Um *framework* para o desenvolvimento de aplicações interativas para a Televisão Digital

Adriano Simioni, Valter Roesler

Departamento de Informática – Universidade do Vale do Rio dos Sinos (UNISINOS)
Av. Unisinos, 950 – São Leopoldo – RS – Brasil

adriano@mcm.inf.br, roesler@unisinos.br

Resumo. *Uma das bases da implantação da TV Digital no Brasil é a interatividade, pois o objetivo do governo é obter inclusão social através da inclusão digital, e isso é obtido através da inserção da população à rede via canal de interatividade. Dessa forma, surge a necessidade do desenvolvimento de aplicações interativas visando tirar proveito das vantagens geradas pela TV Digital. Este trabalho apresenta um framework desenvolvido para permitir a criação de interfaces interativas para as aplicações de televisão digital. O objetivo é facilitar o desenvolvimento desse tipo de aplicação, porém sem perder as funcionalidades desejadas pelo programador.*

1. Introdução

A capacidade de penetração da Televisão Digital faz dela a porta de entrada para o mundo da informação de grande parte da população brasileira, que ainda não tem acesso às facilidades da informática e da Internet [RFP 2004]. Segundo IBGE (2005), 90,3% da população possui acesso à televisão, enquanto que apenas 13,2% possui acesso à Internet.

Assim como a Internet, a TV Digital interativa[1] (TVDI) representa a possibilidade de acesso a um mundo virtual de informações e serviços. Ela pode liberar as pessoas da necessidade de possuir um computador em casa e de saber operá-lo, pois a TV Digital interativa encapsula em si um sistema informatizado que é operado como uma TV, por meio de um controle remoto.

Neste cenário de convergência e interatividade, faz-se necessária a criação de recursos que facilitem o desenvolvimento desse tipo de aplicação. Com base nisso, este trabalho visa montar e validar um *framework* que agilize o desenvolvimento de aplicações interativas.

Para executar aplicativos na TV Digital, faz-se necessário o uso de um terminal de acesso, o *set-top box*[2]. É através deste que o usuário poderá controlar e gerenciar estas aplicações. Isso, contudo, só é possível porque nos terminais existe uma camada de software responsável por abstrair a complexidade do hardware, o *middleware* – vide Figura 1, baseada em FERNANDEZ (2004). 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. Resumidamente, a interatividade é de sua total responsabilidade.

---

[1] TV Interativa é a coleção de serviços que suporta escolhas e ações iniciadas pelo telespectador e que são relacionadas a um ou mais canais de programação de vídeo [SCHWALB 2004].

[2] O *set-top box* é responsável por receber o sinal transmitido, decodificá-lo e exibi-lo.

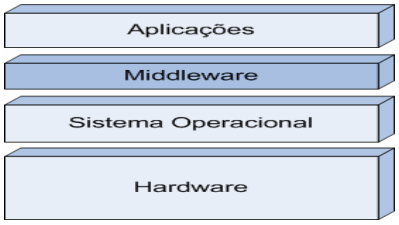

Figura 1. Localização do middleware no sistema do Terminal de Acesso.

Existem diversos *middlewares* hoje no mercado, alguns fechados, como o OpenTV, utilizado em sistemas de TV paga, e outros abertos, como os utilizados nos sistemas de TV Digital padronizados atualmente, que são o MHP (*Multimedia Home Platform*), utilizado no padrão Europeu DVB - *Digital Video Broadcasting Project* [DVB 2003], O DASE (*DTV Application Software Environment*), utilizado no padrão americano ATSC - *Advanced Television Systems Committee* [ATSC 2006], e o ARIB (*Association of Radio Industries and Businesses*), utilizado no sistema Japonês ISDB - *Integrated Services Digital Broadcasting* [ARIB 1997].

