Desenvolvendo, Executando e Controlando Aplicações de Televisão Digital Interativa no SBTVD

Paulyne M. Jucá[1], Andrino Coêlho[1], Carlos Ferraz[2]

[1]Centro de Estudos e Sistemas Avançados do Recife – C.E.S.A.R
Rua Bione, 220 – 50.030-390 – Recife – PE – Brasil

[2]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 7.851 – 50.732-970 – Recife – PE – Brasil

{paulyne.juca, andrino.coelho}@cesar.org.br, cagf@cin.ufpe.br

Abstract. *This paper describes the middleware components necessary to execute digital television applications. These components are responsible for support the development, execution a life cycle control of these applications. After that, a short description of those components developed to support interactivity in a brazilian digital television system (SBTVD) middleware will be shown.*

Resumo. *Este artigo descreve os componentes necessários para possibilitar a execução de aplicações de televisão digital em um middleware. Esses componentes são responsáveis por dar suporte ao desenvolvimento, à execução e ao controle do ciclo de vida dessas aplicações. Depois uma breve descrição desses componentes desenvolvidos para suportar a interatividade em um dos middlewares do sistema brasileiro de televisão digital (SBTVD) será fornecida.*

1. Introdução

O sistema de televisão brasileiro passará por mudanças radicais. Mudanças estas que não dizem respeito apenas à qualidade de áudio e vídeo - que passarão a ser digitais -, mas que introduzirão novos conceitos – TV participativa – e, englobarão novos serviços. Em meio a estas modificações, uma desponta como mais significativa, a interatividade. Dentro deste novo conceito de televisão, o usuário poderá executar diversas aplicações que representarão serviços de comércio (*t-commerce*), bem como jogos, serviços governamentais - voto eletrônico e declaração de isento -, interagindo com todas através do controle remoto de seu televisor.

Neste cenário de convergência e interatividade, faz-se necessário o uso de um terminal de acesso, o *set-top box*. É através deste que o usuário da nova TV poderá manusear, controlar e gerenciar estas aplicações. Isto, contudo, só será possível porque nos terminais existe uma camada de software responsável por adaptar a execução dessas aplicações, o *middleware* – vide Figura 1. É nele que se encontram os componentes responsáveis por executar as aplicações, desenhá-las na tela do televisor, gerenciar os eventos captados pelas mesmas, bem como controlar todas as fases de seu ciclo de vida.

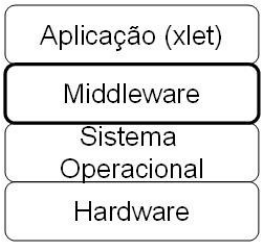

Figura 1: Arquitetura do terminal de acesso.

Neste artigo, apresentaremos os componentes relacionados com a interatividade, procurando detalhar suas arquiteturas, funções e principais características. Introduziremos, então, nossos componentes, tomando como base a especificação de seus precursores, enfatizando as diferenças e melhorias propostas, como parte de nossas contribuições na especificação e desenvolvimento de um dos *middlewares* propostos para o Sistema Brasileiro de Televisão Digital (SBTVD)[14]: o FlexTV[6][7]. Consórcio que contou com a participação de diversas universidades – entre elas o Cin-UFPE –, institutos de pesquisa – como o C.E.S.A.R – e, empresas[1].

2. Trabalhos Relacionados

A televisão digital é contemplada hoje, por diversos padrões, alguns abertos e outros fechados. A maior parte dos estudos é voltada para os padrões abertos, assim como este artigo. Dentre os mesmos, destacam-se o DVB[5], o ATSC[2] e o ISDB[1] - europeu, americano e japonês respectivamente. Cada um destes, define um conjunto de requisitos para a implementação de um *middleware* compatível, cabendo a um órgão interno de cada organização certificar a conformidade das mais variadas implementações. Alguns bons exemplos são as implementações produzidas pelo IRT[2], Thomson[3] e Motorola[4]. Além destas, que são certificadas, existem outras que contemplam parcialmente a especificação de requisitos, sendo de código aberto e/ou gratuitas, como: *Xletview*[15] e o *OpenMHP*[16].

