UNIVERSIDADE LUSÓFONA
de Humanidades e Tecnologias
Humani nihil alienum

Departamento de Informática

Licenciatura em Engenharia Informática

2009

Plataforma Gráfica de Broadcast

David Salvador Nr.20065708

Orientado Empresa: Alexandre Fraser

Orientado Académico: Inês Oliveira

11/20/2009

Índice

Índice de ilustrações

Índices de tabelas

Resumo

Este projecto consiste no desenvolvimento de uma plataforma de criação de conteúdos gráficos sobre/e com sinal de broadcast para a transmissões televisivas nos formatos de SD (Pal e Ntsc) e HD (720p, 1080i, 1080p).

Consistirá numa API que incorpora um subsistema de plugins, suporte para diversos sistemas de hardwares gráficos (BlackMagic, Bluefish, DVS) e integração com a "Framework V3" actualmente em produção na empresa onde trabalho e desenvolvi este projecto.

O resultado desta API tem como objectivo suportar a criação de diversos produtos, deverá incorporar tecnologias integradas nos mercados de broadcast tais como: transmissão de Fill & Key Playout de Vídeos contínuos, mudança dinâmica de input, suporte de diversos codec's de alta qualidade, suporte de colorspaces de 24 e 32 bits, capacidade de reprodução em RGB, RGB+A e YUV.

Palavras-chave

Video Server; Computer Graphics; Broadcast; Television Technologies; Channel Maker; OnAirGraphics;

Abstract

This project consists in a development of a platform to create graphic contents with and/or over the broadcast feed from the television transmission, in the SD (PAL and NTSC) and HD (720p, 1080i, 1080p).

The goal is to originate an API featuring a subsystem of plugins, support for a variety of different hardware cards (BlackMagic, Bluefish, DVS) and a fully integration with the “Framework V3” technology, propriety of the company where I work and did the development of this project.

The result of this API is designed to support the creation of several products, will incorporate integrated technologies in markets such as broadcast and create products as: continuous video playout with Key & Fill, dynamic change of input, support of high-quality codec's, support for 24 and 32 bits colorspaces, reproduction in RGB, RGB + A and YUV.

Keywords

Video Server; Computer Graphics; Broadcast; Television Technologies; Channel Maker; OnAirGraphics;

Introdução

Este projecto visa a avaliação da cadeira Projecto final de curso 2008/2009 de 3º ano do curso de Engenharia Informática.

O projecto foi desenvolvido com a colaboração da empresa wTVision onde desempenho a função de investigador e programador no departamento de desenvolvimento de plataformas. A wTVision é uma empresa especializada em desenvolvimento de software para o mercado audiovisual, designadamente info-grafísmo, sistemas de corporate TV, sistemas de broadcast e sistemas de recolha de dados (scouting). Com sede em Portugal, está representada em 4 países.

Actualmente com o mercado televisivo em expansão a requerer produtos em HD, surge a necessidade de recorrer a maquinas gráficas para broadcast de novos canais, maquinas essas tais como Wipe Effects (gerador de efeitos gráficos A/V), Vídeo Servers, Ad-Insert e Character Generators.

No seguimento da necessidade de uma solução de grafismo SD/HD de custos reduzidos e comum maior flexibilidade no suporte a diversas soluções de hardware foi proposto a minha entidade patronal o desenvolvimento de uma solução que cubra as necessidades de Wipe Effects, VideoServers e Ad-Inserter.

Sendo o objectivo deste projecto o desenvolvimento de um motor gráfico que permita a criação de produtos tais como: Video Servers, Ad-Insert, Wipe Effects, com a capacidade de ser multi-resolução (PAL/NTSC/HD), independente de hardware e facilmente reutilizado em diversos projectos de software.

Enquadramento

Para a realização deste projecto foi necessária a aplicação de conhecimentos adquiridos durante todo o curso de Licenciatura de Engenharia Informática leccionado na Universidade Lusófona, acrescendo de investigação na área de computação gráfica, computação para televisão e tecnologias proprietárias de diversas marcas.

Conceitos dos módulos:

Videoserver

Módulo que disponibiliza uma api de suporte a um motor de playout de vídeos com suporte para SD/HD e diversos formatos de vídeo e identificação dos media items com os seus atributos. Os formatos usados terão colorspaces de 24 bits, RGB tendo 8 bits por canal de cor ou de 30+2 bits com a utilização de 10 bits de cor por canal, obtendo uma diversidade de cor superior.

