DESENVOLVIMENTO E VALIDAÇÃO DE UM PLAYER DE VÍDEO PARA A TV DIGITAL

BECKMANN, Marco; GONÇALVES, Juliano

Grupo de Arquiteturas e Circuitos Integrados
Universidade Federal de Pelotas

BRISOLARA, Lisane; DINIZ, Eliane; AGOSTINI, Luciano

Grupo de Arquiteturas e Circuitos Integrados
Universidade Federal de Pelotas

1 INTRODUÇÃO

Com o avanço das tecnologias de comunicação e a difusão da TV Digital, que permite a introdução de recursos de computação nos sistemas de TV, surge um novo ramo na computação responsável pelo desenvolvimento de aplicativos para a televisão digital. O processo de desenvolvimento de aplicativos para televisão digital é muito semelhante ao usado no projeto de software para computador. Porém, estes aplicativos deverão ser transmitidos juntamente com o sinal de televisão (multiplexadas com áudio, vídeo, legendas, etc.) e executadas num *set-top-box.* A baixa capacidade de armazenamento, processamento e memória dos aparelhos existentes são fatores limitantes para aplicações que precisem usar mais recursos, limitando as possibilidades de extensão dos componentes [MARQUES, 2008].

Atualmente, aplicações para televisão digital são desenvolvidas geralmente utilizando a linguagem Java com bibliotecas de componentes específicos para o desenvolvimento de aplicações para a TV digital. No desenvolvimento dos programas, existem algumas tecnologias que se destacam como, por exemplo, o Java-TV2 [JAVADTV, 2010] e o Ginga [OPENGINGA, 2010]. O Ginga é um midlleware desenvolvido para o Sistema Brasileiro de TV Digital. Este serve como uma camada de software que abstrai detalhes do hardware visando facilitar o desenvolvimento de aplicativos. Para compor o middleware foram propostas duas soluções distintas: uma procedural, o GingaJ [FILHO et al, 2007] suportando aplicativos escritas em Java, e outra declarativa, o GingaNCL [SOARES, RODRIGUES, MORENO, 2007], suportando aplicativos escritos na linguagem *Nested Context Language* (NCL).

Com o objetivo de criar um núcleo de execução comum entre as duas abordagens foi criado o GingaCC que provê métodos fundamentais para o *middleware*, como a sintonização de canais, a exibição de mídias, controle gráfico, controle do canal de retorno em diante. O componente Media Processing, um dos principais componentes do GingaCC, é responsável pela decodificação de áudio e vídeo do middleware. Esse componente foi desenvolvido na linguagem C++ e, atualmente, está em fase de testes para posteriormente ser integrado a outros componentes formadores do middleware GingaCC. Este componente não está restrito ao *middleware* Ginga e pode ser utilizado no desenvolvimento em aplicações que envolvam decodificação de áudio e vídeo.

O objetivo desse trabalho consiste em descrever a experiência de construção de um player de vídeo, utilizando os métodos disponibilizados pelo componente Media Processing. Desta forma, a produção do player será usada para

verificar os métodos fornecidos pelo componente e demonstrar seu uso no desenvolvimento de aplicativos para TV digital brasileira.

2 METODOLOGIA

A seguir serão apresentados os conceitos utilizados no desenvolvimento desse trabalho.

2.1 O Media Processing

O Media Processing é um componente do GingaCC, sendo diretamente envolvido na renderização e exibição de fluxos de vídeo. Este componente recebe um fluxo de dados fornecido pelo Demux, que pode ser composto de vídeos, áudios e legendas, e o decodifica e envia para o componente Graphics, último componente envolvido diretamente na exibição de vídeos, como ilustrado na Figura 1.

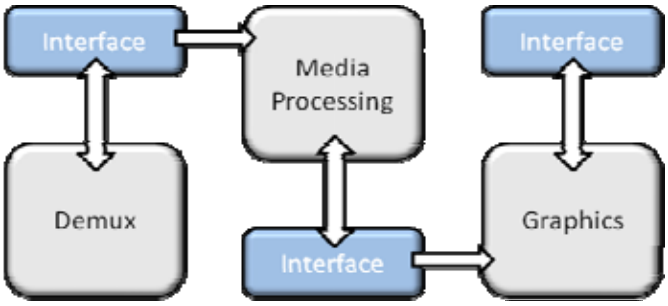

Figura 1. Interconexão entre os componentes Media Processing, Demux e Graphics.

