1
Introdução

A possibilidade de se encapsular dados, juntamente com o áudio e vídeo, em sistemas de TV Digital, abre espaço para uma vasta gama de oportunidades capaz de proporcionar uma maior interatividade para o telespectador. Essa integração permite que a programação seja enriquecida através da disponibilização de diversos serviços, como: guias eletrônicos de programação, votações, jogos, acesso a outras redes de comunicação, entre outros serviços. Essa possibilidade, contudo, exige um maior poder computacional da parte dos receptores do sinal de TV digital para que, desta forma, tenham a capacidade de executar as aplicações para eles desenvolvidas.

Devido à enorme diversidade de fornecedores de terminais de acesso, torna-se necessária, a fim de promover uma execução global das aplicações desenvolvidas pelos produtores de conteúdo, a criação de uma abstração denominada *middleware*. O *middleware* fornece um conjunto comum de Interfaces de Programação de Aplicativos (*Application Programming Interfaces* – APIs) a serem oferecidas por todos os receptores que, entre outras facilidades, permitem o uso dos diversos recursos disponíveis nesses dispositivos.

Dentre os sistemas de TV Digital terrestre em operação, os seguintes padrões de *middleware* figuram como principais:

1. O europeu, denominado *Multimedia Home Platform* (MHP) (ETSI, 2005b), definido pelo grupo do DVB (*Digital Video Broadcasting Project*);
2. O americano, conhecido como *DTV Application Software Environment* (DASE) (ATSC, 2003a), definido pelo grupo ATSC (*Advanced Television Systems Committee*); e
3. O japonês, *Integrated Services Digital Broadcasting* (ISDB) (ARIB, 2004), definido pelo grupo ARIB (*Association of Radio Industries and Businesses*).

No âmbito brasileiro, encontra-se em desenvolvimento o *middleware* a ser utilizado pelo sistema de TV digital deste país, batizado como Ginga, e parte do Sistema Internacional para Difusão Digital Terrestre (*Terrestrial International System for Digital Broadcasting* – ISDB-T), também conhecido como SBTVD.

1.1. Motivação

O *middleware* europeu MHP, que contou com uma rápida popularização, adotou em seu ambiente procedural a linguagem JAVA (Gosling, 1996). Com isso, foi criado um nível maior de portabilidade para suas aplicações, permitindo o reuso de componentes de software provenientes de outras plataformas. Surgiram, então, algumas iniciativas para sua implementação sobre outras plataformas internacionais. Entidades responsáveis pela padronização de sistemas de TV Digital, fora da Europa, manifestaram o interesse em ter parte do *middleware* MHP implementado em suas especificações. Dessa forma, seria aproveitado todo seu desenvolvimento tecnológico e mantida uma compatibilidade que permitisse que as novas aplicações, como aquelas já desenvolvidas, pudessem ser executadas/apresentadas em terminais de acesso desses outros padrões.

Diante do interesse em se fazer uma integração entre os *middlewares* procedurais já existentes surgiu, então, o *Globally Executable MHP* (GEM) (ETSI, 2005a). O GEM, que pode ser considerado como um acordo de harmonização entre os principais padrões existentes, tem como base o MHP. Isso porque, além de capturar um subconjunto das interfaces, e toda a semântica definida por este padrão de *middleware*, o GEM também inclui as necessidades impostas por outros padrões internacionais.

Com o GEM despontando como ambiente de execução procedural mínimo para os principais padrões de TV Digital (MHP, DASE e ARIB) e de mídia empacotada (como o *Blue-Ray Disc*), tem-se aí uma base para a criação de aplicações e programas interativos portáveis para diversas plataformas. *Middlewares* procedurais compatíveis com a especificação do GEM vêm sendo adotados por vários países no mundo todo (MHP, 2006), inclusive pelo Brasil.

Contudo, no GEM, como mencionado, é definido apenas um ambiente de execução procedural, deixando livre para a implementação a definição de um

possível ambiente declarativo. Na Figura 1 são mostradas as especificações de ambientes declarativos adotados pelos principais padrões que implementam o GEM. Nessa figura percebe-se, ainda, uma série de recomendações propostas pelo órgão ITU (*International Telecommunication Union*).