Consiste em 2 API's, uma para utilização local que comunica directamente com o hardware e uma api de utilização remota, que apenas identifica os atributos dos vídeos, como por exemplo a sua resolução e a sua duração.

API Remota é composta pelo seguinte diagrama de Classes:

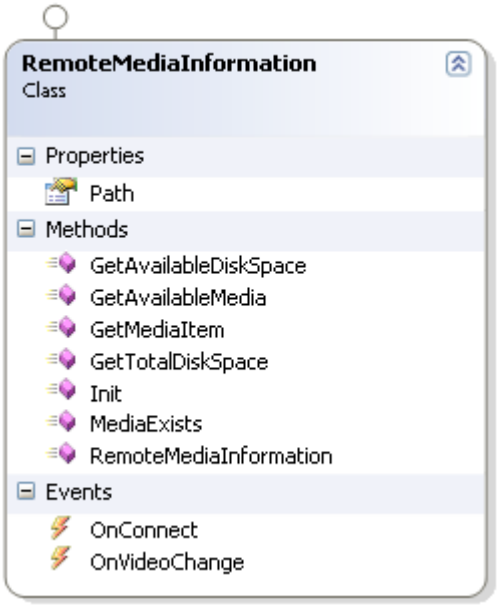

Ilustração 1 – UML da classe RemoteMediaInformation

Esta classe tem como objectivo permitir reconhecer informação relativamente a media item's e estados do servidor sem a necessidade de instanciar a API mais completa com o motor de playout.

API Local é composta pelos seguintes diagramas de Classes:

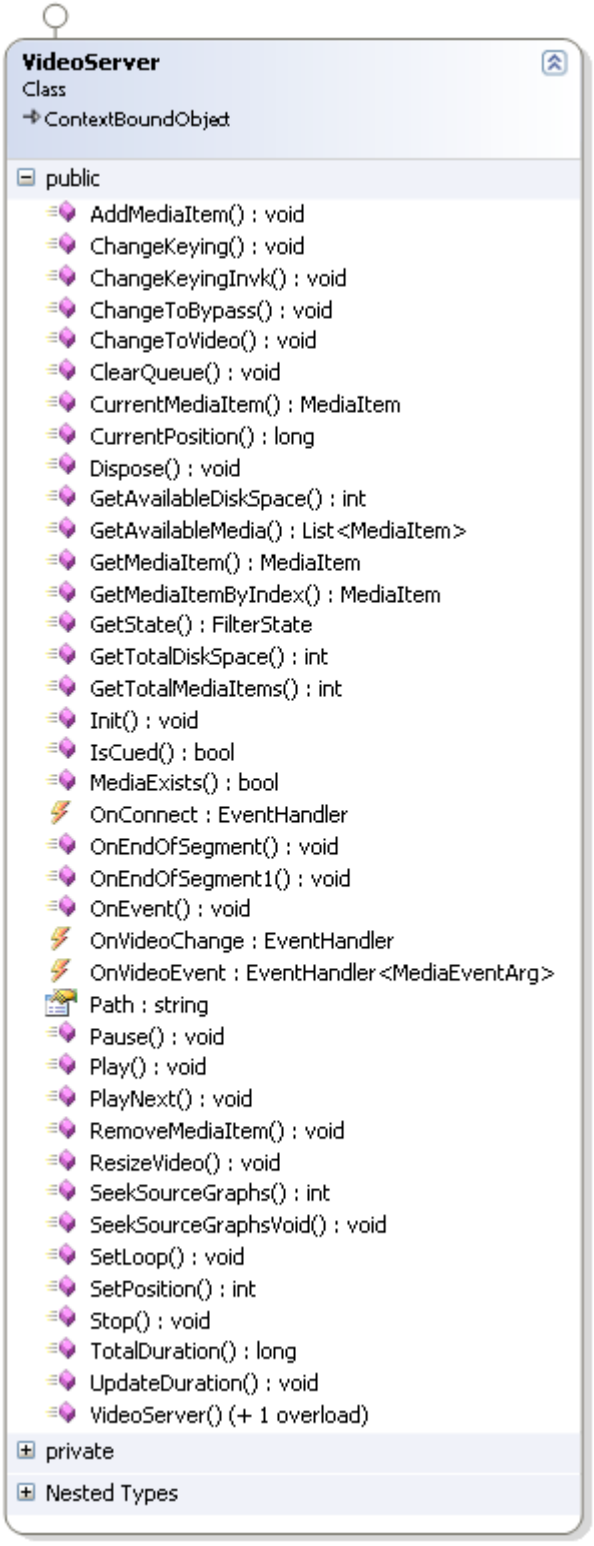

Ilustração 2 – UML da Classe VideoServer (apenas os elementos públicos)

Ad-Insert

O objectivo deste módulo é de substituir a transmissão televisiva por vídeos tocados internamente por um motor de playout quando necessário. Este módulo acrescente ao módulo de videoserver a capacidade de comutar entre live input (internal keyer) e vídeo.

