Direct Show를 사용한 동영상 플레이어 개발 연구

김민기* · 박대우*

*호서대학교 벤처전문대학원 IT응용기술학과

A Study of Media Player Program Development using Direct-Show

Min-Gi Kim* · Dea-Woo Park*

*Dept. of IT Application Technology, Hoseo Graduate School of Venture

E-mail : *mingi84@naver.com · *prof1@paran.com

요 약

여러 가지 형태의 동영상을 재생하기 위해서는 특정 디코더 및 Demux가 필요하다. 필요에 따라, 많은 개발자들이 필터 개발을 하고 있지만, 기존의 GOM플레이어나 KMP플레이어는 API가 Open되어있지 않아, 작성된 필터 사용의 어려움이 있고, 동영상 재생에 사용되는 Codec의 선택에 대한 문제점이 있다. 본 논문에서는 작성된 필터를 이용하여, 손쉽게 플레이할 수 있는 라이브러리를 개발하며, 화면을 찾기 위한 Seek기능을 설계하고, 화면캡처, 전체화면, 재생화면, 확대, 축소 등 사용자 편의성을 위한 DLL파일로 구축하여 개발한다. 본 연구는 멀티미디어 제작 S/W기술과 정보통신 발전에 기여 할 것이다.

ABSTRACT

Various forms in order to play the video decoder and Demux specific needs. Depending on need, many developers to develop a filter, but KMP existing GOM player or players do not have the Open API, created difficulties in the use of filters and used for video playback issues are the selection of Codec. In this paper, written by the filter, easy to play and develop the library and search for Seek function designed to screen and screen capture, full screen, your screen, zooming DLL files for user convenience, such as building to the development. This study multimedia S/W technology and information and communication will contribute to the development.

키워드

Media Play, Direct Show, Codec, Filter SDK, Movie Player

Ⅰ. 서 론

현재 국내에서 사용되고 있는 대표적인 플레이어로는 그림 1처럼 GRETECH사의 곰플레이어, 판도라TV의 KMP플레이어, Daum의 다음팟플레이어 등이 점유율을 유지하며 사용되고 있다[1].

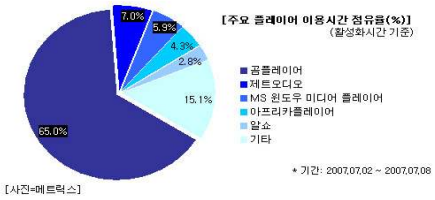

그림 1. 국내 플레이어 이용현황

개발자를 위한 SDK(Software Development Kit)를 공개한 플레이어는 KMP플레이어 하나뿐이다. 또한 SDK를 공개하였지만 시스템 코덱을 사용하지 않고, KMP플레이어에서 독자적으로 사용하는 전용 코덱을 설치하여 사용하도록 권장하고 있으며, 개발자가 개발한 별도의 필터를 테스트하기 위해서는, 플레이어의 렌더러와, 연동 가능한 인터페이스를 개발하여야 한다[2].

이렇게 개발된 필터는, KMP플레이어가 설치되어 있지 않은 시스템에서는 구동의 안정성을 확신할 수 없으며, 현재 여러 업체에서 제작하고 있는 여러 형태에 대한 디코더의 호환성도 보장할 수 없다[3].

본 논문에서는 개발자가 개발한 필터를 직접

테스트 해볼 수 있고, 시스템 코덱에 대한 셋팅 및 환경을 직접 다룰 수 있도록 연구 개발한다. 또한 윈도우 기반의 MFC프로젝트에서 DLL파일로 작성하여 DLL파일을 Import만하여 손쉽게 사용할 수 있는, 플레이어 API를 개발하고, Direct Show 기술을 쉽게 적용하여 시스템 코덱에 대한 전반적인 셋팅을 손쉽게 익힐 수 있도록 한다.

II. 관련연구

2.1. Direct-Show

DirectShow기술은 Video Capture의 주요기술인 VfW(Video for Window)를 모체로 하고 있다. Video Capture기술은 초기에는 단순히 비디오 편집을 위해 입력된 영상을 파일에 저장하는 목적이었으며, PC에서 멀티미디어의 요구가 늘어나면서 단순 저장보다는 화상회의 같은 다양한 어플리케이션 형태로 발전하게 되었다.

