PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Programa de Pós-Graduação em Informática

EXPLORANDO O CANAL DE RETORNO EM SISTEMA DE TELEVISÃO DIGITAL INTERATIVA: uma abordagem centrada no suporte à comunicação entre aplicações e provedores de serviços

Rogerio Baldini das Neves

Belo Horizonte
2010

Rogerio Baldini das Neves

EXPLORANDO O CANAL DE RETORNO EM SISTEMA DE TELEVISÃO DIGITAL INTERATIVA: uma abordagem centrada no suporte à comunicação entre aplicações e provedores de serviços

Dissertação apresentada ao Programa de Pós-Graduação em Informática como requisito parcial para obtenção do Grau de Mestre em Informática pela Pontifícia Universidade Católica de Minas Gerais.

Orientador: Lucila Ishitani
Co-orientador: João Benedito dos Santos Júnior

Belo Horizonte
2010

FICHA CATALOGRÁFICA

Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

Neves, Rogério Baldini das

N518e Explorando o canal de retorno em sistema de televisão digital interativa: uma abordagem centrada no suporte à comunicação entre aplicações e provedores de serviços / Rogério Baldini das Neves. - Belo Horizonte, 2010.

117f. : il.

Orientadora: Lucila Ishitani.

Dissertação (Mestrado) - Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-graduação em Informática.

Bibliografia.

1. Televisão digital - Teses. 2. Televisão interativa - Brasil. I. Ishitani, Lucila. II. Pontifícia Universidade. Católica de Minas Gerais. III. Título.

CDU: 681.3.023:621.397

Bibliotecário: Fernando A. Dias - CRB6/1084

PUC Minas
Programa de Pós-graduação em Informática

FOLHA DE APROVAÇÃO

"EXPLORANDO O CANAL DE RETORNO EM SISTEMA DE TELEVISÃO DIGITAL INTERATIVA: uma abordagem centrada no suporte à comunicação entre aplicações e provedores de serviços"

ROGÉRIO BALDINI DAS NEVES

Dissertação defendida e aprovada pela seguinte banca examinadora:

Profa. Lucila Ishitani - Orientadora (PUC Minas)
Doutora em Ciência da Computação - UFMG

Prof. João Benedito dos Santos Júnior - Co-orientador (PUC Minas)
Doutor em Ciência da Computação e Matemática Computacional - USP

Prof. José Marcos da Silva Nogueira - (UFMG)
Doutor em Engenharia Elétrica - UNICAMP

Prof. Zenilton Kleber Gonçalves do Patrocínio Junior - (PUC Minas)
Doutor em Ciência da Computação - UFMG

Belo Horizonte, 13 de maio de 2010.

Aos familiares e amigos, que souberam entender minhas ausências.

Saibam que podemos conseguir tudo que queremos.

Mas só conseguimos quando temos o apoio das pessoas que acreditam e confiam em nós.

AGRADECIMENTOS

Agradeço a todos aqueles que, de alguma forma, contribuíram para o desenvolvimento deste trabalho.

À minha esposa que sempre me dá amor e carinho para que eu tenha forças para trabalhar e viver uma vida muito feliz.

Aos meus pais, irmão e familiares próximos que me amam incondicionalmente, apoiando a todo momento e entendendo o motivo da distância, e as ausências em momentos especiais.

Aos amigos que cobram, confiam e me ajudam a fazer dessa vida um parque de diversões.

Aos meus orientadores, pela paciência, sabedoria e consciência em me guiar por situações difíceis.

E agradeço a Deus pela saúde e tranquilidade para terminar mais essa jornada, que continue me iluminando para que eu consiga ter sucesso na jornada de ser pai da minha lindinha Marina.