WipeCG

O módulo de wipe é usado para criar efeitos em sobreposição à transmissão televisiva, usa 2 sinais de output, Fill & Key, que quando ligados a um DSK permitem usar o sinal de Key para fazer o "corte" na imagem para poder introduzir cor com o Fill. Este efeito usa um sinal de 32 bits, que é repartido por RGB+A sendo o canal de A (Alpha) usado para calcular o sinal que saíra no output de Key. Assim sendo este módulo dota a plataforma da capacidade de tocar vídeos em 32bits RGBA.

Devido ao facto de ser uma tarefa crítica, este módulo usa vídeos em formatos descomprimidos para obter um menor tempo de processamento e uma maior qualidade, quando neste modo é recomendado que a maquina não seja responsável por mais nenhuma tarefa.

Arquitectura

A arquitectura desenhada para este projecto, assenta numa arquitectura de 3 camadas, sendo a 1º a camada gráfica, a 2ª camada de lógica e regras e a 3º camada de interligação com as api's dos hardwares.

1ª Camada, Gráfica

Para a realização da primeira camada da arquitectura, camada gráfica, foi necessário investigar diversas tecnologias existentes no mercado, dado que o desenvolvimento de uma tecnologia própria teria um custo de desenvolvimento que inviabilizaria este projecto.

As tecnologias analisadas visavam características como: amplo suporte para hardware já existente no mercado, 2D/3D, capacidade de crescimento e suporte de standards gráficos. Depois de uma pesquisa exaustiva no mercado os resultados foram:

- Microsoft DirectX
- Microsoft XNA
- OpenGl
- OpenTk

Após uma investigação a cada uma das tecnologias a escolha recaiu para o Microsoft DirectX, com o seu sub-módulo DirectShow.

Devido ao facto do sistema de directshow ser um sistema modular que suporta diversos tipos de filtros, foi necessário analisar as necessidades de Áudio/Video para desenvolver os filtros/codecs necessário para uma reprodução sem atrasos e com qualidade de broadcast. Ficando como decisão primária o suporte para MPEG2 para as resoluções SD (PAL/NTSC) e MPEG4 para resoluções HD (720p/1080i) a 24 Bits (RGB), em processamento de 32 Bits (RGBA) a decisão recaiu para o formato AVI sem compressão.

DirectShow

Introdução

Microsoft DirectShow é uma arquitectura de streaming media para a plataforma Microsoft Windows. O DirectShow providencia uma alta qualidade de playback e de captura de todo o tipo de fluxos multimédia.

A arquitectura do DirectShow suporta uma elevada variedade de formatos, tais como, Advanced Streaming Format (ASF), Motion Picture Experts Group (MPEG), Áudio-Video Interleaved (AVI), MPEG 2, MPEG 4, MPEG Áudio Layer 3 (MP3), Transport Stream (TS), DivX, Xvid, WAVEform audio format (WAV). Tem suporte de captura usando o Windows Driver Model (WDM) e para dispositivos de captura de vídeo mais antigos.

DirectShow esta integrado nas tecnologias DirectX, automaticamente detecta e usa aceleração áudio e vídeo quando existe suporte do hardware, suportando também qualquer tipo de hardware sem aceleração.

O DirectShow foi desenhado de maneira a simplificar as tarefas de captura, conversão e “playback”. Ao mesmo tempo, providencia acesso a arquitectura de controlo para o desenvolvimento de aplicações personalizadas, permitindo inclusive a criação de filtros DirectShow para suportar novos formatos, efeitos e diversos tipos de transformações.

Existem diversos tipos de aplicações que podem ser desenhadas com a tecnologia DirectShow, tais como, aplicações de edição de vídeo, conversores de formatos, captura de vídeo, sistemas de “Playout” de vídeo. Sendo esta última o tipo de aplicações que deste projecto derivará.