O ITU-T e ITU-R, em conjunto com outras organizações, como DVB, ARIB, ATSC, OpenCable e SMPTE, se organizaram para consolidar uma tendência de harmonização que já vinha sendo demonstrada desde o estabelecimento do GEM. A recomendação ITU-T J.200 (ITU-T, 2001) foi criada visando promover uma arquitetura de alto nível para um conjunto de formatos e APIs mais harmônicos, capazes de prover várias funcionalidades necessárias para aplicações interativas mais avançadas a serem oferecidas pelas redes de televisão para os usuários domésticos. É o núcleo comum dos ambientes de aplicação para serviços de TV Digital interativa. A recomendação estabelece tanto um ambiente de execução declarativo (detalhado em ITU J.201) (ITU-T, 2004) quanto procedural (detalhado em ITU-T J.202) (ITU-T, 2003). Contudo, esses ambientes não precisam ser necessariamente independentes; podendo ser definidas pontes entre eles.

A Figura 1 mostra a especificação ITU-T J.201 como compreendendo todo o ambiente declarativo das três principais implementações de *middleware*. Apesar da especificação prever um núcleo comum, mostrado no centro da área pontilhada, é prevista sua extensão para certos requisitos específicos da implementação.

O núcleo é formado por:

- Módulos definidos pelo padrão *XHTML Modularization*, especificado pelo W3C (*World Wide Web Consortium)* (W3C, 2002);
- O padrão CSS para descrever o estilo da apresentação, definido pelo W3C (W3C, 1998);
- A API DOM para alterar dinamicamente o conteúdo dos documentos XHTML, definida pelo W3C (W3C, 2004a); e
- ECMAScript definido pela ECMA *International* (ECMA, 1999).

Ainda na Figura 1, assim como no ambiente declarativo, a especificação do ambiente procedural prevê um núcleo comum que é justamente o *framework* proposto pelo GEM. A especificação também permite a extensão desse núcleo

comum apresentando, em caráter informativo, os três padrões anteriormente mencionados.

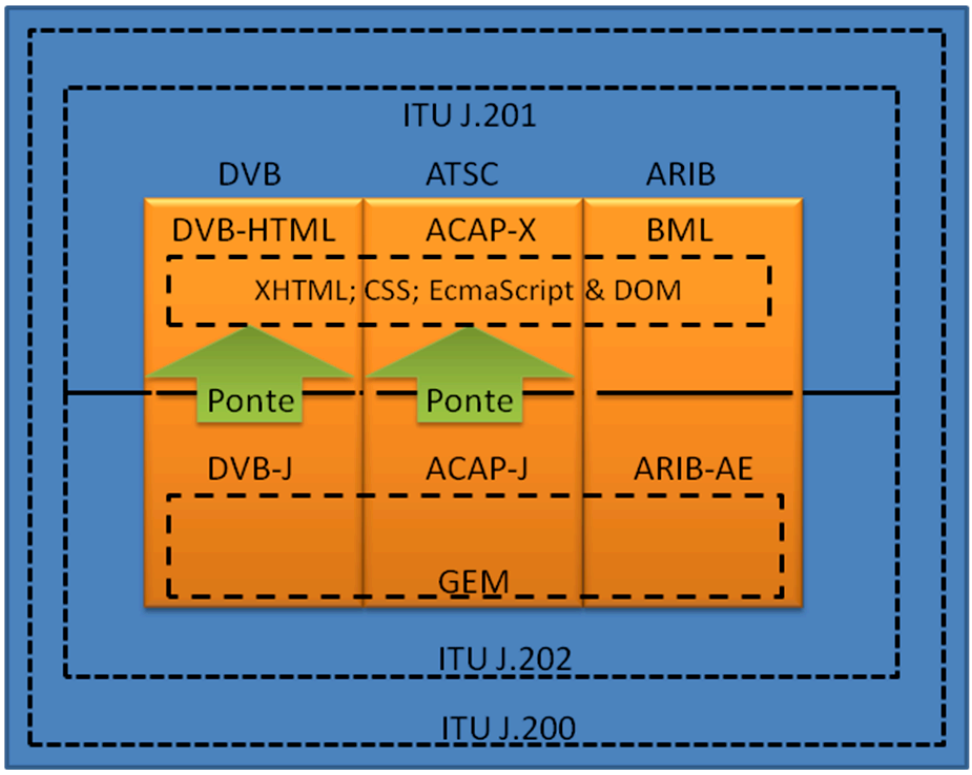

Figura 1 – O GEM e os demais padrões.