RESUMO

Com a evolução das tecnologias, que envolvem o ambiente de geração, transmissão e apresentação da programação, somada à criação e execução de sistemas para TV Digital Interativa Brasileira (TVDI), surgem demandas de produtos e serviços relacionados a essa nova plataforma. As aplicações para TV Digital Interativa apresentam um novo paradigma de desenvolvimento, tanto no que diz respeito às ferramentas de desenvolvimento quanto às funcionalidades como usabilidade e comunicação de dados. Um estudo sobre os padrões envolvidos na TV Digital Brasileira foi realizado visando a criação de uma camada de *software* relacionada à comunicação de aplicações com o canal de retorno, para prover serviços de interação em *middleware* genérico. Neste trabalho são apresentadas as estratégias para o desenvolvimento de um sistema residente no terminal de acesso do telespectador que tem a capacidade de enviar informações para um servidor externo. Como resultado do trabalho realizado, criou-se um módulo de sistema denominado JiTVPSI-CommService. Esse módulo de comunicação, que está integrado à plataforma JiTV, é um sistema que visa explorar os padrões envolvidos na TV Digital Brasileira, implementando o envio de dados de aplicações pelo canal de retorno. Com o sistema JiTVPSI-CommService, é possível enviar mensagens para servidores externos por meio de comunicação por *e-mail*, *socket*, submissão de formulário HTTP, *Web-Service* e FTP. O ambiente de testes envolveu a plataforma JiTV e suas aplicações JiTVDengue e JiTVEleicao. Com os testes realizados, notou-se uma significativa diminuição na complexidade do código das aplicações no que diz respeito à comunicação com o Provedor de Serviços Interativos através do canal de retorno. As tarefas de comunicação e controle de acesso ao canal de retorno foram retiradas das aplicações e completamente transferidas para a aplicação JiTVPSI-CommService.

Palavras-chave: TV Digital, Interatividade, *Middleware*, Comunicação, Canal de Retorno

ABSTRACT

With the development of technologies that involve the environment of generation, transmission and presentation of TV programs, coupled with the creation and implementation of systems to Brazilian Interactive Digital TV, demands for products and services related to this new platform are emerging. Applications for Interactive Digital TV present a new paradigm of development. This new paradigm is related to development tools and features such as usability and communication. A study of the patterns involved in the Brazilian Digital TV system was developed with focus on creating a layer of software related to the communication of applications with the return channel to provide interaction services in a generic middleware. This work presents strategies for the development of a resident system within the terminal with the ability to send information to an external server. As a result of this work, a system module called JiTVPSI-CommService was created. This communication module, integrated with JiTV platform, is a system designed to explore the patterns involved in the Brazilian Digital TV System, implementing the data delivery by return channel. With the system JiTVPSI-CommService, it is possible to send messages to external servers through communication by email, socket, HTTP form submission, Web-Service and FTP. The test environment involved JiTV platform and its applications JiTVDengue and JiTVEleicao. With the tests, it was found a significant reduction in the complexity of application code regarding the communication with the Provider of Interactive Services by the return channel. The tasks of communication and control access to the return channel were withdrawn and completely transferred to the application JiTVPSI-CommService.

Key-words: Digital TV, Interactivity, Middleware, Comunication, Return Channel

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

64-QAM *Quadrature Amplitude Modulation*

8-VSB *Vestigial Sideband Modulation*

AAC *Advanced Audio Coding*

ACAP *Advanced Common Application Platform*

ADSL *Assymmetric Digital Subscriber Line*

AIT *Application Information Table*

API *Application Programming Interface*

ARIB *Associations of Radio Industries and Business*

ARIB STB-B23 *Application Execution Engine Platform for Digital Broadcasting*

ARIB STB-B24 *Data Coding and Transmission Specification for Digital Broadcasting*

ATSC *Advanced Television Systems Committee*

AWT *Abstract Window Toolkit*

BML *Broadcast Markup Language*

BP *Baseline Profile*

BST-OFDM *Band Segmented Transmission Orthogonal Frequency Division Multiplexing*