Filtros DirectShow

A aplicação directshow usa uma sequência de filtros para processar a informação multimédia. Tipicamente o fluxo de informação multimédia origina num filtro fonte (source filter) com destino num filtro de destino (render filter). Pode-se acrescentar

filtros intermédios para processar vários tipos de alterações, captura de informação, transformação de cor, transformação de sinal, multiplicação de dados ou mesmo a sua integração com mais de uma fonte de origem, estes filtros são denominados parser filters.

Esta tecnologia permite diversas aplicações como a fonte de informação de dados poderá ser uma fonte local ou localizada num computador na rede ou mesmo uma fonte de informação remota acessível por URL derivada de um website.

Exemplo

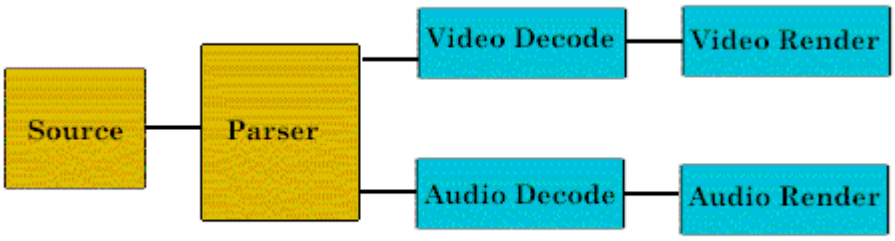

Ilustração 3 – Gráfico de fluxo de DirectShow

Gráfico fluxo DirectShow 1

O exemplo demonstra um esquema típico de montagem de uma combinação de filtros, que contem os seguintes filtros da esquerda para a direita:

- Filtro de origem, responsável pela obtenção da informação multimédia.
- Filtro Parser, separa o sinal misturado em sinal de áudio e de vídeo.
- Filtros de descodificação de áudio e vídeo, responsável por descodificar e descomprimir os dados para o formato apropriado.
- Filtros de finalização, que usa a informação para tocar o som ou mostrar as imagens obtidas.

Os filtros são objectos COM providenciando um conjunto de COM interfaces, desenvolvidos maioritariamente em C++, têm por base o interface IBaseFilter. Contêm

diversos números de "pins", sub objectos do IBaseFilter, que permitem a ligação e troca de dados entre 2 filtros. Na altura da ligação entre os 2 pins, o de origem e o de destino, é negociado um tipo de media type que seja reconhecido pelos 2 filtros que irá ser usado na troca de dados. Caso não existe um tipo de media type comum entre os 2 a ligação entre esses 2 filtros não é criada.

A criação e combinação destes filtros é feita dentro de um objecto do tipo filter graph manager, onde poderá ser aplicado o interface IMediaControl para controlar o fluxo de informação permitindo o inicio, pause, seek e a sua paragem.

Cada filtro reside dentro de uma thread, podendo o mesmo processar os dados que lhe são fornecidos de forma assíncrona e independente. A sequência dos dados é assegurada pelo filter graph manager que assegura que todos os filtros no gráfico recebem os pacotes de dados sincronizador. Qualquer objecto que suporte o interface IreferenceClock pode ser usado como relógio de referência, sendo a precisão na ordem dos 100 nanossegundos.

Tipo de Media

O DirectShow, tal como já foi referido, suporta diversos tipos de formatos, colorspaces e compressões, junto segue uma lista de alguns dos tipos suportados e a sua referência:

Tabela 1 – Exemplos de alguns tipos de colorspaces

Referência	Descrição
MEDIASUBTYPE_RGB1	RGB, 1 bit per pixel (bpp), palettized
MEDIASUBTYPE_RGB4	RGB, 4 bpp, palettized
MEDIASUBTYPE_RGB8	RGB, 8 bpp
MEDIASUBTYPE_RGB555	RGB 555, 16 bpp
MEDIASUBTYPE_RGB565	RGB 565, 16 bpp

MEDIASUBTYPE_RGB24	RGB, 24 bpp
MEDIASUBTYPE_RGB32	RGB, 32 bpp, no alpha channel
MEDIASUBTYPE_ARGB32	RGB, 32 bpp, alpha channel

Tabela 2 – Exemplos de tipos de formados

Referência	Descrição
FORMAT_None or GUID_NULL	None
FORMAT_DvInfo	DVINFO
FORMAT_MPEGVideo	MPEG1VIDEOINFO
FORMAT_MPEG2Video	MPEG2VIDEOINFO
FORMAT_VideoInfo	VIDEOINFOHEADER
FORMAT_VideoInfo2	VIDEOINFOHEADER2
FORMAT_WaveFormatEx	WAVEFORMATEX

2ª Camada, lógica e interoperabilidade

A camada lógica e de interoperabilidade foi criada usando tecnologias Microsoft, nomeadamente usando a .net Framework 3.5. O recurso a uma linguagem de programação C# 3.0, deve-se a esta ser uma linguagem orientada por objectos com a sua flexibilidade, fiabilidade, diversidade e facilidade de obtenção de bons resultados versus tempo de desenvolvimento.