Existe uma tendência mundial para a adoção de padrões abertos, dentre os quais a MHP é o padrão mais aceito neste mercado [MORRIS e SMITH-CHAIGNEAU 2005]. Inicialmente, a MHP foi projetada para executar nas plataformas DVB, mas acabou se tornando referência mundial e gerou demanda para estender a interoperabilidade que oferece a outras plataformas de televisão digital [DVB 2005]. Esta demanda originou a GEM (*Globally Executable MHP*) [GEM 2006], uma estrutura que permite a outras organizações definirem especificações baseadas na MHP. Na prática, a GEM estabelece um conjunto de funções (API) comum aos diferentes *middlewares*, permitindo a interoperabilidade entre os Terminais de Acesso desenvolvidos ao redor do mundo. Com base nisso, o *framework* apresentado nesse trabalho se baseia na GEM, com o objetivo das aplicações desenvolvidas com base na mesma poderem ser utilizadas em qualquer Terminal de Acesso compatível com GEM.

Este artigo está dividido da seguinte forma: a seção 2 apresenta o *framework* desenvolvido, enquanto a seção 3 mostra um exemplo de aplicativo para demonstrar o *framework*. A seção 4 tece as considerações finais desse trabalho.

2. Descrição do framework desenvolvido

Existem várias idéias de tipos de serviços a serem disponibilizados na TV Interativa, sendo que muitas são reais e outras tantas estão ainda distantes. Algumas aplicações podem ser desenvolvidas em um futuro próximo, mas muitas ainda dependem de recursos que não estão tão acessíveis, como memória e processamento [MAIOR 2002]. Os aparelhos receptores são equipamentos com baixo poder de processamento e pouca memória, mas mesmo assim, as aplicações devem executar de forma muito rápida.

Considerando as limitações citadas, foi criada uma solução para melhorar o desempenho no transporte e leitura de registros em um repositório de dados desenvolvido especificamente para essa finalidade. Ou seja, o *framework* é responsável por:

- Comprimir dos dados transmitidos visando economia de banda;
- Controlar e gerenciar as informações recebidas no Terminal de Acesso.

Na Figura 2 é exibido de modo geral o contexto da utilização do *framework*, que é composto pela camada "Catálogo", responsável pela gerência da base de dados, e pela camada "Painel", responsável por montar a interface da aplicação.

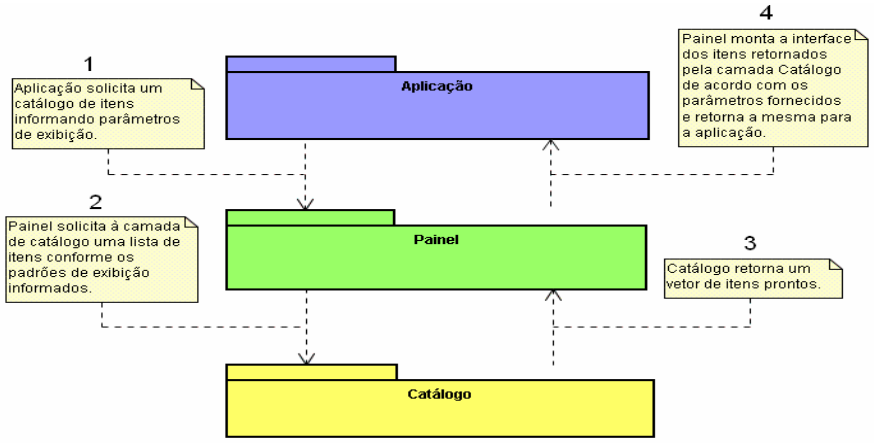

Figura 2. Modelo de relacionamento entre as camadas previstas.

