP(Digital Media Project) 표준이 제시되었다. 이는 콘텐츠의 보호를 위해 툴(Tool)을 사용하고 이기종간의 상호 운용성과 콘텐츠의 지적재산권 보호를 제공한다.

본 논문에서는 오픈소스 멀티미디어 프레임워크인 Gstreamer를 이용하여 DMP 표준을 준수하는 DRM 시스템을 설계 및 구현하였다.

ABSTRACT

Due to illegal copying and distribution via the internet, the digital media that have been introduced create certain intellectual property issues. Because of this, several intellectual property groups have been created to create and maintain Digital Rights Management systems (DRM), in order to crack down on illegal and unauthorized copying. However, these systems are often created by many different companies, which leads to problems of interoperability. Despite attempts to regulate DRM systems, including the MPEG group's Intellectual Property Management and Protection" system, the system is still struggling to maintain some sort of interoperability between differing protocols.

In this paper, we designed and implemented a DRM system with Gstreamer, following the DMP.

키워드

DMP, DRM, Gstreamer, ToolPack, Chillout

Ⅰ. 서 론

IPTV의 상용화, WiBro(Wireless Broadband), HSDPA(High-Speed Downlink Packet Access)등 새로운 통신 서비스의 확대와, 통방융합의 가시화로 인해 국내 디지털 콘텐츠 산업은 안정적인 성장세를 유지할 것으로 전망된다[1].

그러나 사용자에 의한 변형 및 복제가 용이한 디지털 콘텐츠의 특성으로 인해 신뢰성 있는 유통체계의 확립은 디지털 콘텐츠 시장에서 필수

적 요건으로 요구되어지고 있다. 이에 콘텐츠 업체 및 IT 업체들은 콘텐츠를 보호하고, 신뢰성 있는 유통체계 확립을 위해 DRM등의 보호기술들을 콘텐츠에 적용하여 유통하고 있다. Apple사의 iTunes와 Microsoft사의 Windows Media DRM등의 예는 성공적인 DRM 사례이지만, 시스템의 독립성으로 인하여 상호간의 포맷 통합이 불가능하다[2]. 이와 같은 DRM 시스템 간의 상호 운용성의 결여는 사용자의 콘텐츠 이용을 제한하는 문제를 유발한다.

이에 본 논문에서는 IDP(Interoperable DRM Platform)표준을 제안하는 DMP의 IDP-2기반 오픈 소스 레퍼런스인 Chillout과 Gstreamer를 이용하여 DRM 시스템을 구현하였다.

II. 관련 연구

2.1 DMP(Digital Media Project)

DMP는 IDP 기술의 개발 및 표준을 제안하고 있고 IDP-2 기반 레퍼런스 소프트웨어 Chillout을 배포하고 있다. DMP가 제안하는 Value Chain은 그림 1과 같다.