하지만, VfW기술은 이러한 기술을 충족하기에는 부족하며, 벤더들은 자사의 독점기술을 이용하여 VfW를 구성하기 시작하였다.

결론적으로, 시장의 표준이 무너지고 호환되지 않는 기술들이 난립하면서 사용자의 불편이 가중되었다. 이를 해결하기 위하여 그동안 벤더들이 독자적으로 확장한 VfW기술을 흡수하고, 윈도우에 흩어져있는 각종 멀티미디어 기술들을 통합하는 DirectShow기술을 발표하게 된다.

2.2. Windows SDK

SDK는 헤어파일, 라이브러리, 사용예제와 설명서까지 포함되어있는, 개발도구이다. Microsoft에서는 Windows를 새로운 발표하면서, Platform SDK, Windows SDK라는 이름들로, 개발도구를 함께 발표해왔다.

DirectShow기술 또한, Platform SDK에 포함되어 배포되었다가, VS2005를 발표하면서, Windows XP와 함께, Windows SDK에 포함되어 배포되게 된다. 최근 개발도구로 사용되어지는 VS2008을 설치하면, v6.0의 Windows SDK가 설치되고, VS2008 SP1의 경우, v6.0A버전의 SDK가 설치되게 된다. 가장 최근 발표된 SDK버전은, Windows SDK for windows Server 2008 and .Net Framework3.5와 함께 배포된 v6.1이며, 최근 발표되고 있는 SDK에는 모두 DirectShow에 관련된 라이브러리가 포함되어있기 때문에, 개발의 편의성을 증대시키고 있다.

III. 동영상 플레이어의 설계

3.1. 동영상 플레이어 설계 분석

본 논문에서 제안한 동영상 플레이어의 설계는 윈도우 기반의 MFC프로젝트에서 DLL파일로 작성하여 DLL파일을 Import만하여 손쉽게 사용할 수 있는, 플레이어 API를 개발하고, 시스템 코덱에 대한 전반적인 셋팅을 손쉽게 익힐 수 있도록 한다.

3.2. 동영상 플레이어 프로그램 설계

동영상 플레이어에 대한 기본적인 프로그램 설계는 그림 2와 같다.

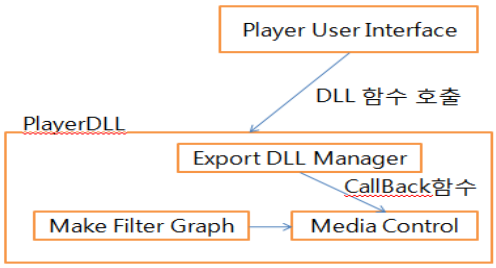

그림 2. 동영상 플레이어 설계

Player User Interface의 경우, MFC SDI프로젝트[4]로 생성하여, 플레이 되고 있는 화면을 렌더링할 수 있는 DC를 얻어올 수 있으면 된다.

또한, 재생, 일시정지, 프레임이동, 화면캡처 등의 기능이 기본적으로, 포함되어 있기 때문에, 몇가지 버튼을 추가하여, 위 기능들과 함께 테스트할 수 있도록 한다.

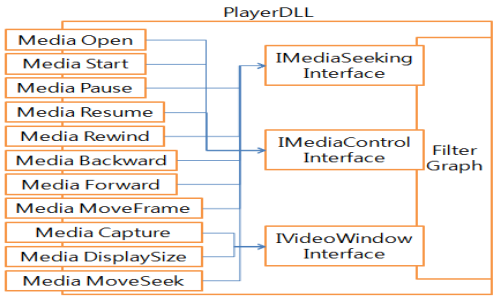

그림 3. DLL 파일 설계

시스템에서 사용하는 DLL은 총 12개의 인터페이스를 가지고 있으며, 그림 3과 같이 위 API중 Rewind와 Backward의 경우, DirectShow기술 자체가 일방통행적인 미디어 스트림의 이동이기 때문에, 뒤로 재생에 있어서는 많은 어려움이 있다[4]. 현재 재생중인 스트림에 대해서, 메모리에 저장하고, 뒤로 재생을 할 경우, 메모리 부족으로 인해, 동영상 플레이를 원활히 진행할 수 없어DirectShow 기술을 사용한 플레이어의 경우 이를 지원하는 경우가 없다. 따라서 임의 초전, 임의 프레임 전으로 이동을 지원함으로써 뒤로 재생을 지원할 수 있도록 한다[5].