Apoiando-se nos principais padrões já citados, o Brasil criou consórcios, através da FINEP[5], a fim de fornecer apoio técnico à escolha do Sistema Brasileiro de Televisão Digital. Sistema, este, que não só fosse compatível com seus precursores, como que agregasse novas características aos padrões, almejando adequar-se às particularidades do espaço brasileiro. Como já explanado, contribuímos para o consórcio FlexTV, que ficou responsável pela especificação de um dos *middlewares* para o SBTVD. No consórcio, após a definição de uma arquitetura, cada integrante responsabilizou-se pela definição das devidas funcionalidades e, também, pela implementação de cada componente, cabendo à dupla CIn/CESAR os componentes responsáveis pela interatividade.

---

[1] Participaram também deste projeto: LARC-USP, LSI-USP, PUC-Rio, UFRN, UFG, UFPB, Itautec-Philco e Samsung

[2] Institut für Rundfunktechnik, *www.irt.de*

[3] Parceiro de indústrias de Mídia e Entretenimento, *www.thomson.net*

[4] Empresa de celulares, *www.motorola.com*

[5] Financiadora de Estudos e Projetos

3. Os Principais Padrões de Televisão Digital

Essa seção contém uma breve descrição dos principais padrões de televisão digital e também da proposta GEM[8] para a construção de um subconjunto comum entre todos esses padrões. Porém este artigo não pretende servir como base de conhecimento para o estudo dos padrões.

3.1. DVB

Sigla para *Digital Video Broadcasting*. É o padrão de televisão digital europeu que definiu três padrões para a tecnologia de transmissão: via rádio-difusão terrestre (DVB-T), via satélite (DVB-S) e via cabo (DVB-C); bem como um *middleware* próprio: o MHP – *Multimedia Home Plataform*. Este, é baseado em tecnologias como JAVATV[11], DAVIC[4] e HAVi[9], tendo sido completamente desenvolvido em Java™.

Sua maior ênfase é na interatividade, sem esquecer do suporte à qualidade de imagem de alta definição: HDTV – *High Definition Television*. Possui também suporte à transmissão para dispositivos móveis, mas exige uma banda extra (DVB-H).

3.2. ATSC

Sigla para *Advanced Television System Committee*. É o padrão norte-americano que definiu dois padrões técnicos de transmissão: via cabo (ATSC-C) e via rádio-difusão (ATSC-T), sendo o modelo de transmissão a cabo o mais difundido; bem como seu próprio *middleware*: o DASE – *DTV Application Software Environment*.

Esse padrão privilegia a transmissão de alta qualidade (HDTV), contudo não dá suporte à transmissão de televisão digital para terminais móveis, sendo um dos seus grandes pontos negativos.

3.3. ISDB

Sigla para *Integrated Service Digital Broadcasting*. É o padrão japonês, o qual dá suporte às transmissões terrestres (ISDB-T e ISDB-TSB ou *Terrestrial Sound Broadcasting*), via satélite (ISDB-S) e via cabo (ISDB-C). Definiu, também, um *middleware* próprio: o ARIB – *Association of Radio Industries and Business*.

É o mais novo dentre os três principais padrões e seu grande diferencial é a possibilidade de transmitir sinal de televisão para aparelhos móveis como celulares e PDAs dentro da faixa usada para televisão terrestre sem a necessidade da operadora de telefonia, todavia somente em broadcast.

3.4. GEM (Globally Executable MHP)

Os *middlewares* dos padrões de televisão digital eram muito diferentes entre si, o que dificultava muito o desenvolvimento único de aplicações. A fim de se garantir a interoperabilidade de aplicações entre os diversos padrões é que o DVB-GEM[1] foi especificado.

---

[1] Globally Executable MHP

As principais intenções do GEM visam maximizar a interoperabilidade entre as especificações baseadas no GEM de diferentes organizações, maximizar a presença de componentes MHP no *middleware* e, por fim, facilitar o desenvolvimento de aplicações portáveis para os mais diversos *middlewares*.

A mostra a intersecção entre os padrões proposta pelo GEM.