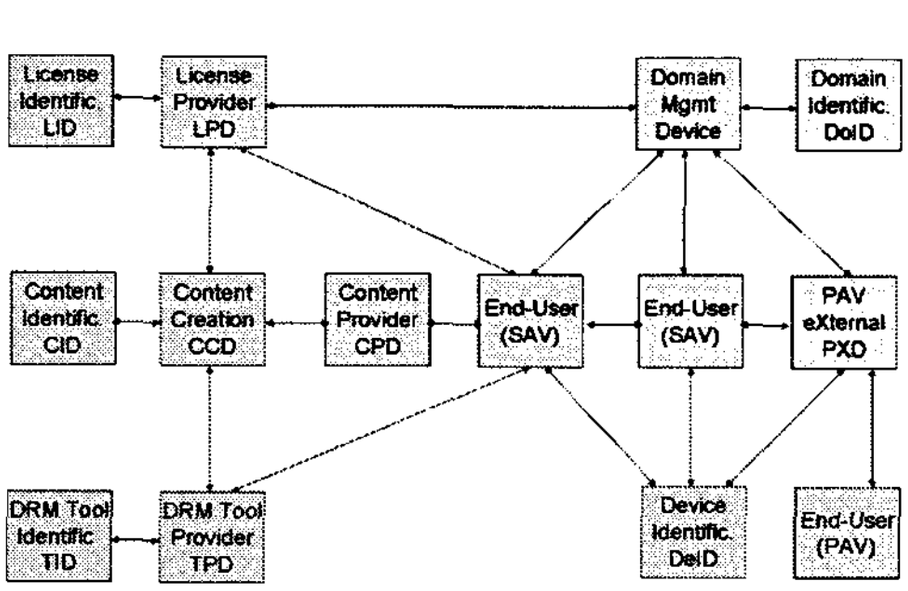

그림 1. DMP Value Chain 구조

Value Chain은 여러 장치(Device)로 구성되는데, TID(Tool Identification Device)는 생성된 툴과 툴에 대한 툴 정보를 등록 관리한다. LID(License Identification Device)는 콘텐츠에 대한 라이센스를 생성 관리하고, CCD(Content Creation Device)는 보호된 콘텐츠를 생성한다. 콘텐츠 생성시 라이센스와 암호화 툴은 LPD(License Provider Device)와 TPD(Tool Provider Device)를 통하여 전달 받고, CCD는 라이센스 정보와 툴 정보를 담은 XML과 암호화된 리소스정보를 합쳐서 Mpeg-21 파일 포맷을 기반으로 한 DCF(DMP Content File) 또는 DCI(DMP Content Informat ion) 파일을 생성한다[3,4]. 이것은 SAV(Stationary Audio Video) 또는 PAV(Portable Audio Video)를 통해 엔드 유저가 사용하게 된다.

DMP에서는 Value Chain을 구현한 오픈소스 레퍼런스인 Chillout을 제공한다. MPL(Mozilla Public License) 1.1을 따르고, 플랫폼에 독립적이며 크로스 컴파일이 가능한 Java 플랫폼을 기반으로 JMF (Java Media Framework API) 2.0과 JAXB(Java Architecture for XML Binding)를 이용한다.

2.2 Gstreamer

Gstreamer는 오픈소스 멀티미디어 프레임워크로서 특징으로는 다음과 같다.

- 광범위한 운영체제와 프로세서 그리고 컴파일러의 환경에서 사용 가능
- 적은 자원의 환경에서 고성능을 보장
- MPEG-4 또는 H.264 등의 최신 포맷 지원
- 용도에 맞는 커스텀 필터 개발에 용이한 환경을 제공
- 콘텐츠의 MIME 타입에 따라 필터를 자동 구성하는 기능
- 다른 프로그래밍 언어로 binding 제공
- Gstreamer는 크게 Source, Filter, Sink 엘리먼트로 구성
- 목적에 따라 각각의 엘리먼트의 링크를 통해 데이터의 pipeline을 구성
- 기능 또는 목적에 따라 Bin이라는 큰 단위의 엘리먼트로 구성
- Bin 또한 서로 링크를 통해 pipeline을 구성

이러한 Gstreamer의 특징은 DMP의 툴을 엘리먼트로 개발하여 원하는 위치에 적용할 수 있고, 다양한 포맷에 유연한 대처가 가능 하다[5].

III. 시스템 설계

본 시스템은 DMP의 IDP-2 표준을 준수한 레퍼런스 Chillout의 성능과 호환성의 개선에 목적을 두고 있다. DCF와 DCI의 처리에 대한 시나리오 또한 Chillout의 구조를 일부 유지하는 관점에서 설계를 하였고, Chillout의 기반 플랫폼의 유지를 위해 Java로 바인딩 한 Gstreamer-java를 함께 사용하였다.

3.1 DCF와 DCI의 처리 과정

DCF와 DCI는 권한의 유무에 따라 Governed와 Non-Governed 타입으로 나뉘고 각각의 타입은 Resource의 암호화 유무의 따라 Protected와 Non-Protected로 나뉜다.

그림 2는 각 조건에 따른 DCF와 DCI 처리 과정을 활동 다이어그램으로 나타낸 것이다.