CDC *Connected Device Configuration*

CDMA *Code Division Multiple Access*

CLDC *Connected Limited Device Configuration*

COFDM *Coded Orthogonal Frequency Division Multiplex*

CSS *Cascade Style Sheet*

DAE *Declarative Application Environment*

DASE DTV *Application Software Environment*

DAVIC *Digital Audio Visual Council*

Dolby AC *Digital Audio Compression*

DOM *Document Object Model*

DSM-CC *Digital Storage Media - Command and Control*

DVB *Digital Video Broadcasting*

DVBRCT *DVB - Return Channel through Terrestrial*

DVD *Digital Video Disc*

FP *Foundation Profile*

FTP *File Transfer Protocol*

GEM *Globally Executable* MHP

HAVi *Home Audio Video Interoperability*

HAVi-UI *Home Audio Video Interoperability - User Interface*

HDTV *High Definition Television*

HE-AAC *High Efficiency Advanced Audio Coding*

HiP *High Profile*

HSQLDB *Hyper SQL Database*

HTML *HyperText Markup Language*

HTTP *Hypertext Transfer Protocol*

IEEE *Institute of Electrical and Electronics Engineering*

ISDB *Integrated Services Digital Broadcasting*

ISDTV *International System for Digital Television Terrestrial*

JAS *JiTV Application Suite*

JADK *JiTV Application Development Kit*

Java EE Java *Enterprise Edition*

Java ME Java *Micro Edition*

Java SE Java *Standard Edition*

JiTV *Java Interactive Television*

JMF *Java Media Framework*

JPEG *Joint Photographic Experts Group*

JRE *Java Runtime Environment*

Lavid Laboratório de Aplicações em Vídeo Digital

LWUIT *LightWeight User Interface Toolkit*

Mbps Megabits por segundo

MHEG *Multimedia and Hypermedia Experts Group*

MHP *Multimedia Home Platform*

MIDP *Mobile Information Device Profile*

MPEG *Moving Picture Experts Group*

MPEG2-AAC *Moving Picture Experts Group 2 - Advanced Audio Coding*

MPEG2-BC *Moving Picture Experts Group 2 - Backwards-Compatible*

NCL *Nested Declarative Language*

NTSC *National Television System Committee*

OCAP *OpenCable Application Platform*

OCF *OpenCard Framework*

OSGI *Open Services Gateway initiative*

PAE *Procedural Application Environment*

PBP *Personal Basis Profile*

PDA *Personal Digital Assistants*

PJAE *Personal Java Application Environment*

PLC *Power Line Communication*

PNG *Portable Network Graphics*

PP *Personal Profile*

PS *Parametric Stereo*

PSI Provedor de Serviços Interativos

PSK *Phase-Shift Keying*

PVR *Personal Video Recorder*

QPSK *Quadrature Phase-Shift Keying*

SBR *Spectral Band Replication*

SBTVD Sistema Brasileiro de TV Digital

SDTV *Standard Definition Television*

SMIL *Synchronized Multimedia Integration Language*

SMTP *Simple Mail Transfer Protocol*

SOAP *Simple Object Access Protocol*

SQL *Structured Query Language*

TVDI TV Digital Interativa Brasileira

UHF *Ultra High Frequency*

VHF *Very High Frequency*

WiMAX *World Interoperability for Microwave Access*

XHTML *Extensible Hypertext Markup Language*

XML *Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

A evolução das tecnologias para tratamento de áudio e vídeo juntamente com o aperfeiçoamento de técnicas de comunicação de dados em sistemas de TV Digital proporcionam melhoria na qualidade de transmissão e a possibilidade de maior interação da programação com o telespectador. Além da alta definição de imagens e som com envolvimento ambiente, diversas categorias de serviços, como guias eletrônicos de programação, aplicações de enquete ao usuário, pesquisas governamentais e votação eletrônica se adicionam à programação de TV tradicional.

Com o desenvolvimento da TV Digital em todo o mundo e a perspectiva de que essa nova tecnologia se torne cada dia mais presente no dia-a-dia das pessoas, percebe-se a oportunidade de unir o alcance populacional que a TV possui com essa nova maneira de comunicação e relacionamento bidirecionais com o usuário, já que cerca de 95,1% das residências no Brasil possuem pelo menos um aparelho televisor, enquanto que somente 31,2% dos domicílios possuem um computador (Instituto Brasileiro de Geografia e Estatística, 2008).

Ao redor do mundo, surgiram diferentes sistemas de TV Digital, cada um com seu objetivo específico. O sistema japonês foi lançado no mercado nos anos 90 e aborda características direcionadas para a exibição em múltiplos dispositivos que não somente o televisor. Nos Estados Unidos, o sistema foi lançado em 1996, com objetivo de transmitir imagens em alta definição. E o sistema europeu, que atualmente é o sistema de TV Digital mais usado no mundo, foi lançado com foco em interatividade com o telespectador. O Brasil optou por criar e adotar um sistema próprio, diversificando as características dos outros três sistemas. O sistema brasileiro de TV Digital é chamado Sistema Internacional para Difusão Digital Terrestre (*Terrestrial International System for Digital Broadcasting* - ISDB-T), também conhecido como SBTVD.

Com a diversidade de características e fabricantes de receptores, para o desenvolvimento de aplicações é imprescindível a existência de uma camada chamada *middleware*

que é responsável por abstrair a complexidade dos receptores das aplicações sendo executadas. O *middleware* disponibiliza uma forma única de comunicação entre as aplicações e o sistema operacional do receptor, de maneira que as aplicações possam usar facilmente os recursos dos dispositivos. Cada sistema de TV Digital possui seu próprio *middleware*. O *middleware* do sistema brasileiro de TV Digital encontra-se em desenvolvimento e é denominado Ginga.