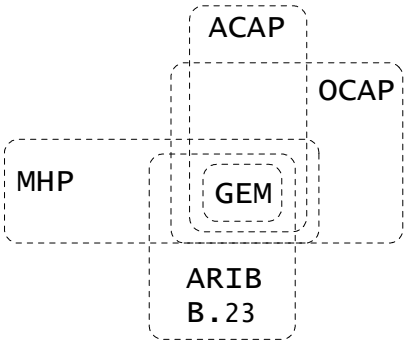

Figura 2: Interseção do GEM e os diversos padrões de TV Digital [13]

4. Arquitetura FLEXTV

O FLEXTV é o *middleware* proposto para o Sistema Brasileiro de Televisão Digital (SBTVD). A arquitetura modular foi definida de forma que permitisse expansões e retiradas de alguns componentes. Um *middleware* flexível – FLEXTV.

Desse consórcio participam diversos institutos de pesquisa, universidades e empresas. Essas instituições definiram então a arquitetura que evoluiu segundo os requisitos levantados por cada participante para seus componentes e gerou a arquitetura abaixo.

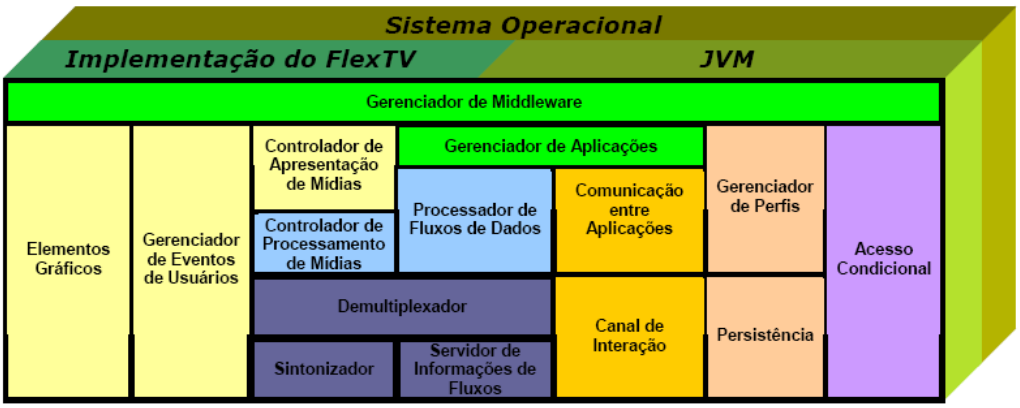

Figura 3: Arquitetura FlexTV [7]

O sistema proposto também levou em consideração a compatibilidade com os sistemas já existentes e adotou todas as recomendações do GEM[8]. Porém diversas inovações foram propostas e desenvolvidas. Um protótipo foi criado e exibido em alguns eventos, sendo a maior parte deles fechados para os contratantes do projeto e membros do governo brasileiro.

5. Elementos Gráficos e Eventos do Usuário

Como o tratamento de eventos do usuário e o desenho de componentes gráficos estão muito interligados, essa sessão apresenta os dois componentes.

Para manter a compatibilidade com o GEM e com isso permitir que aplicações desenvolvidas para o *middleware* brasileiro também pudessem ser executadas nos outros *middlewares* facilitando assim a exportação de aplicações, os componentes gráficos escolhidos para o FLEXTV também foram os da API HAVi [9].

Além do desenvolvimento dos componentes HAVi, foi especificado e desenvolvido outro conjunto de componentes para facilitar o desenvolvimento de aplicações para a televisão digital, acrescentando, por exemplo, funcionalidades para a entrada de texto usando o controle remoto. Esses componentes apenas fazem parte do *middleware* FLEXTV e seu uso inviabiliza a exportação das aplicações uma vez que esses componentes não existem nos outros *middlewares*. Os componentes resultantes das inovações são em sua maioria facilmente exportados em conjunto com as aplicações, porém um pequeno grupo depende das inovações propostas e desenvolvidas no *middleware* brasileiro e podem ser usados apenas por aplicações desenvolvidas para o Brasil.