A camada Catálogo irá manusear os arquivos recebidos pelo carrossel de dados e fornecer à camada Painel os itens que forem solicitados. Esta, por sua vez, tem a função de disponibilizar uma interface para utilização pela aplicação. A aplicação utiliza o *framework* através de chamadas de código, conforme exemplo a seguir, onde é solicitada uma interface de 600x800 pixels, uma coluna de 100 pixels, uma linha de 200 pixels e um único item por tela. Basicamente servirá para inserir uma imagem no texto, entretanto, seu uso ficará mais claro na próxima seção.

```
public void MontaCatalogo {
       CatalogIntf catalogo = new CatalogIntf(); //cria objeto
       catalogo.setAltura(600);   //altura da tela (em pixels)
       catalogo.setLargura(800);  //largura da tela (em pixels)
       catalogo.setNro_colunas(1);//número de colunas na tela
       catalogo.setNro_linhas(1); //número de linhas na tela
       catalogo.setColuna(1,100); //tamanho da coluna 1
       catalogo.setLinha(1,200);  //tamanho da linha 1
       catalogo.setQuantTela(1);  //número de itens por tela
       catalogo.setOrdem(catalogo.LINHA); //ordem de exibição
       catalogo.setOperImagem(catalogo.CORTAR); //cortar se exceder tamanho
       catalogo.exibir(); //exibe na tela o catálogo pronto
}
```

A Figura 3 ilustra uma possibilidade de utilização do *framework* desenvolvido, onde a aplicação apresenta informações financeiras de acordo com o perfil do usuário.

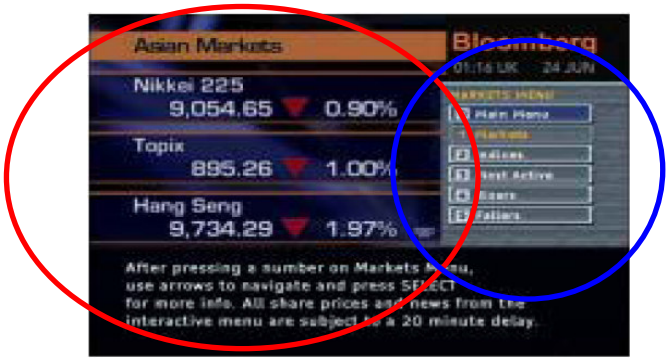

Figura 3. Personalização de Informações.

Para a criação desta interface, o desenvolvedor deverá efetuar duas chamadas ao método que irá criar o catálogo. Uma delas informando parâmetros para a criação da tabela contida no círculo à esquerda e outra para a que está no círculo à direita.

A figura 4 ilustra as classes da camada "Catálogo", descrita anteriormente. A descrição de cada uma delas é efetuada a seguir.

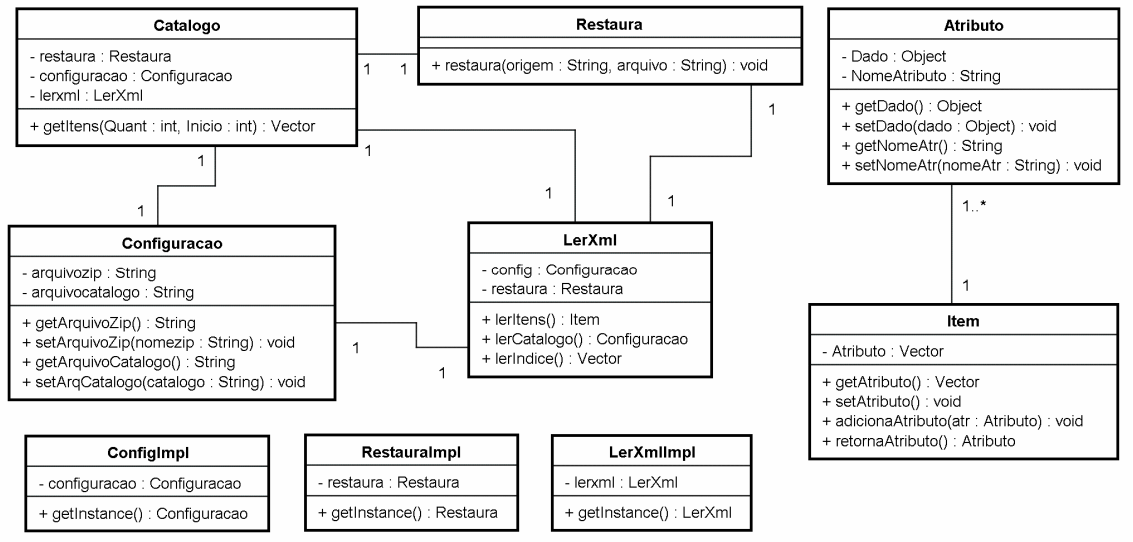

Figura 4. Classes responsáveis por manter a camada inferior do framework, chamada de Catálogo.