Algumas das técnicas de programação usadas incidem em técnicas de engenharia de software tais como design patterns, metodologias de desenvolvimento, modelação em UML.

Design patterns, surgem como soluções de reengenharia para problemas comuns no mundo do desenvolvimento orientado a objectos, sendo agnósticas às linguagem de programação, fornecem desenhos de arquitecturas práticos e eficazes. Neste projecto foram usadas as seguintes design patterns:

- Factory - Usada para criar classes que suportem as diversas API's de controle do hardware gráfico, permite com uma chamada a um método estático devolver uma objecto derivado de uma classe base conhecida.

```
public static OutputDevice OutputFactory(EOutputDevice outputDevice)
{
    switch (outputDevice)
    {
        case EOutputDevice.Decklink:
            return new DecklinkDirectShowDevice();

        case EOutputDevice.BlueFish444:
            return new BlueFish444DirectShowDevice();

        case EOutputDevice.WindowVideo:
            return new WindowVideoDirectShowDevice();

        case EOutputDevice.DVS:
            return new DVSDirectShowDevice();
    }
    return null;
}
```

Ilustração 4 – Método estático OutputDevice

- Proxy – Usada para estender e compatibilizar as classes derivadas das API's para classes suportadas na 2ª camada.

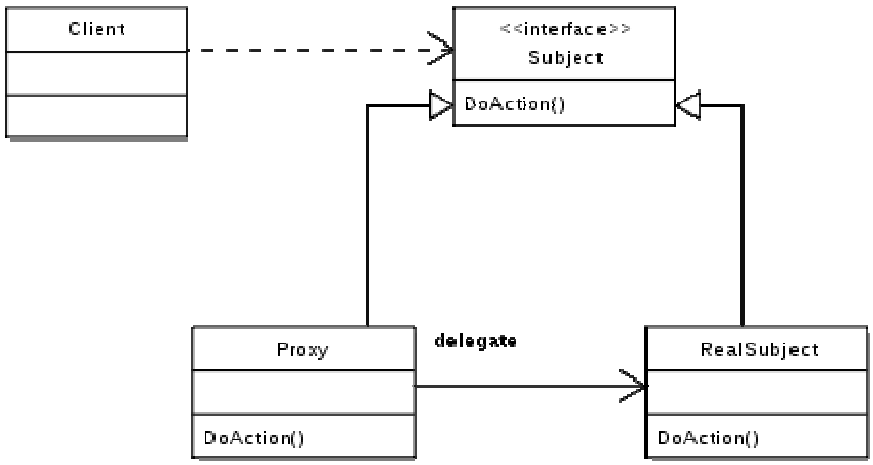

Ilustração 5 – UML Proxy design pattern (wikipedia)

A utilização de uma linguagem orientada a objectos como o C#, permitiu que fossem criadas interops, classes proxy a dll não CLR Complaint.

3ª camada, hardware gráfico

Com base nas diversas características do projecto era necessário encontrar soluções de hardware que suportassem as seguintes funcionalidades:

- Habilidade para reprodução em formatos SD (Pal e NTSC) e formatos HD (720p e 1080i).
- Suporte de referência por referência analógica (BlackBurst) e referência digital (Tri-Level).
- Conexões de vídeo em SDI digital e analógico.
- 1 Input de vídeo SD/HD.
- Capacidade de realizar internal e external Keyer.
- Possibilidade de output de vídeo digital e analógico.
- Saídas de áudio digital, analógico e áudio embebido.
- Disponibilização de uma API com documentação.

Feito o levantamento, o hardware encontrado foi: BlackMagic Design com a sua série Decklink, DVS com o seu produto Centaurus II e BlueFish com 444GREED.

Foram criadas classes wrapper para todas as api's para que essas implementassem os interfaces definidos. Foi necessário recorrer a TLB's e a IDL's para obter as definições das classes em unmanaged code e não CLR complaint, originando interop dessas classes para o seu uso dentro do .net.