1.1 Motivação

Com o surgimento de novas tecnologias que envolvem o ambiente de geração, transmissão e apresentação da programação e execução de sistemas para TV Digital Interativa Brasileira (TVDI), surgiu também a dificuldade de entendimento desses novos aparatos e de toda a tecnologia que os engloba (VRBA; CVRK; SYKORA, 2006). São novos padrões a serem desvendados pelos desenvolvedores de aplicações. As ferramentas que atualmente são usadas para desenvolvimento de aplicações em ambiente cliente/servidor e Internet não se adequam à TV e provavelmente terão que ser revistas. Algumas ferramentas já apresentam facilidades para o desenvolvimento de aplicações para diversos dispositivos, como celulares, *hand-helds* e TV, mas ainda nada comparado às funcionalidades de ferramentas para desenvolvimento de aplicações tradicionais. A principal motivação da pesquisa realizada foi estudar os padrões envolvidos na TV Digital Brasileira, com a proposta de desenvolver uma camada de software relacionada à comunicação de aplicações com o canal de retorno, para prover serviços de interação.

De uma maneira geral, o desenvolvimento de tecnologias para suporte à implementação de aplicações para TVDI terá cada vez mais importância no cenário das aplicações das Tecnologia da Informação e da Comunicação. A Televisão Digital Interativa apresenta um novo paradigma no desenvolvimento de sistemas e traz desafios tecnológicos para empresas de software e de telecomunicações. Os desafios estão relacionados às novas tecnologias envolvidas no ambiente de TV Digital, como a difusão de dados, e às restrições dos aparelhos receptores e dos canais de comunicação com o meio externo. Nesse contexto, o conhecimento sobre técnicas de construção desse tipo de aplicação trará contribuições para o mercado, a indústria de software e o meio acadêmico.

Com um ambiente ainda em evolução, os diversos trabalhos publicados na área de TV Digital mostram que esse é um ramo de pesquisa inicial e ainda muito promissor. As abordagens relacionadas a trazer o ambiente de Internet para a TV se mostraram sem muito sucesso devido às inúmeras diferenças entre os dois ambientes. Assim, os trabalhos

futuros tendem a aperfeiçoar as aplicações quanto às especificidades do ambiente de TV Digital e também melhorar a abstração de componentes do ambiente para facilitar a implementação de sistemas.

1.2 Objetivos

Este trabalho tem como objetivo principal propor um Sistema de Comunicação no ambiente do Terminal de Acesso do Telespectador, em *Middleware* Genérico, para facilitar e garantir a transmissão de dados para o Provedor de Serviços Interativos (PSI) através do canal de retorno, armazenando dados ainda não enviados no terminal de acesso e permitindo possível envio de informações a servidores remotos ao terminal, inclusive ao sistema transmissor de TV.

São objetivos específicos:

- Propor um estratégia de comunicação entre as aplicações de terceiros e a aplicação desenvolvida, chamada JiTVPSI-CommService, de forma que possa haver a coleta de dados, e esses dados possam ser enviados ao PSI;
- Apresentar formas de comunicação entre o Terminal de Acesso do Telespectador e o PSI através do canal de retorno e implementar protocolos de transmissão de dados entre eles. Os protocolos implementados são: SMTP (*e-mail*), Socket, HTTP, HTTP por WebService e FTP;
- Conceber um mecanismo de armazenamento das informações em Banco de Dados Relacional no Terminal de Acesso do Telespectador quando os dados não puderem ser enviados para o meio externo;
- Adaptar aplicações de protótipo para Televisão Digital Interativa em domínios de aplicação diversos, tais como Educação a Distância (*t-learning*), Comércio Eletrônico (*t-commerce*) ou Governo Eletrônico (*t-gov*) para validar a solução proposta.

1.3 Estrutura da Dissertação

O restante deste documento se apresenta organizado da seguinte forma. O Capítulo 2 descreve um histórico sobre os principais sistemas de TV digital existentes atualmente, realizando ainda uma análise comparativa. No Capítulo 3 são apresentadas as especificações de *software* para os diversos *middlewares* e as tecnologias com maior importância

no âmbito das aplicações de TV Digital e interação com o usuário. O Capítulo 4, por sua vez, discorre sobre a arquitetura do projeto proposto, detalhando seus módulos e suas funcionalidades e a forma de implementação de cada uma de suas partes, apresentando pontos positivos e negativos das abordagens adotadas. O Capítulo 5 apresenta os resultados alcançados nos testes práticos do projeto descrito nesta dissertação. Finalmente, o Capítulo 6 encerra a dissertação, mostrando as conclusões obtidas a partir do trabalho realizado e as possibilidades de trabalhos futuros.