1. Non-Governed Non-Protected인 경우 파싱 과정을 거친 후 리소스 정보를 추출하여 재생 프로세서를 생성하고 재생을 위한 pipeline을 생성 후 재생 과정을 진행한다.

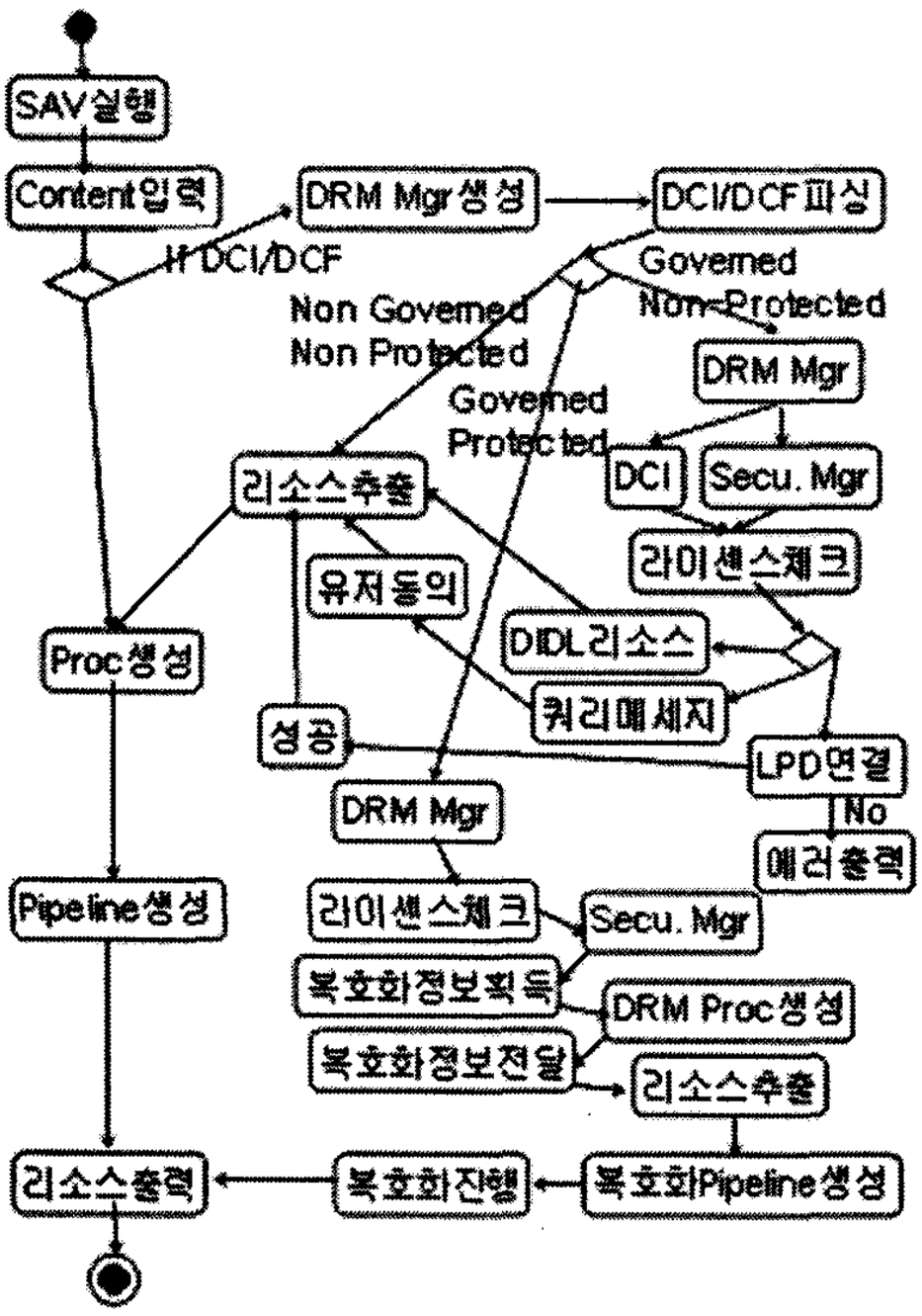

그림 2. DCF, DCI 처리과정 활동 다이어그램

2. Governed Non-Protected인 경우 파싱 과정을 거친 후 DCI에서 라이센스를 검사하거나 DRM Manager 또는 Security Manager에서 라이센스를 검사한 후 라이센스에 따라 리소스 정보를 추출한 다음 재생 프로세서를 생성하고 재생을 위한 pipeline을 생성 후 재생 과정을 진행한다.
3. Governed Protected인 경우 파싱 과정과 DRM Manager의 과정은 (2)와 동일하고 DCI와 DCF를 안에서 Security Manager가 복호화 키와 툴 정보를 구하고 DRM Processor를 생성한다. 리소스 정보와 복호화 정보를 얻은 DRM Processor는 재생을 위한 pipeline 생성 시 복호화 정보를 함께 전달하고, 암호화된 리소스는 복호화 정보에 따라 로드된 툴에 의해 실시간으로 복호화 과정을 거쳐 재생을 진행한다.