Já em relação aos eventos dos usuários, os componentes desenvolvidos seguem as recomendação tanto da API do HAVi quanto do DAVIC [4]. Todas as funcionalidades de recepção exclusiva de eventos – para o caso de digitação de senhas, por exemplo –, recepção simultânea e a recepção de eventos para a aplicação com foco foram desenvolvidas.

O mapeamento de eventos do controle remoto foi respeitado e as teclas não existentes no teclado foram acrescentadas de forma a permitir que o usuário possa interagir com a aplicação desenvolvida utilizando ambos dispositivos. O mapeamento para o teclado foi definido especialmente para facilitar o desenvolvimento de aplicações com o uso de um emulador produzido com base nos componentes desenvolvidos. Assim, as aplicações poderiam ser desenvolvidas em estações de trabalho, compiladas e testadas no emulador com um alto grau de garantia de execução no terminal de acesso real.

6. Gerenciador de Aplicações

O gerenciador de aplicações é responsável principalmente por controlar o ciclo de vida das aplicações de televisão digital - *xlets*. Cada aplicação de televisão digital segue um ciclo de vida que incluem estados carregado, executando, pausado e destruído.

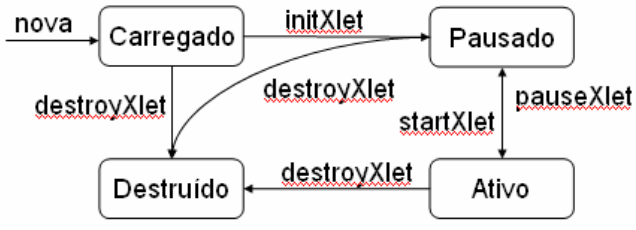

Figura 4: Ciclo de vida de um *Xlet*.

As aplicações de televisão digital são separadas em pequenos pacotes e enviadas diversas vezes via broadcast usando um mecanismo chamado carrossel de dados. É gerado um fluxo elementar que possui dentre outras coisas as informações dessa aplicação. Este fluxo elementar é multiplexado a outros fluxos existentes, sendo enviados em conjunto, via rádio-difusão, para o terminal (*set-top box*) do usuário.

Quando esse fluxo alcança o terminal do usuário, as camadas mais próximas do sistema operacional percebem que um fluxo de dados chegou e que contém uma aplicação. No momento dessa descoberta, o gerenciador de aplicações é então notificado – o sistema trabalha com *listeners* – da chegada da aplicação. Nesse ponto o gerenciador de aplicações carrega seu serviço interno de segurança dos dados, que verifica a integridade da aplicação e de seus respectivos recursos (dados próprios).

É responsabilidade do gerenciador também verificar requisitos de segurança e permissão de acesso para as aplicações a fim de evitar que códigos "mal-intencionados" tenham acesso ao sistema. Faz parte também das atribuições do gerenciador, iniciar a aplicação, executar os comandos, alocar e liberar recursos para a aplicação, pausar, parar e remover a aplicação, quando sua execução chegar ao fim. Esta última ação depende diretamente de configurações realizadas no momento do encapsulamento da aplicação.

6.1. O Gerenciador Proposto

Durante o estudo sobre outros gerenciadores de aplicações[1] tornaram-se evidentes as limitações existentes em cada um, quando isolados. Contudo, ao se unir os conceitos e funcionalidades apresentados nestes, têm-se um gerenciador mais abrangente. Isto aumenta o grau de extensibilidade do sistema, além de facilitar a manutenibilidade e adaptabilidade do mesmo.

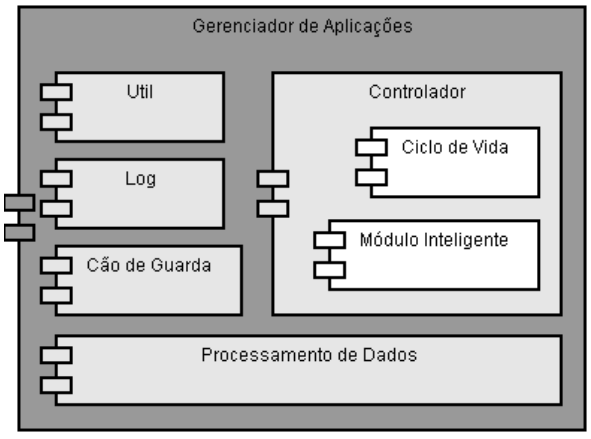

Figura 5: Arquitetura simplificada do Gerenciador de Aplicações

Visando alcançar as características supracitadas, bem como introduzir algumas inovações para capacitar os gerenciadores de aplicações a suportar novos tipos de aplicações e serviços – cada vez maiores e mais complexos – é que foi especificado o