2 TELEVISÃO DIGITAL E TELEVISÃO DIGITAL INTERATIVA

Um sistema de TV Digital consiste em um conjunto de definições que se integram para possibilitar a construção dos dispositivos para transmissão e recepção do sinal de TV. As primeiras discussões relacionadas à implantação da TV Digital no país estavam direcionadas a qual sistema de TV utilizar. A questão principal era se o governo brasileiro adotaria um padrão já desenvolvido em outros países ou se partiria para o desenvolvimento do seu próprio padrão. Assim, três plataformas se mostraram concorrentes nesse processo (BOLAñO; VIEIRA, 2004):

- *Integrated Services Digital Broadcasting* (ISDB);
- *Advanced Television Systems Committee* (ATSC);
- *Digital Video Broadcasting* (DVB).

O sistema japonês ISDB vem sendo trabalhado desde os anos 70 e foi lançado no mercado nos anos 90. Nos Estados Unidos, ATSC foi lançado somente em 1996, sendo resultado de estudos feitos naquele país desde 1987. E, por último, temos o sistema europeu DVB, que atualmente é o sistema de TV Digital mais usado no mundo (MHP, 2009).

Todos os três sistemas apresentaram características relevantes para o cenário brasileiro. No sistema japonês, a funcionalidade mais importante foi a mobilidade (MARGALHO; FRANCêS; COSTA, 2007). No sistema americano ATSC, se destaca o objetivo de transmissão de imagens em alta definição, com a desvantagem de não possibilitar a transmissão em dispositivos móveis (BOLAñO; VIEIRA, 2004). No sistema europeu DVB deram ênfase à interatividade com o telespectador. Diante dessa diversidade, a opção do governo brasileiro foi desenvolver um novo sistema para TV Digital, com o intuito de aproveitar as vantagens encontradas nas três plataformas apresentadas. Com as várias pesquisas desenvolvidas no país, e o projeto de TV Digital sendo conduzido com o apoio de centros

de pesquisas e universidades, foi criado e adotado pelo Brasil o *International System for Digital Television Terrestrial* (ISDTV-T) (ISDTV-T Forum, 2006).

Os sistemas para TV Digital podem ser representados por meio de uma arquitetura em camadas. Cada camada oferece serviços para a camada superior e utiliza serviços da camada inferior. A representação da arquitetura é a mesma para todos os sistemas. As diferenças se mostram presentes nos diversos pontos da arquitetura relacionados ao processo de modulação, codificação, compressão e transmissão e execução de aplicações e *middleware* adotado. Nas seções que se seguem, serão apresentados os três principais sistemas internacionais de Televisão Digital, e alguns conceitos importantes dessa plataforma.

2.1 Sistemas Internacionais de Televisão Digital

2.1.1 DVB

O padrão europeu para TV Digital é chamado *Digital Video Broadcasting* (DVB). É utilizado na maioria dos países da Europa e em outros países como Austrália, Malásia, Hong Kong, Índia, África do Sul e em muitos outros. O DVB é o sistema de TV Digital mais usado atualmente no planeta. A Inglaterra é o país em que o DVB está mais consolidado. Em abril de 2009, aproximadamente 10 milhões de receptores MHP estavam em funcionamento no mundo (MHP, 2009).

O DVB está dividido em alguns padrões de transmissão: DVB-T (transmissão terrestre por radiodifusão), DVB-C (transmissão via cabo), DVB-S (transmissão por satélite), DVB-MC (transmissão por micro ondas com frequência de até 10GHz) e DVB-MS (transmissão por micro ondas com frequência acima de 10GHz) (FERNANDES; LEMOS; SILVEIRA, 2004). Esses padrões de transmissão adotam diferentes esquemas de modulação para obter maior desempenho de acordo com características de cada um deles. A modulação é o processo pelo qual alguma característica de uma onda portadora é alterada para se adaptar ao sinal no formato a ser transmitido. O principal foco do sistema DVB é a transmissão terrestre com o uso da modulação *Coded Orthogonal Frequency Division Multiplex* (COFDM), que possui alto desempenho para recepção de antenas internas. O sistema terrestre (DVB-T) possui taxa de transmissão que varia de 5 a 31,7 Mbps operando em canais de 6, 7 e 8 MHz. Para transmissões com DVB-C, utiliza-se a modulação 64 *Quadrature amplitude modulation* (64-QAM), e para transmissões com satélite, o DVB-S recomenda a modulação *Quadrature Phase-Shift Keying* (QPSK). Nas transmissões por