Foi criada uma classe abstracta (OutputDevice), que implementa o interface IBaseFilter, de onde todas as classes herdam os métodos e atributos obrigatórios a implementar:

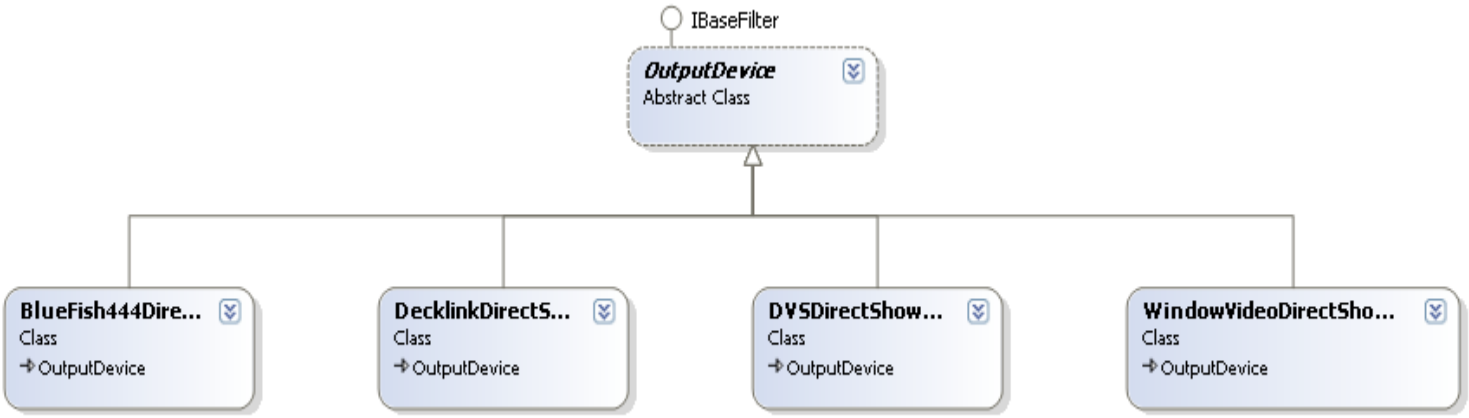

Ilustração 6 – UML das classes que herdam da classe abstracta *OutputDevice*

A utilização da classe abstracta OutputDevice, que implementa o interface IBaseFilter, permitiu que a solução de arquitectura usada na ligação da 2ª camada com a 3ª camada torna-se agnóstica da opção de hardware escolhida e permite que com alguma facilidade seja criadas novas classes para suportar novos hardwares gráficos.

Resultado e Aplicação

Como resultado da arquitectura de 3 camadas, foi obtido uma classe que gere todas as camadas e que é tem uma api de fácil utilização, suporta a criação de diversas aplicações com suporte a diversos hardwares.

Devido ao facto de um dos objectivos deste projecto ser a utilização com o actual software em produção na wTVision, foi criada uma classe que permite converter a classe principal em plugin, sendo possível instanciar dinamicamente e integrar com a Framework V3 como um simples plugin. Sendo que este objecto fica disponível para ser manipulado pela camada de scripting em IronPython existente na Framework V3.

Método

Numa primeira fase foi realizada o levantamento de requisitos necessárias ao funcionamento dos módulos.

Feito a identificação dos requisitos foi realizado um levantamento das capacidades de hardware existentes no mercado. Da análise de todas as soluções de hardware foram seleccionadas as que tinham melhor custo/benefício.

Segue-se o trabalho de investigação sobre os API's disponibilizados pelos fabricantes, com base na documentação disponível, foram realizadas aplicações de testes para comprovar as funcionalidade e consistência das soluções.

No passo seguinte foi realizado um trabalho de investigação sobre os SDK's gráficos existentes no mercado, DirectX, OpenGL, XNA, OpenTK. Depois de análise, foi determinado que o sdk a ser usado seria o DirectX, propriedade da Microsoft.

No desenvolvimento da camada lógica foi necessário ter em conta diversos factores, tais como a interoperabilidade, facilidade de integração com softwares desenvolvidos e em fase de pipeline, integração com a plataforma de trabalho proprietária da wTVision, criação de uma API de fácil utilização e capacidade de utilização em ambientes de broadcast.

Foi criado um plugin para a integração com a Framework V3, que permitirá à wTVision controlar os diversos módulos usando uma camada de código scripting, (IronPython).

Os desenvolvimentos aqui apresentados foram realizados na sua grande maioria na empresa onde trabalho, sendo que algumas das tecnologias utilizadas são propriedade da mesma.