Embora exista uma tendência clara à harmonização do *middleware* procedural, realizada pelo GEM, o mesmo não se pode dizer, de fato, com relação ao *middleware* declarativo. Apesar dos três padrões da Figura 1 adotarem um ambiente baseado no padrão XHTML, com exceção do sistema japonês, esses ambientes ainda não foram comercialmente implementados a contento.

As limitações do modelo XHTML, a serem discutidas a seguir, e a falta da implementação de um ambiente declarativo encorajam a criação de novos mecanismos que possibilitem a execução de aplicações/programas dessa natureza, através da criação de máquinas de apresentação (declarativas) baseadas na API procedural comum estabelecida pelo GEM.

Na definição do subsistema declarativo do *middleware* brasileiro foi padronizada a utilização da linguagem NCL (*Nested Context Language* – Linguagem de Contextos Aninhados) (Soares & Rodrigues, 2006). Ao contrário dos padrões citados anteriormente, a NCL possui o foco no sincronismo de mídias e não é baseada no padrão XHTML. No entanto, essa linguagem mantém a compatibilidade com os demais padrões por permitir a definição de um exibidor

XHTML que viabiliza a utilização de conteúdo desta natureza como mídia a ser sincronizada pelo documento NCL.

Apesar de ser possível a reprodução do conteúdo declarativo dos demais padrões pelo *middleware* brasileiro, o contrário não é possível. De forma a viabilizar tal compatibilidade, uma das alternativas é utilizar a API procedural fornecida pelos padrões internacionais para a criação de um ambiente de apresentação declarativo para documentos NCL, que é o foco principal desta dissertação.

1.1.1. Ambiente Declarativo x Ambiente Procedural

Em um sistema de TV digital, um ambiente de execução procedural é aquele capaz de executar programas criados seguindo o paradigma de programação imperativo. Esses códigos apóiam-se, basicamente, na execução algorítmica de instruções fornecidas pelo programador. Dessa forma, o programador possui um maior controle da execução, sendo capaz de estabelecer todo o fluxo de controle e execução de seu programa. Contudo, é necessário que o programador seja mais qualificado e conheça bem os recursos da linguagem.

Ambientes de apresentação declarativos, por sua vez, executam códigos criados seguindo o paradigma de programação declarativo. No âmbito dos quatro padrões anteriormente mencionados, incluindo o brasileiro, suas linguagens declarativas são todas baseadas em XML (*eXtensible Markup Language*). Entre os exemplos de linguagens declarativas baseadas em XML (chamadas de aplicações XML) encontram-se a linguagem NCL (*Nested Context Language* – utilizada pelo SBTVD-T), a linguagem SMIL (*Synchronized Multimedia Integration Language* – padrão W3C para sincronismo temporal de mídias) (W3C, 2005) e o XHTML, sendo esta a mais difundida atualmente.

Ao contrário da linguagem procedural, que realiza a decomposição do problema em implementações algorítmicas, a linguagem declarativa enfatiza a declaração descritiva da solução do problema. A linguagem declarativa é mais indicada para aplicações cujo foco casa com o objetivo específico para o qual a linguagem foi desenvolvida. Na maioria dos casos, para TV Digital, as aplicações (não apenas aquelas carregadas pelo canal de difusão, mas também aquelas

carregadas pelo canal de retorno) lidam com a sincronização de objetos de mídia. A interação do usuário e a sincronização temporal com o áudio e vídeo principal são exemplos de relacionamentos comuns entre objetos presentes nessas aplicações.