micro ondas, são previstos dois tipos de modulação: para frequências abaixo de 10GHz, o sistema recomenda o uso de 64-QAM e para frequências acima de 10GHz, o uso da modulação QPSK. Na Figura 1, adaptada de Fernandes, Lemos e Silveira (2004), pode-se analisar os padrões utilizados no sistema DVB em comparação com os outros padrões em cada camada que envolve a TV Digital.

O padrão de codificação utilizado pelo DVB é o *Moving Picture Experts Group*-2 (MPEG-2). O sinal de áudio é codificado com a recomendação de MPEG2-BC e o sinal de vídeo é codificado com a recomendação MPEG2-Video com qualidade *Standard Definition Television* (SDTV). O formato SDTV mantém a imagem em 4:3, que é similar à transmissão analógica, mas com resolução de DVD com 720 linhas. O padrão MPEG-2 de codificação de áudio e vídeo permite uma otimização da banda de frequência. O modelo analógico, pelo seu padrão de codificação, permitia a transmissão de apenas um canal de TV por banda de frequência e o MPEG-2, em SDTV, com qualidade digital, permite a transmissão de várias programações e mais o canal de dados em cada banda. Na camada de transporte (responsável pela junção dos fluxos de áudio, vídeo e dados em um fluxo único), o MPEG2-Sistemas é utilizado para multiplexação e demultiplexação de conteúdo de vídeo, áudio e dados. O *middleware* utilizado é o MHP, cuja especificação é denominada DVB-MHP. O MHP suporta execução de aplicações procedimentais em Java TV (Sun Microsystems, 2009c) que possui a nomenclatura de DVB-J e, opcionalmente, a execução de programas declarativos em linguagem HTML com DVB-HTML. O MHP será detalhado na Seção 3.1.1.

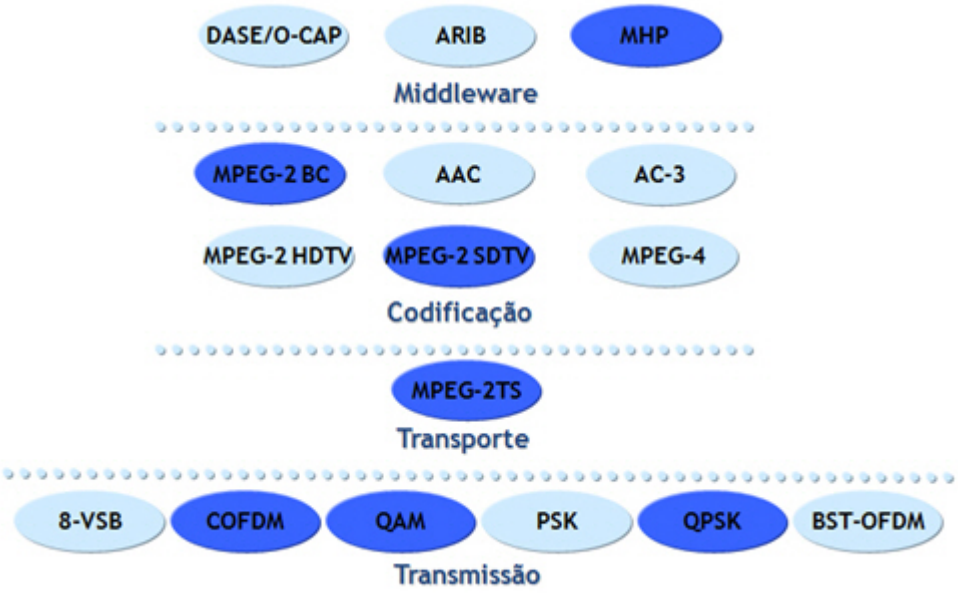


Figura 1: Arquitetura DVB

2.1.2 ATSC

O padrão para TV Digital ATSC, conhecido também como padrão americano de televisão digital, descreve um sistema desenhado para transmitir áudio e vídeo de alta qualidade e dados dentro de um canal simples de 6, 7 ou 8 MHz em sistema terrestre. Esse padrão está em funcionamento nos Estados Unidos desde 1998 e já foi adotado por Canadá, Coreia do Sul, Taiwan, Argentina e México (FERNANDES; LEMOS; SILVEIRA, 2004).