현재 존재하고 있는 여러 타입의 미디어들에 대해서 지원하기 위해 그림 4처럼 Filter Graph를 구성하는 Basic Stream클래스를 작성하고, 이를 상속받아, 미디어 타입별, 각기 다른 필터를 셋팅하도록 한다. 동영상 플레이어의 경우, 미디어의

특성에 영향을 많이 받기 때문에, 같은 타입일 경우라도, 디코더가 틀리다면, 해당 디코딩을 지원하는 Demux를 셋팅해줄 필요가 있다[6].

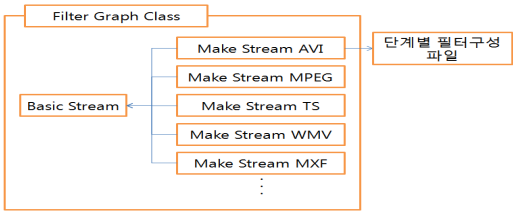

그림 4. Filter Graph 구성도

또한, 필터가 구성되는 단계마다 파일을 남겨, 사용자가 필터의 구성단계를 눈으로 확인 가능하도록 한다.

IV. 동영상 플레이어 구현 및 테스트

4.1. 동영상 플레이어 프로그램 설계

본 논문에서 설계한 플레이어를 개발하고 테스트하기 위한 사양은 다음과 같다.

Visual Studio 2005, Windows SDK 6.1, K-lite Codec Pack, Microsoft Windows XP Professional Version 2002 Service Pack 3, Intel(R) Core(TM)2 Duo CPU P8400 @ 2.26 GHz 1.58GHz, 2GB RAM, 500G HDD

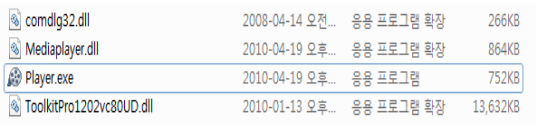

그림 5. 원시파일 구조도

작성된 플레이어의 원시파일은 그림 5와 같다. User Interface와 프로젝트의 실행파일인 Player.exe와 Filter Gerraph를 작성하고, DirectShow Interface를 담당하고 있는 Mediaplayer. dll파일과 MFC프로젝트라이브러리 dll파일이 존재한다. 실제 동작에 필요한 것들은 실행파일과, Mediaplayer.dll파일만 필요하며, 사용자는 Mediaplayer.dll만 로드하여, 사용하면 다른 플레이어를 제작할 수 있다. 해당 DLL파일을 로드하는 부분을 살펴보면 그림 6과 같다.

```
TCHAR *currentDirectory = new TCHAR[MAX_PATH];
int size = sizeof(TCHAR)*MAX_PATH;
ZeroMemory(currentDirectory, size);
GetModuleFileName(AfxGetInstanceHandle(), currentDirectory, size);
szPath = currentDirectory;
delete []currentDirectory;

while(szPath.Right(1)!=_T("₩") && szPath.GetLength() )
    szPath.Delete(szPath.GetLength()-1,1);

CString szDllPath = szPath + szDllName;
m_hDll= LoadLibrary(szDllPath);
if(NULL == m_hDll)
{
    CString szMsg;
    szMsg.Format(_T("Error Loading DLL = %s"),szDllPath);
    AfxMessageBox(szMsg);
}

m_lpfnArchiveClose  = (LPFN_CQCLOSE )GetProcAddress(m_hDll,"CQClose" );
m_lpfnArchiveStart  = (LPFN_CQSTART )GetProcAddress(m_hDll,"CQStart" );
m_lpfnArchiveOpen   = (LPFN_CQOPEN )GetProcAddress(m_hDll,"CQOpen" );
m_lpfnArchiveResume = (LPFN_CQRESUME)GetProcAddress(m_hDll,"CQResume" );
m_lpfnArchivePause  = (LPFN_CQPAUSE )GetProcAddress(m_hDll,"CQPause" );
m_lpfnArchiveRewind = (LPFN_CQREWIND )GetProcAddress(m_hDll,"CQRewind" );
m_lpfnArchiveBackward = (LPFN_CQBACKWARD )GetProcAddress(m_hDll,"CQBackward" );
m_lpfnArchiveForward  = (LPFN_CQFORWARD )GetProcAddress(m_hDll,"CQForward" );
m_lpfnArchiveMoveFrame = (LPFN_CQMOVEFRAME )GetProcAddress(m_hDll,"CQMoveFrame" );
m_lpfnArchiveCapture  = (LPFN_CQCAPTUREFRAME )GetProcAddress(m_hDll,"CQCapture" );
m_lpfnArchiveDisplaySize = (LPFN_CQDISPLAYSIZE )GetProcAddress(m_hDll,"CQDisplaySize" );
m_lpfnArchiveMoveSeek  = (LPFN_CQMOVESEEK )GetProcAddress(m_hDll,"CQMoveSeek" );
```