1.2. Objetivos

Este trabalho tem como objetivo propor um ambiente de apresentação de programas declarativos desenvolvidos na linguagem NCL para sistemas que implementem o GEM.

No ambiente "virtual" implementado são utilizadas apenas as APIs JAVA fornecidas pelo framework GEM. Seu desenho foi concebido de forma a ser capaz de operar sobre plataformas de recursos limitados.

A solução proposta tem como base o Formatador NCL da implementação de referência do *middleware* Ginga realizada pelo Laboratório TeleMídia da PUC-Rio. Contudo, foi necessária uma adaptação da implementação existente para *desktops,* no intuito de permitir a compatibilidade com o GEM, sem perder contudo a conformidade com o ambiente declarativo do Ginga.

Além do formatador, alguns de seus exibidores de mídia também foram adaptados.

Como uma das grandes dificuldades encontradas ao se executar aplicações em ambientes de TV Digital é o seu tempo de iniciação, foram pesquisados mecanismos para otimização do tempo de iniciação da execução do documento. Um destes mecanismos é o carregamento modular do formatador. No desenvolvimento realizado, apenas os módulos do formatador necessários para a apresentação em questão são obtidos do fluxo MPEG-2 Sistema. Além disso, quando o receptor permitir o armazenamento de aplicações ou *plugins*, é priorizado o carregamento do módulo pré-existente no receptor em detrimento daquele transportado no fluxo MPEG-2 Sistema. Para realizar o processo de implantação de tais módulos, foi criada uma estrutura independente. Essa estrutura possui a capacidade de abstrair dos demais módulos o acesso ao fluxo

MPEG-2 para o carregamento de bibliotecas e outros recursos necessários para a apresentação de um documento hipermídia.

Outro mecanismo de otimização utilizado é a pré-conversão do documento NCL. Esse mecanismo visa diminuir o número de etapas necessárias pelo receptor para iniciar a execução de um documento, além de eliminar a necessidade de uma biblioteca para o processamento de documentos XML.

1.3. Organização da Dissertação

O restante desta dissertação encontra-se organizada como a seguir. No Capítulo 2 são expostos conceitos acerca dos principais *middlewares* procedurais existentes sendo, a seguir, apresentado o GEM como harmonização de tais *middlewares*. Após a apresentação do GEM é mostrado o primeiro padrão de mídia empacotada a implementá-lo: o *Blue-Ray Disc*. É trazida, então, ao conhecimento do leitor a recomendação da máquina virtual Java mínima para ambientes procedurais de sistemas de TV Digital. A seguir, são apresentadas as tecnologias envolvidas na construção de um ambiente declarativo para sistemas que implementem o GEM.

O Capítulo 3 apresenta os trabalhos relacionados com o sistema proposto. Inicialmente são discutidas as alternativas para a implementação dos middlewares declarativos baseados no XHTML como aplicações dos middlewares procedurais correspondentes. A seguir, são discutidas duas aplicações para TV Interativa que fazem uso de uma máquina de apresentação de documentos SMIL no ambiente de execução do *middleware* MHP. Ao final do capítulo, são apresentados os resultados de testes realizados por uma implementação SMIL em um sistema de *broadcast* de TV Digital.

O Capítulo 4 apresenta a implementação Java do Formatador NCL desenvolvida pelo laboratório de Telemídia da PUC-Rio e que serviu como base para a elaboração deste trabalho, uma vez que o foco desta dissertação está na especialização do Formatador NCL como uma aplicação GEM. A seguir, são mostradas todas as adaptações necessárias à implementação Java do Formatador NCL para a execução em um ambiente compatível com o GEM.

O Capítulo 5 apresenta uma proposta de um ambiente declarativo baseado na linguagem NCL para sistemas que implementem o GEM. Nesse capítulo é descrita a arquitetura e a implementação do sistema proposto. É mostrado como integrar tal implementação a sistemas MHP utilizando as facilidades introduzidas por sua interface de *plug-ins* e armazenamento de aplicações. Ao final do capítulo, são apresentados os resultados de alguns testes realizados.

O Capítulo 6 apresenta uma análise dos resultados obtidos e sugestões para suscitar trabalhos futuros.