O ATSC, para radiodifusão, utiliza a modulação 8-*Vestigial sideband modulation* (8-VSB) com taxa de transmissão de 19,8 Mbps. No padrão de modulação 8-VSB, a transmissão de TV para dispositivos móveis não foi levada em consideração. Isso ocorre pelo fato de que esse padrão foi desenvolvido antes da telefonia móvel ter obtido tamanha relevância. Focaram então na elaboração de padrões que possibilitassem a transmissão de imagens de alta qualidade ao telespectador, definindo assim o padrão de modulação 8-VSB como inadequado para funcionamento em aparelhos portáteis. Esse fato levou o Brasil a rejeitar o sistema ATSC como seu sistema padrão para TV Digital, em testes realizados no país. Além disso, apresenta limitações quanto à recepção em antenas internas, principalmente em grandes cidades. No ATSC, para transmissão via satélite, foi adotada a modulação QPSK, e para transmissão por cabo, a modulação 64-QAM. A Figura 2, adaptada de Fernandes, Lemos e Silveira (2004), ilustra esquematicamente a arquitetura ATSC.

O ATSC padronizou o padrão de codificação MPEG-2, com perfil principal (*Main Profile*) em nível alto (*High Level*) para codificação de vídeo e o padrão ATSC *Digital Audio Compression* (*Dolby* AC-3) para codificação de áudio. A grande ênfase em qualidade resultou no advento do *High Definition Television* (HDTV) e do som *surround* multicanal (RICHER et al., 2006). O vídeo pode ser transmitido no formato 16:9 e com uma definição de até 1920x1080 *pixels*, resultando em uma qualidade 6 vezes superior à do padrão analógico anteriormente utilizado, o *National Television System Committee* (NTSC). Na camada de transporte, o padrão multiplexa os fluxos elementares de áudio, vídeo e dados usando as recomendações MPEG-2 Sistemas. Para a camada de software, foi criado o DTV *Application Software Environment* (DASE) para execução de serviços de interação no sistema terrestre, permitindo a execução de aplicações baseadas em linguagem procedimental ou declarativa. Como linguagem procedimental, o DASE adotou a execução de aplicações Java TV (Sun Microsystems, 2009c) e, como linguagem declarativa, suporta uma versão estendida da linguagem HTML (FERNANDES; LEMOS; SILVEIRA,

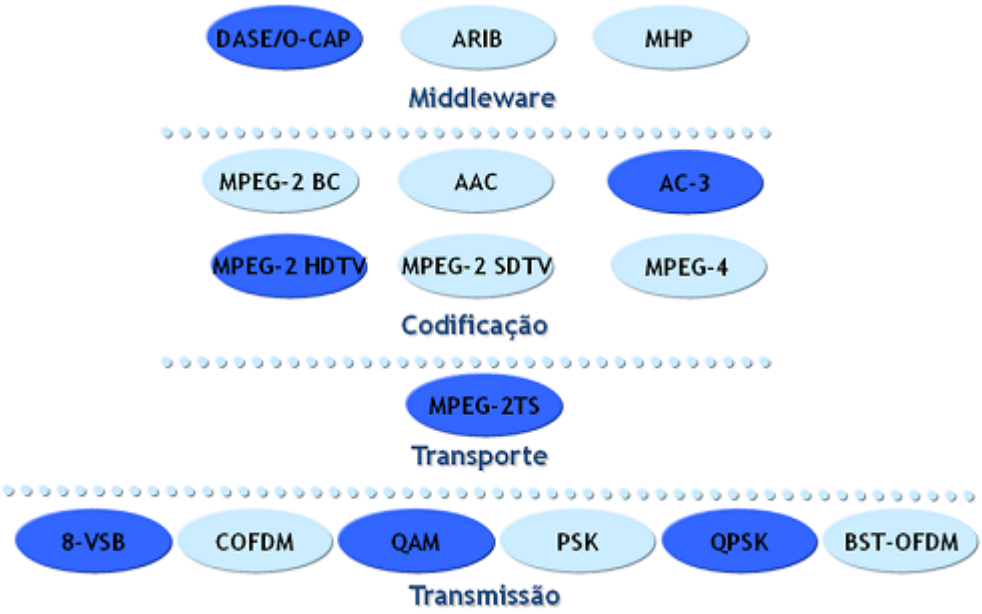


Figura 2: Arquitetura ATSC

2004). O *middleware* DASE evolui para os padrões *OpenCable Application Plataform* (OCAP) no ambiente de TV a cabo e, posteriormente, *Advanced Common Application Plataform* (ACAP) para um ambiente comum. Mais detalhes sobre os *middlewares* serão apresentados na Seção 3.1.