그림 6. DLL 함수 로드

DLL에 포함되어 있는 인터페이스를 콜백함수로 등록하는 부분이다. 해당 DLL파일의 경로를 가져와서, LoadLibrary함수를 이용하여 DLL파일을 로드하고, 각각의 함수 포인터를 등록한다. 이후에는 콜백함수를 호출하는 것으로써, DLL연동이 가능하다.

그림 7. 실행화면

프로그램을 실행하면, 그림 7과 같은 화면이 나오고, 미디어를 재생할 수 있다.

4.2. 동영상 플레이어 분석

미디어 재생에 성공하면, 그림 8과 같은, 필터그래프 파일을 확인할 수 있다. 그림 8은 Grapha01.grf부터, GraphFinal.grf파일까지 미디어가 재생되기까지 필터들의 연결 과정을 나타낸다.

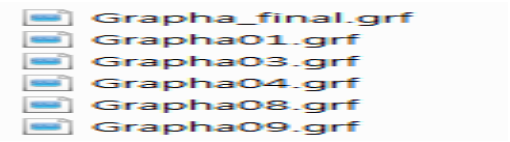

그림 8. 필터그래프 구성 확인파일

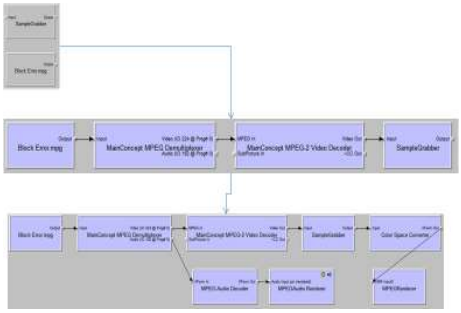

그림 9. 필터그래프 확인

작성된 필터그래프 파일을 확인하면, 그림 9와 같다. 기본적으로 시스템에 설치되어 있는 여러 가지 코덱을 이용하여, 필터들이 등록되고, 연결되는 과정은 Windows에서 작동을 한다. 필터그래프를 생성하고, 필터를 등록하는 과정은 다음과 같다.

```
hr = CoCreateInstance(CLSID_SampleGrabber, NULL, CLSCTX_INPROC_SERVER,
    IID_IBaseFilter, reinterpret_cast<void**>(&pGrabberFilter));
hr = pGrabberFilter->QueryInterface(IID_ISampleGrabber,
    reinterpret_cast<void**>(&m_pGrabber));
hr = m_pGraph->AddFilter(pGrabberFilter, L"SampleGrabber");
```

그림 10. 필터 등록 및 그래프 삽입과정

그림 10은 화면 캡처를 위한, Sample Grabber 필터를 그래프에 등록하는 과정이다. CoCreateInstance함수를 이용하여, Sample Grabber필터의 CLSID를 등록하고, 필터그래프 클래스의 AddFilter 함수를 이용하여, 필터를 등록하면, 그림 8에서와 같이, SampleGrabber필터가 등록된 것을 확인할 수 있다.