A versão atual da implementação do Media Processing é capaz de suportar fluxos de vídeo e legendas, não provendo suporte a fluxos de áudio. Dentre as principais funcionalidades suportadas pelo componente, estão a alocação de recursos e o controle do fluxo de mídia, o recebimento e decodificação de fluxos mídia em diversos formatos, a carga, seleção e exibição de legendas e a captura da tela. Além disso, o componente provê várias informações sobre o vídeo, como a duração total, tempo de reprodução atual, resolução e taxa de quadros por segundo e suporta a transmissões de vídeo usando os protocolos HTTP, FTP, *User Datagram Protocol* (UDP) e *Real-Time Transfer Protocol* (RTP).

O componente Media Processing foi desenvolvido na linguagem C++ usando a biblioteca libVLC [LIBVLC; 2010]. Para ser integrado com os outros componentes, o modelo de componente utilizado foi o FlexCM [FILHO; 2007].

2.2 Linguagem C++ e o framework QT4

Para o desenvolvimento do player foi utilizada a linguagem de programação C++ juntamente com o framework QT. O QT é um sistema multiplataforma, que provê um conjunto de bibliotecas e ferramentas para o desenvolvimento de programas de interface gráfica. A principal linguagem utilizada no Qt é o C. Neste trabalho foi usada a versão 4 do QT e a ferramenta QtDesigner que facilita a construção de interfaces gráficas, associação de botões e páginas.

3 RESULTADOS E DISCUSSÕES

A partir dos métodos fornecidos pelo componente Media Processing foi desenvolvido um player gráfico para reprodução de vídeo. Além da funcionalidade de reprodução de vídeo, o aplicativo fornece outras funcionalidades relacionadas com a manipulação de mídias e de legendas. O diagrama de casos de uso UML da Figura 2 representa as principais funcionalidades do player.

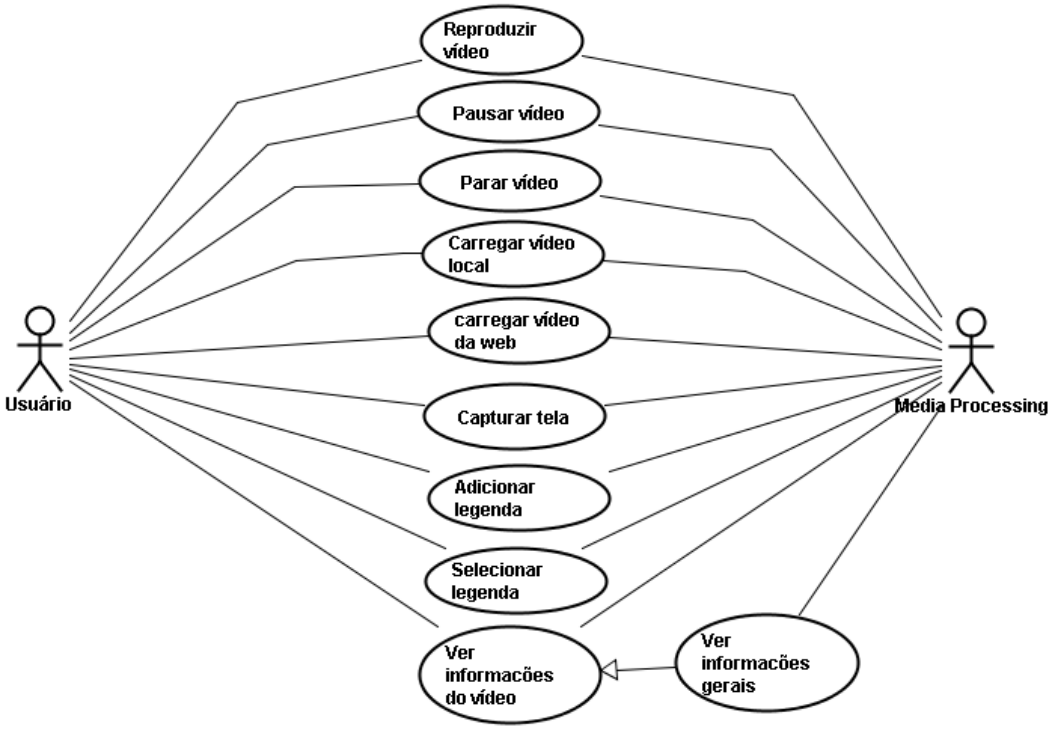

Figura 2. Diagrama de casos de uso UML do *player* de vídeo.