Resultados

Os resultados obtidos neste projecto vão de acordo com os objectivos propostos, sendo que foi possível realizar uma API user-friendly, independente de hardware e com integração em diversas aplicações com a tecnologia proprietária da wTVision.

Com a realização deste projecto foi possível a criação com sucesso de uma vasta gama de produtos SD/HD, tais como:

- Ad-Insert CG, usando os módulos de VideoServer e de Ad-Insert encontra-se actualmente em produção realizando diversos canais para o mercado português

Ilustração 7 – Produto Ad-Insert, criado com base neste projecto.

- Wipe CG, desenvolvido usando o módulo de Wipe, permite a aplicação d vídeos com alpha no downstream da televisão. Este produto já foi usado em diversas ocasiões como: Volta a Portugal em bicicleta 2009, Eleições

Europeias 2009, UEFA Europa League e diversos eventos desportivos em Portugal.

Ilustração 8 – Sequência de imagem do WipeCG

- Video Server: com base no módulo Video Server, usado em diversas situações como Eleições Europeias, Autárquicas e Legislativas.

Conclusões e trabalho futuro

Os objectivos deste projecto eram vasto dado a necessidade do mercado, existiram muitas dificuldades no desenrolar do projecto tais como, desenvolvimento com diferentes tecnologias e falta de documentação, dificuldades que foram ultrapassadas com alguns esforço mas com muita dedicação e vontade.

As diversas decisões tomadas no decorrer do projecto mostraram-se válidas e compatíveis com o resultado desejado.

Como próximos passos, existe a necessidade de ampliar a API para suportar diversos tipos de elementos gráficos, tais como templates 2D e a necessidade de num futuro evoluir para um motor gráfico em 3D e incluir módulos para realização de cenários virtuais e chromas.

Bibliografia

Livros:

Programming Microsoft DirectShow for Digital Video and Television / Mark D. Pesce / Microsoft Press

Pro C# 2008 and the .NET 3.5 Platform / Andrew Troelsen / apress

Practical API Design - Confessions of a Java Framework Architect / Jaroslav Tulach / apress

Head First Design Patterns / Eric Freeman, Elisabeth Freeman / O'REILLY

Design Patterns Explained / Alan Shalloway, James R. Trott / Addison-Wesley

The Pragmatic Programmer / Andrew Hunt, David Thomas / Addison-Wesley

Learning XNA 3.0 / Aaron Reed / O'REILLY

Website:

http://en.wikipedia.org/ – Wikipedia

http://www.opentk.com/ – Official site for Open Toolkit

http://msdn.microsoft.com – MSDN – Microsoft Developer Network

http://www.mpegla.com/ – MPEGLA

http://www.fourcc.org/ - FOURCC.org - Source for video codec and pixel format information.

http://www.smpte.org/ - Society of Motion Picture and Television Engineers

http://msdn.microsoft.com/en-us/aa937791.aspx - XNA

http://msdn.microsoft.com/en-gb/directx/default.aspx - DirectX Developer Center

http://www.microsoft.com/com/ - Microsoft COM

http://www.opengl.org/ - Open GL

http://msdn.microsoft.com/en-us/library/dd375454(VS.85).aspx – Microsoft DirectShow MSDN website

http://www.blackmagic-designg.com – BlackMagic website

http://www.bluefish444.com/ - Bluefish444 website

http://www.dvs.de/ - DVS website

Glossário

Pal – Acrónimo de Phase Alternate Line, designa a forma como os codifica a cor nos sistemas de transmissão televisivas, usado em diversas partes do mundo, tem como resolução 720x576.

NTSC – Acrónimo de National Television Standards Committee, sistema de televisão usado nos Estados Unidos, tem como resolução 720x480

SD – Standart Definition, standard que engloba as resolução standard Pal e Ntsc

HD – High Definition, standard que engloba as resoluções de alta-definição tais como 720p, 1080i e 1080p.

DirectX – API proprietária da Microsoft, permite desenvolvimento de diversos tipos de aplicações, distribuído em todos os sistemas operativos da Microsoft.

DirectShow – Subsistema da plataforma DirectX, propriedade da Microsoft, tem como objectivo lidar com todos as funcionalidade media dos sistemas Windows.

OpenGL – API livre de computação gráfica, permite o desenvolvimento de jogos e aplicações, tem suporte para diferentes sistemas operativos

XNA – plataforma para desenvolvimento de jogos usada para desenvolver jogos para a Xbox360, PC's, dispositivos móveis.