3.2 Tool의 설계

구현할 Chillout에서 Tool은 Gstreamer의 pipeline을 구성하는 형태로 구현되어야 한다. 이는 Gstreamer가 제공하는 API를 사용하여 구현되어야 하고, DRM Processor가 전달하는 복호화 키와 위치 정보를 수신해야 한다. 구현을 위한 툴 샘플은 XOR 연산과 DES 알고리즘 암호화, 복호화 Tool을 설계하였고, 복호화 과정에 사용되는 키 값은 8 바이트 문자열로 외부에서 입력 가능 하

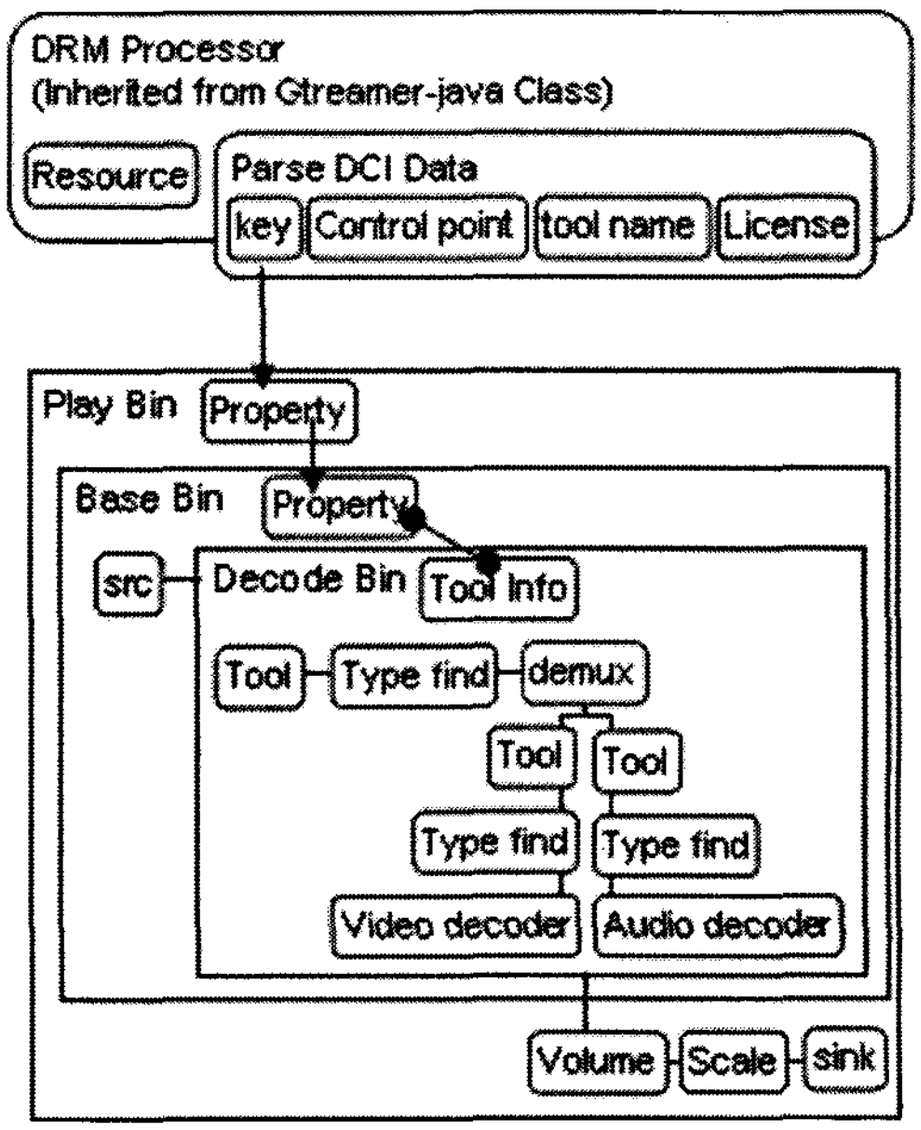

그림 4. 복호화 툴 처리를 위한 pipeline 구조

도록 설계하였다.

3.3 Gstreamer와 Chillout의 연동

Gstreamer는 C를 기반으로 하고, Chillout은 Java를 기반으로 한다. 연동을 위해 Java로 Binding된 Gstreamer-java를 사용하였다. 이것은Gstreamer의 구성요소를 Java에서 접근 가능한 객체로 제공한다. 그림 3과 같이 DRM Processor는 Gstreamer-java를 이용해 원하는 위치에 툴 적용이 가능하다.