[1] Durante o projeto, foram estudados diversos *middlewares* como: OCAP, MHP e ARIB. Eles foram comparados levando-se em consideração vários aspectos arquiteturais e funcionais. Os resultados desse estudo levantaram algumas funcionalidades desejáveis para o *middleware* proposto.

gerenciador de aplicações do FLEXTV. Este integra aplicações internas, sistema de segurança e certificação, e atua como servidor de serviços interativos. Sua arquitetura simplificada encontra-se na Figura 5.

6.1.1. Cão de Guarda

Este componente foi desenvolvido na forma de um *thread* independente que fica constantemente checando se o gerenciador está respondendo. Caso o gerenciador não responda dentro de um determinado intervalo, o sistema é reiniciado.

6.1.2. Controlador

Este é o núcleo do Gerenciador de Aplicações. As principais funções estão contidas nesse componente. É a partir dele que as decisões são tomadas e as ações delegadas aos respectivos componentes.

O controlador é responsável por notificar o Cão de Guarda da correta execução do sistema, por gerenciar o ciclo de vida das aplicações, iniciar todos os componentes, carregar arquivos e variáveis, associar domínios de segurança específicos a cada aplicação, disponibilizar acesso aos recursos e variáveis compartilhadas do sistema e delegar ações.

Todo Gerenciador de Aplicações deve possuir sua própria AIT e a mesma deve ser mantida e gerenciada pelo Controlador. Logo, em sua inicialização este componente deve carregar a AIT a partir de um arquivo de configuração escrito em XML, usando para isso o componente interno de Processamento de Dados.

Para cada aplicação sinalizada na AIT o Controlador associa um *proxy* e adiciona *listeners* a eventuais mudanças de estado deste *proxy*. Um destes listeners é o módulo do *middleware* que cuida dos eventos de usuário. Além dos *proxys* a AIT, manuseada pelo Controlador, deve possuir registros das aplicações que estão em execução, quais os seus tipos e seus estados.

6.1.3. Log

Componente responsável por armazenar todo o log do sistema. Este log pode ser acessado pelos fabricantes para manutenção do equipamento, verificando erros ocorridos durante o uso.

6.1.4. Processamento de Dados

Este componente trata da leitura e gravação de dados em arquivos de configuração do Gerenciador de Aplicações. Para tanto, são utilizados arquivos no formato XML.

6.1.5. Util

Componentes que não se integram em nenhum outro se enquadram aqui. Pode ser um algoritmo, um manipulador de caracteres, rotinas que se repetem etc.

7. Um Emulador para Ajudar no Desenvolvimento

Todo desenvolvedor de aplicações para televisão digital reconhece que sua maior dificuldade durante o desenvolvimento é o teste da aplicação, ou seja, executá-las em

um emulador. As implementações comerciais dos emuladores custam caro, enquanto que as gratuitas são implementações parciais, o que exclui inúmeras funcionalidades.

A partir do desenvolvimento dos três componentes citados para o FLEXTV, tornou-se possível elaborar e criar um emulador de aplicações. Os principais objetivos para sua criação foram:

- Ter um ambiente onde testar os componentes gráficos desenvolvidos: a necessidade de testar os componentes gráficos desenvolvidos era fundamental a fim de garantir a qualidade e a confiabilidade dos componentes tanto da API HAVi quanto das extensões implementadas.
- Testar o gerenciador de aplicações: Sua integração com os componentes gráficos, bem como seu desempenho na simulação do carregamento de aplicações vindas de um fluxo de elementar. Com esta necessidade, foram criadas diversas pequenas aplicações para fazer todos os testes necessários.
- A integração entre os componentes: desde o início do desenvolvimento, a integração foi se tornando cada vez mais necessária tornando-se, em um certo momento, vital para uso e testes. Como conseqüência, os três componentes (a saber: gerenciador de aplicações, componentes gráficos e eventos do usuário) passaram a fazer parte de uma única linha de desenvolvimento.