O caso de uso “Reproduzir vídeo” realiza a decodificação do vídeo. Antes de reproduzir o vídeo este deve ser carregado, o que implica em colocar o reprodutor no estado “play” e uma precondição para a execução deste caso de uso é que o player deve estar no estado “stop” ou “pause”. Se outro vídeo estiver sendo reproduzido, será executado o caso de uso “Parar vídeo”, que desaloca os recursos para que o novo vídeo possa ser carregado. O vídeo a ser carregado pode estar localmente armazenado (“Carregar vídeo local”) ou pode estar disponível na web (“Carregar vídeo da web”). O usuário também pode pausar a reprodução, esta funcionalidade está representada pelo caso de uso “Pausar vídeo”, que coloca o reprodutor no estado de “pause”.

O aplicativo permite também adicionar uma legenda, bem como selecionar a legenda a ser usada. No caso de uso “Adicionar legenda”, a nova legenda é adicionada ao vídeo atual. No caso de uso “Selecionar legenda”, o usuário escolhe uma legenda específica para ser reproduzida junto ao vídeo. Uma precondição para a execução deste último caso de uso é que haja um vídeo alocado e uma legenda carregada.

No caso de uso “Ver informações do vídeo” são providas informações sobre a taxa de frames por segundo (FPS), o tempo atual do frame corrente em microssegundos e as dimensões do vídeo (altura e largura do vídeo), além da duração total do vídeo alocado. Já o caso de uso “Ver Informações gerais” uma série de informações como nome do arquivo, artista, álbum, etc, são providas. Por fim, o caso de uso “Capturar tela”, que retira uma *screenshot* do *frame* atual mostrado pelo

player criando um arquivo de imagem no formato JPEG. Uma precondição para a execução do “Capturar tela” é que o player esteja no estado de “play” ou “pause”.

4 CONCLUSÕES

Este trabalho apresentou o desenvolvimento de um *player* através do reuso de recursos do componente Media Processing, desenvolvido para o *middleware* Ginga. Com este trabalho, foi possível verificar o componente e demonstrar que este pode ser reusado para o desenvolvimento de aplicativos que trabalhem com mídias.

Como trabalhos futuros, pretende-se agregar suporte para áudio e algumas outras funcionalidades relacionadas, tais como controle de volume e o seletor de *stream* de áudio. Além disso, planeja-se a realização de comparações de desempenho do player desenvolvido com outros player existentes, além da implementação de outras aplicações reutilizando o player apresentado neste trabalho.

5 REFERÊNCIAS

FILHO, Guido, et al.: GingaJ: The Procedural Middleware for the Brazilian Digital TV System. **J. of the Brazilian Computer Society** vol.12, 47--56 (2007).

FILHO, Sindolfo, et al.: FLEXCM - A Component Model for Adaptive Embedded Systems. In: **31ST IEEE INTERNATIONAL COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE**, pp.119—126. IEEE Press, New York (2007).

JAVADTV **Java DTV API 1.0**, Disponível em: <http://java.sun.com/javame/technology/javatv>. Acesso em: 01 Agosto de 2010

LIBVLC. **“libVLC – VideoLAN Wiki”.** Disponível em: <http://wiki.videolan.org/Libvlc>. Acesso em: 04 Agosto de 2010.

MARQUES, V. C. Desenvolvimento de uma aplicação para TV Digital utilizando o Middleware nacional Ginga-NCL voltado ao aprendizado de crianças. Trabalho de Gradução. UCPel, 2008.

OPEN GINGA **Middleware Ginga-J**, Disponível em: <http://dev.openginga.org/projects/ginga-j>. Acesso em: 03 Agosto de 2010

SOARES, Luis. Fernando; RODRIGUES, Renato; MORENO, Marcelo.: GingaNCL: the declarative environment of the Brazilian Digital TV System. **Journal of the Brazilian Computer Society** vol.1, 37- 46 (2007).