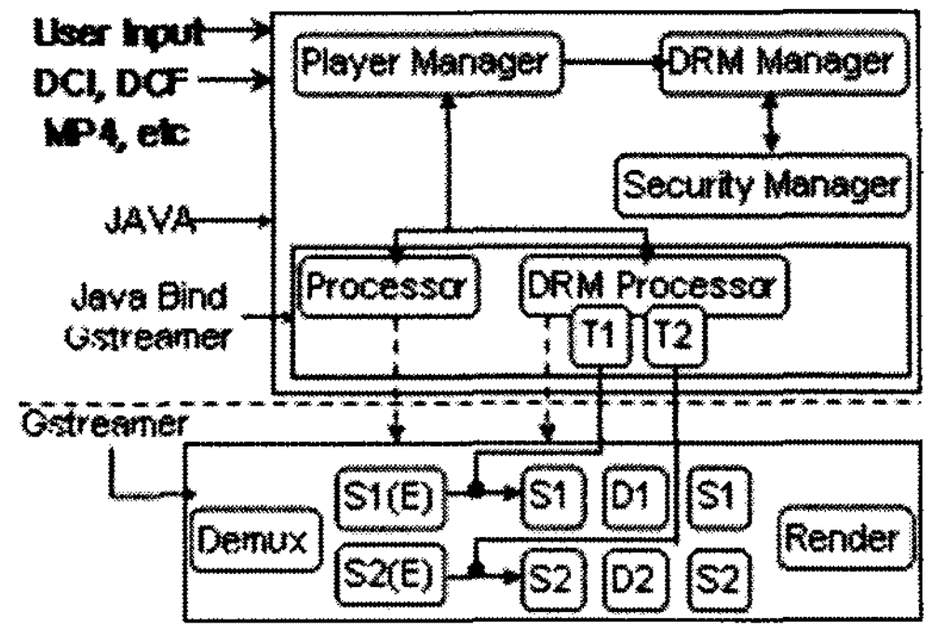

그림 3. 구현할 Chillout SAV의 구조

3.4 Gstreamer의 pipeline 생성

Gstreamer는 미디어 종류에 따라 엘리먼트를 선택적으로 연결해서 pipeline을 구성해야 하는데, Gstreamer는 미디어의 MIME 타입 또는 각각의 pipeline의 연결 간의 스트림 타입을 감지해 필요한 엘리먼트를 자동으로 연결해주는 방법을

제공한다. 하지만 암호화된 미디어에 대해서는 스트림이나 MIME 타입에 따른 감지가 난해하다.

이 문제점을 해결하기 위한 그림 4의 구조처럼 복호화 툴의 정보와 복호화 키의 값 그리고 복호화 위치 등을 DRM Processor에게 전달 받도록 설계하였다.

Ⅳ. 시스템 구현

본 시스템은 IBM-PC 호환 컴퓨터의 MS Windows XP Service Pack 2 운영체제를 기반으로 개발 하였고, Chillout을 위해 JDK 1.5와 멀티미디어 처리를 위해 MS Visual C++ 6.0과 Gstreamer 0.10의 개발 환경에서 구현하였다.

4.1 Gstreamer 구현

Gstreamer의 pipeline에서 복호화를 위해 툴은 Gstreamer의 엘리먼트 형태로 구현하였고, 재생 시 재생을 위한 pipeline을 동적으로 생성하도록 구현하였다. DRM Processor가 생성된 pipeline으로 DRM 정보를 인터페이스 가능 하도록 구현하였으며, 재생 시 화면의 출력은 Java 플랫폼으로 출력 되도록 하였다.

```
Plugin Details:
  Name:                 DESTool
  Description:          DES algorithm encrypt or decrypt Tools
  Filename:             E:\workspace\vc\gst\example\destool\Debug/destool.dll
  Version:              0.10.5.1
  License:              LGPL
  Source module:        gst-plugins-toolpack
  Binary package:       GStreamer Tool Plug-ins
  Origin URL:           Unknown package origin

GObject
 +----GstObject
       +----GstElement
             +----Gstdesdec

Pad Templates:
  SRC template: 'src'
    Availability: Always
    Capabilities:
      ANY

  SINK template: 'sink'
    Availability: Always
    Capabilities:
      ANY
```