Um emulador de aplicações para televisão digital interativa foi o produto final alcançado ao término deste desenvolvimento. Vale ressaltar que este é totalmente compatível com o SBTVD, sendo denominado de “CESARCin Emulator”.

8. Conclusões

A interatividade é fator essencial no futuro da televisão digital. Mesmo que ainda não seja largamente utilizada pelo mundo, com a convergência que inevitavelmente virá o desafio é fornecer mecanismos para possibilitar o desenvolvimento de aplicações e serviços cada vez mais robustos e relevantes para a televisão.

Nesse contexto, os componentes desenvolvidos são de total relevância, pois sintetizam a interatividade no *middleware* FLEXTV. Sem esses componentes, nenhum tipo de interatividade seria possível.

Os desafios de não apenas repetir especificações existentes, mas definir conceitos inovadores e fornecer a base para a inclusão de novas funcionalidades ainda não pensadas foram de extrema importância. Porém, apesar da inovação, a compatibilidade era fator ainda mais relevante no processo.

O trabalho realizado foi de extrema relevância para a pesquisa no país e gerou frutos ainda mais relevantes para a tomada de decisão do padrão adotado no Brasil. Além de formar pesquisadores conscientes das possibilidades de cada padrão existentes, o trabalho realizado e demonstrado nesse artigo serve como base para mais estudos e propostas para a convergência.

Cada componente desenvolvido possibilitou a execução confiável de aplicações interativas robustas, forneceu mais elementos para o desenvolvimento de novas

aplicações e a futura introdução de novos mecanismos de entrada de dados. Em conjunto, deram origem a uma ferramenta que possibilita o teste de aplicações.

9. Agradecimentos

Este trabalho contou com a contribuição dos membros do grupo de televisão digital do C.E.S.A.R e do Centro de Informática da UFPE. Gostaríamos de agradecer a todos e em especial a Mário Fried.

10. Referências

[1] ARIB. "ARIB STD-B10: Service Information for Digital Broadcasting System, ARIB Standards v 3.8". www.arib.or.jp. Junho. 2004.

[2] ATSC. "A/65B: Program and System Information Protocol for Terrestrial Broadcast and Cable". disponível em www.atsc.org. 2003.

[3] DASE. "DTV Application Software Enviroment". Disponível em www.dase.org.

[4] DAVIC. "DAVIC Specification Reference Models and Scenarios". Disponível em www.havi.org. 1998.

[5] DVB. "Digital Video Broadcasting". www.dvb.org.

[6] FLEXTV1. "FlexTv – Visão de Contexto do Middleware". 2005.

[7] FLEXTV2. "FlexTv – Arquitetura Conceitual do Middleware". 2005.

[8] GEM. "Globally Executable Multimedia Home Platform". GEM 1.0.1 - ETSI TS 102 819 V1.2.1.

[9] HAVi. "Level 2 Graphical User-Interface. Specification of Home Audio/Video Interoperability Architecture". Disponível em www.havi.org. 2001.

[10] IRT. "Institut für Rundfunktechnik". www.irt.de.

[11] JAVATV. "Java TV API specification 1.0". Disponível em http://java.sun.com/products/javatv/. Visitado em 23 de dezembro de 2005.

[12] MHP. "TS 102 812 V1.2.1". Disponível em www.mhp.org.

[13] Morris, S. et all. "Interactive TV Standards – A guide to MHP, OCAP and JavaTV". Focal Press, first edition. Abril de 2005.

[14] SBTVD. Chamadas Públicas – Carta Convite para middleware do SBTVD. http://www.finep.gov.br/fundos_setoriais/funttel/funttel_resultado.asp?codSessao=8&codFundo=7. visitada em 23 de setembro de 2005.

[15] Xletview. http://xletview.sourceforge.net. Visitado em 23 de setembro de 2005.

[16] OpenMHP. www.openmhp.org