OpenTK – Open Toolkit, é uma livraria que encapsula OpenGL, OpenCL e OpenAL permitindo o uso em linguagens como C#.

BlackBurst – sinal de referência analógica, usada para referênciar as resoluções SD e algumas HD.

Tri-Level – sinal de referência digital, usado para referênciar sinal em HD.

SDI – Serial Digital Interface, interface de cabos de áudio e vídeo com capacidade de transmissão de dados até 3Gb/s, possibilitando a utilização para resolução de SD e HD.

Internal Keyer – Modo de funcionamento de televisão que origina na mistura interna de um sinal de vídeo com dados multimédia emitidos pela mesma placa.

External Keyer – Modo de funcionamento de televisão, onde a transmissão é feita por 2 canais Fill e Key, possível de combinar em DSK.

DSK – Downstream Keyer, equipamento que permite misturar um sinal de vídeo com a utilização de um sinal de Key para o corte sobre o input e introdução do canal de Fill, originando um sinal composto por 2 inputs.

Áudio embebido – distribuição de áudio digital em conjunto com o vídeo digital por intermédio de um cabo SDI.

COM - Component Objecto Model- é um mecanismo de identificação e comunicação entre componentes aplicacionais de uma forma genérica. a comunicação COM é realizada por intermédio de interfaces ou grupos de funções implementados pelo COM object.

Anexos

Enumerados - Enum

Ilustração 9 – Enumerado de modo de operação do Keyer do hardware.

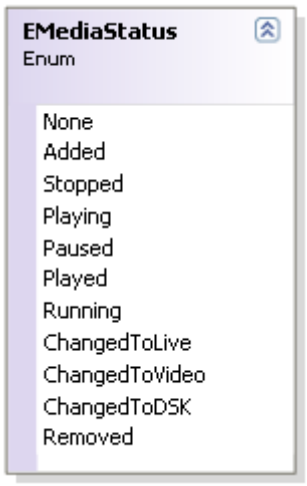

Ilustração 10 – Enumerado de estados de media

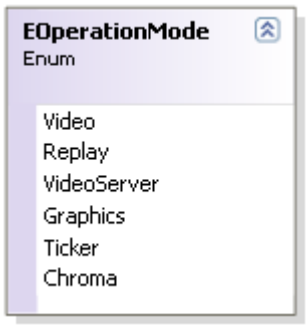

Ilustração 11 – Enumerado de modos de operação

Ilustração 12 – Enumerado do tipo de output devices

Classes dos devices

Ilustração 13 – Diagrama UML das classes de device com herança na OutputDevice, classe abstracta

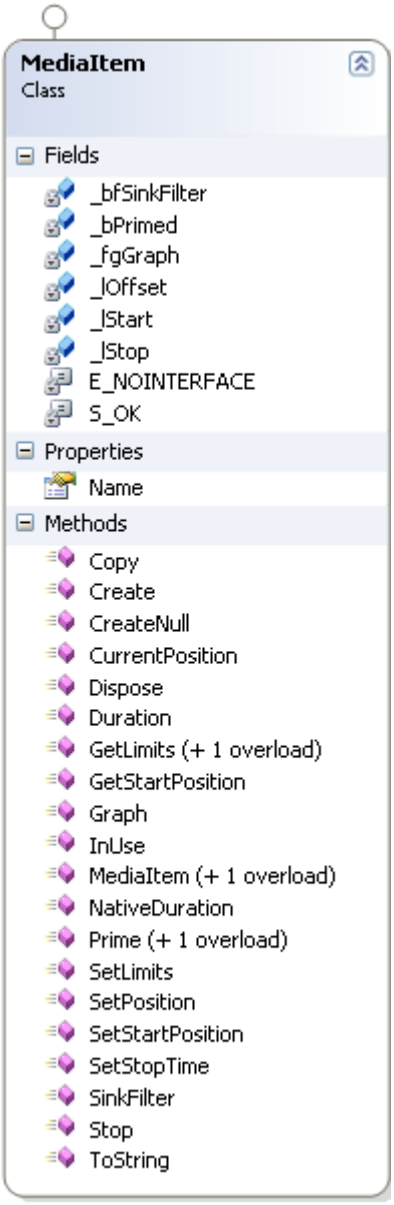

Ilustração 14 – UML do Media Item, originado de um vídeo.

Ilustração 15 – UML da classe estática DirectShowFiltersFactory usada para criar filtros.