- Classe Atributo: responsável por armazenar os atributos que irão compor cada item a ser exibido. Segundo ETSI (2003), as extensões dos arquivos suportados pela MHP são as seguintes: jpg, png, gif, mpg, drip, mp2, txt, xml, css, txt, js, sub, tlx, pfr, class e svc.

- Classe Item: agrupa todos os atributos referentes a cada item que será exibido, assim como suas propriedades de exibição e comportamento.

- Classe Configuração: no início da aplicação esta classe será preenchida com o nome do arquivo compactado, assim como o índice para as referências de arquivos.

- Classe LerXml: responsável por realizar a leitura dos itens contidos no arquivo compactado. Cada item é um arquivo xml.

- Classe Restaura: capaz de restaurar sob demanda arquivos contidos no arquivo compactado. A tarefa de descompactação é feita sob demanda para utilizar o mínimo possível dos recursos de hardware.

- Classe Catálogo: articula o processo de colaboração entre as outras classes, fazendo com que os itens solicitados sejam disponibilizados à camada superior.

- Classes ConfigImpl, RestauraImpl e LerXml: possuem um método estático que fornece a referência a um objeto das respectivas classes.

3. Aplicativo exemplo desenvolvido

Para demonstrar os resultados obtidos até o momento, foi criado um aplicativo de

comércio eletrônico, cuja finalidade é comprar livros através da televisão. Duas telas do aplicativo são vistas na figura 5, mostrando o resultado obtido com a utilização do *framework* numa aplicação simples.

O usuário pode navegar entre os diversos itens disponibilizados através das setas disponíveis no controle remoto, ou ainda posicionar-se nos botões inferiores da aplicação que indicam direção para a direita e para a esquerda e ativá-los com o botão de seleção do controle remoto. Neste instante, a aplicação irá chamar o método *nextItem()*, contido na camada Painel, cuja finalidade é ativar o método *getItens()* da camada Catálogo, para pegar um novo item e recriar a interface, substituindo o primeiro ou o último item, conforme a escolha.

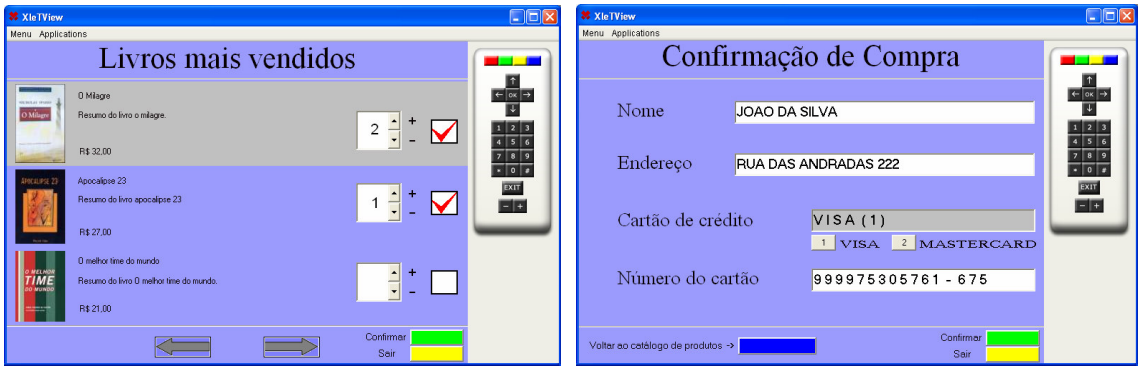

Figura 5. Interface de seleção de itens no aplicativo de comércio eletrônico.