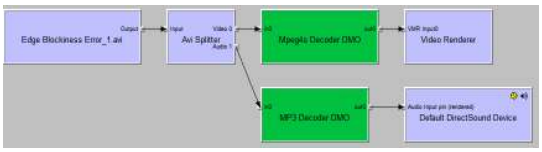

그림 11. AVI파일의 필터구성도

AVI파일의 경우, 그림 11과 같은 필터구성을 필요로 한다. Splitter필터와 디코더, 렌더러는 윈도우를 설치하면, 설치되는 기본적인 필터이며, 각각의 CLSID는 레지스트리에 등록되어 있어, 그림 9의 과정을 이용하면, 손쉽게 필터를 구성하고, 미디어를 재생할 수 있다.

Ⅴ. 결 론

현재 많은 플레이어들이 사용되고 있으며, 일반 사용자도 별 어려움 없이 미디어를 재생하고, 편집하고 있다. 하지만, 미디어는 계속 발전하고 있으며, 현재는 FullHD를 위한, 여러 가지 디코딩, 인코딩장비, 파일래퍼, 파일규칙 등이 생기고 있는 현실이다. Sony나, Panasonic 등의 기타 유명한 방송장비는, 고유의 파일특성을 가지고 있으며, 해당 파일을 디코딩할 수 있는 특수한 디코더를 개발하여, 해당 디코더가 없으면, 미디어의 재생이 불가능하게 된다[7].

따라서 디코더나, Demux가 없는 일반 사용자는 미디어 재생에 불편함을 갖게 되고, 이러한 불편함 때문에 여러 가지 종류에 대한 필터개발이 중요하게 되었다. 하지만, 재생 필터를 개발함에 있어서, 개발의 어려움보다는 필터의 안정성 및, 호환성을 테스트할 수 있는, 플레이어가 국내엔 KMP SDK밖에 존재하지 않고, 해당 SDK역시, KMP플레이어에서 사용하는 전용 디코더를 시스템에 설치함으로써, 원하는 결과를 얻기가 매우 힘들다[8].

본 논문에서, 작성된 미디어플레이어는 사용자가 현재 자신의 필터셋팅을 확인하고, DirectShow기술에 대해 흥미를 가질 수 있게 하며, 자신이 개발한 필터에 대해서 손쉽게 테스트 할 수 있다.

향후 연구는 3차원 동영상플레이어에 대한 연구를 하여 현실감 있는 동영상 플레이어가 실감형으로 재생[9]되는 연구가 필요하다.

참고문헌

[1] 주요 플레이어 이용현황, http://www.etnews.co.kr/news/detail.html?id=200707180089, 인터넷 사이트 리서치 메트릭스, 2007. 7.

[2] 동지연, 박선화, 엄성용, “DirectShow 프로그래밍을 위한 C 소스 코드 자동 생성 기법”, 정보과학회논문지, 컴퓨팅의 실제 제 10권 제 1호, 2004. 2.

[3] 최상길, 김희선, 김상옥, 이광의, 지동해, “MPEG - 4 프리젠테이션 시스템에서의 비디오 랜더링”, 한국정보과학회 1999년도 봄 학술발표논문집 제 26권 제 1호(B), 1999. 4.

[4] 이은성, 최성종, 박민식, “DirectShow 필터를 이용한 DSM-CC Object Carousel 인코더의 설계 및 구현”, 2003년도 한국방송공학회 정기총회 및 학술대회, 2003. 11.

[5] 남기현, 장덕호, 문영식, “DirectShow를 이용한 비디오 특수 효과 개발”, 한국정보과학회 1998년도 가을 학술발표논문집 제 25권 제 2호(Ⅱ), 1998. 10.

[6] 노승경, 이시진, “Linux에서의 통합형 멀티미디어 플레이어 설계 및 구현”, 한국인터넷정보학회 2007 임시총회 및 춘계학술발표대회 제 8권 제 1호, 2007. 6.

[7] 이윤주, 이석필, 조위덕, 김상옥, “MPEG-4 플레이어에서 객체 우선 순위에 의한 장면 구성”, 정보과학회논문지 : 소프트웨어 및 응용 제 30권 제 3 4호, 2003. 4.

[8] 김상옥, 신용경, 이윤주, “지능형 TV를 위한 미디어 플레이어 개발”, 한국정보과학회 1999년도 봄 학술발표논문집 제 26권 제 1호(B), 1999. 4.

[9] 유가용, “영상세대를 위한 DVD 플레이어 디자인 개발”, 한국콘텐츠학회논문지 제 9권 제 11호, 2009. 11.