그림 5. DES 복호화 툴 정보

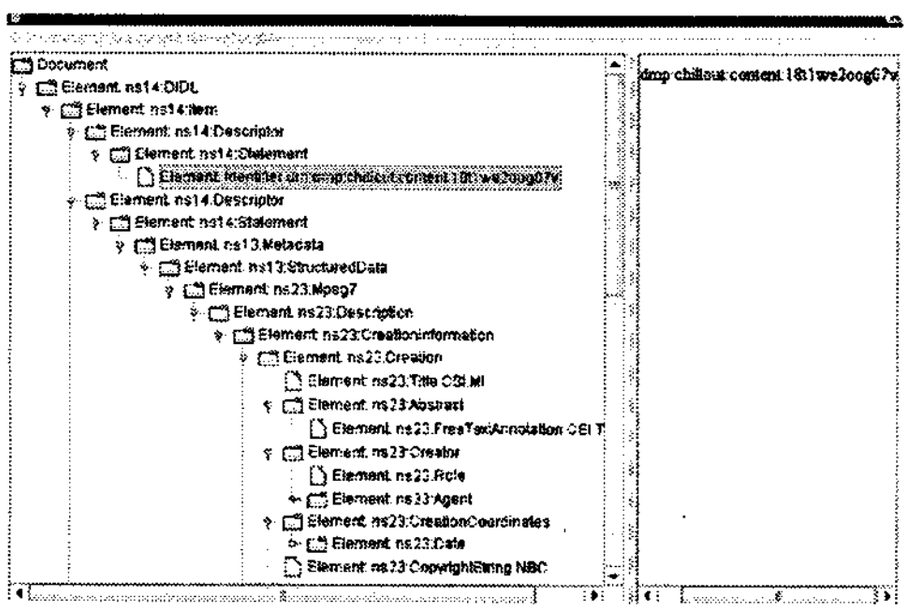

그림 6. CCD에서 생성한 DCI 내용

4.2 CCD의 DCF 생성과 SAV

CCD를 이용하여 새로운 Tool을 적용한 DCI와 DCF를 생성 가능 하도록 하였고, SAV에서 DCF를 열면 라이센스 내용을 출력하고, 재생을 가능하도록 구현하였다.

Ⅴ. 고찰 및 결론

기존 시스템과 비교하여 C 기반의 멀티미디어 프레임워크와 Tool의 사용으로 상당한 성능개선 효과를 보여주었다. 또한 상대적으로 높은 상호운용성과 툴의 확장성, 보안성을 얻을 수 있었다. Gstreamer에서 제공하는 다양한 플러그인으로 새로운 포맷의 미디어에 제한적인 문제도 어느 정도 해결할 수 있었다. JMF 기반의 기존 Chillout과의 Gstramer 기반 Chillout의 비교를 표 1에서 정리해 보았다.

표 1. JMF 기반과 Gstreamer 기반 Chillout 비교

비교항목	JMF 기반	Gstreamer 기반
상호운용성	높음	높음
시스템 의존성	낮음	상대적 높음
성능	낮음	높음
보안성	높음	상대적 높음
미디어지원	낮음	높음
안정성	높음	높음
툴 확장성	보통	상대적 높음

DMP의 표준이 향후 DRM 시장에서 어떠한 평가를 받을지 가늠하기는 시기상조이지만, DRM 시스템의 상호 운용성 측면에서 큰 기여를 할 것으로 사료된다. 다만 본 논문에서 구현된 플랫폼 독립적인 Java 플랫폼과 상대적으로 플랫폼 의존적인 C 기반 Gstreamer 통합은 PAV와 SAV에 있어서 다른 환경으로 이식하기에는 부적합한 구조로 사료된다.

참고문헌

[1] "2006년도 국내 디지털 콘텐츠 산업 시장조사 보고서" - 한국소프트웨어진흥원, 2006

[2] Xiaofan Chen, Tiejun Huan "Interoperability Issues in DRM and DMP Solutions", ICME 2007

[3] ISO/IEC 21000-9, Information technology-Multimedia framework (MPEG-21) - Part 9: File format

[4] L. Chiariglione, N. Earnshaw, "Interoperable DRM Platform (IDP-2)", 2006

[5] Gstreamer features, http://gstreamer.freedesktop.org/features/