Enquanto o usuário navega sobre os itens, o item selecionado fica com cor diferente. Então, se o usuário deseja efetuar uma compra, utiliza os botões "+" e "–" (ou qualquer tecla a ser definida) do controle remoto para realizar um incremento ou decremento, conforme o caso, na quantidade de livros desejada para cada livro.

Após o usuário escolher os itens desejados, ele efetua a compra. Para isso, é necessária a utilização do aplicativo para permitir a entrada dos dados cadastrais do usuário, visto na tela da direita. Como esse aplicativo exige uma comunicação segura com o provedor de serviços, está sendo desenvolvido em paralelo um *framework* de segurança, resolvendo essa situação através de TLS (*Transport Layer Security*), também disponível na MHP.

4. Considerações finais

Este trabalho apresentou diversas questões referentes à disponibilização de aplicativos interativos no Terminal de Acesso, e propôs um modelo de *framework* que facilite o desenvolvimento de aplicações interativas.

Tal *framework* foi implementado em linguagem Java, sendo compatível com a especificação GEM, permitindo que os aplicativos desenvolvidos com o *framework* sejam disponíveis mundialmente, em qualquer Terminal de Acesso compatível com a GEM ao redor do mundo.

Também apresentou-se um aplicativo desenvolvido com o *framework*, visando esclarecer o potencial de utilização do mesmo.

5. Referências

ARIB – *Association of Radio Industries and Businesses*, 1997. Especificação e documentação do padrão ARIB. Disponível em http://www.arib.or.jp/english. Acessado em junho de 2006.

ATSC – *Advanced Television Systems Committee Inc*, 2006. Especificação e documentação do padrão ATSC. Disponível em http://www.atsc.org/standards.html. Acessado em junho de 2006.

DVB. *Digital Vídeo Broadcasting Project*. 2003-2006. Site oficial do projeto DVB. Disponível em: http://www.dvb.org. Acessado em junho de 2006.

DVB-MHP. *Digital Vídeo Broadcasting Multimedia Home Plataform*. 2003-2006. Site oficial da plataforma MHP. Disponível em www.mhp.org. Acessado em maio de 2006.

ETSI TS 102 812 - Digital Video Broadcasting (DVB) – *Multimedia Home Platform (MHP) Specification 1.1.1*, June 2003.

FERNANDEZ, J.; LEMOS, G.; SILVEIRA, G. *Introdução à Televisão Digital Interativa: Arquitetura, Protocolos, Padrões e Práticas*. In JAI-SBC. Salvador, 2004.

GEM – Globally Executable MHP, *A Guide to Platform Harmonisation*. Disponível em: http://www.mhp.org/mhp_technology/gem/wp05.platform%20harmonisation.final.pdf. Acessado em maio de 2006.

HARTE, Lawrence. *Introduction to IP Television (IPTV)*. EUA: ALTHOS, 2005.

IBGE - Instituto Brasileiro de Geografia e Estatística. *Síntese de Indicadores Sociais 2004*, Rio de Janeiro, 2005.

INTERACTIVETVWEB. *Using the return channel*. Disponível em: http://www.interactivetvweb.org/tutorial/mhp/returnchannel.shtml. Acessado em maio de 2006.

JUCÁ, P.; COÊLHO, A.; DUARTE, R.; FERRAZ, C. *Comparação entre o desenvolvimento de aplicações MHP e OpenTV*. In SBRC2006, Curitiba, 2006.

MAIOR, Marcelo Souto; *TV Interativa e seus caminhos*. Universidade Estadual de Campinas, São Paulo, 2002.

MORRIS, S.; SMITH-CHAIGNEAU, *A. Interactive TV Standards*, *A guide to MHP, OCAP, and Java TV*. EUA: Elseiver, 2005.

RFP - Requisição Formal de Proposta Nº 14/2004 – *Canal de Interatividade* (anexo à CARTA CONVITE MC/MCT/FINEP/FUNTTEL - Nº 14/2004), 2004.

SCHWALB, Edward M. *iTV Handbook Technologies and Standards*. New Jersey: Prentice Hall, 2004.