2.1.3 ISDB

O padrão japonês para TV Digital é o *Integrated Services Digital Broadcasting* (ISDB). Atualmente é adotado somente no Japão, apesar de já ter sido testado em outros países da Ásia (Cingapura e Hong Kong) e também no Brasil (BOLAñO; VIEIRA, 2004). É conhecido como o sistema de TV Digital que mais possui facilidades: alta definição (HDTV) e possibilidade de transmissão em dispositivos móveis. O ISDB é baseado no sistema europeu de transmissão DVB, apresentando algumas melhorias quando à imunidade a interferências. Essa imunidade estabelece um bom desempenho de transmissões de alta definição com a recepção móvel (FERNANDES; LEMOS; SILVEIRA, 2004).

Como encontrado em outros sistemas, o ISDB também permite a configuração de diversos esquemas de modulação como transmissão terrestre, via cabo e via satélite. Na transmissão terrestre, a radiodifusão pode operar com largura de banda de 6, 7 ou 8 MHz da mesma forma que o padrão DVB, mas, com algumas pequenas variações, permite uma taxa de transmissão variando entre 3,65 e 23,23 Mbps com modulação COFDM (FERNANDES; LEMOS; SILVEIRA, 2004). Uma melhoria dessa técnica de modulação está

sendo utilizada, chamada *Band Segmented Transmission Orthogonal Frequency Division Multiplexing* (BST-OFDM) (SOARES; BARBOSA, 2008). As modulações por cabo e por satélite no padrão ISDB adotam os padrões 64-QAM e 8-PSK, respectivamente.

O sinal de áudio é codificado pela recomendação de MPEG2 *Advanced Audio Coding* (MPEG2 AAC) e o sinal de vídeo é feito utilizando MPEG2 Vídeo com qualidade HDTV. O sistema japonês é flexível de tal forma que possibilita o uso do MPEG2 Vídeo em diferentes níveis de resolução. Como acontece no DVB e no ATSC, na camada de transporte, a multiplexação do fluxo elementar de áudio, vídeo e dados é suprida pela especificação MPEG2 Sistemas. Na camada de software, o sistema japonês utiliza o padrão de *middleware Associations of Radio Industries and Business* (ARIB), que permite a execução de sistemas em linguagem declarativa *Broadcast Markup Language* (BML). O ARIB, na sua versão mais atual B24, complementa a especificação B23 que é baseada no padrão europeu DVB-MHP, indicando uma tendência em estabelecer uma conformidade com outros padrões de *middleware*. A Figura 3, adaptada de Fernandes, Lemos e Silveira (2004), mostra a arquitetura ISDB.

A inovação do padrão ISDB está na possibilidade de segmentação de banda, em que é possível que um mesmo canal transmita até 13 segmentos diferentes. Essa funcionalidade demonstra claramente o objetivo para o qual esse sistema foi construído: oferecer diversos serviços de comunicação por meio de convergência total com Internet, telefones celulares 3G e outros, em uma mesma plataforma tecnológica (BOLAñO; VIEIRA, 2004). O alcance efetivo de todas essas funcionalidades elevam substancialmente o custo de implantação do sistema japonês, sendo esse, provavelmente, o motivo de estar em funcionamento atualmente somente no seu país de origem.

2.1.4 ISDTV

O Brasil adotou o *International System for Digital Television Terrestrial* (ISDTV-T) (ISDTV-T Forum, 2006) depois de estudos e testes realizados com os outros sistemas internacionais de TV Digital - ATSC, DVB-T e ISDB (BEDICKS et al., 2006). O principal objetivo do projeto brasileiro era definir um modelo de referência para a transmissão de TV Digital no país. Para isso, levou-se em consideração as características do país, principalmente, o seu território muito montanhoso e sua população muito pobre e carente de informação.

O padrão japonês ISDB foi escolhido como tecnologia de base para o sistema brasileiro. Foi criado inclusive um acordo entre os dois países para cooperação tecnológica. Decidiu-se

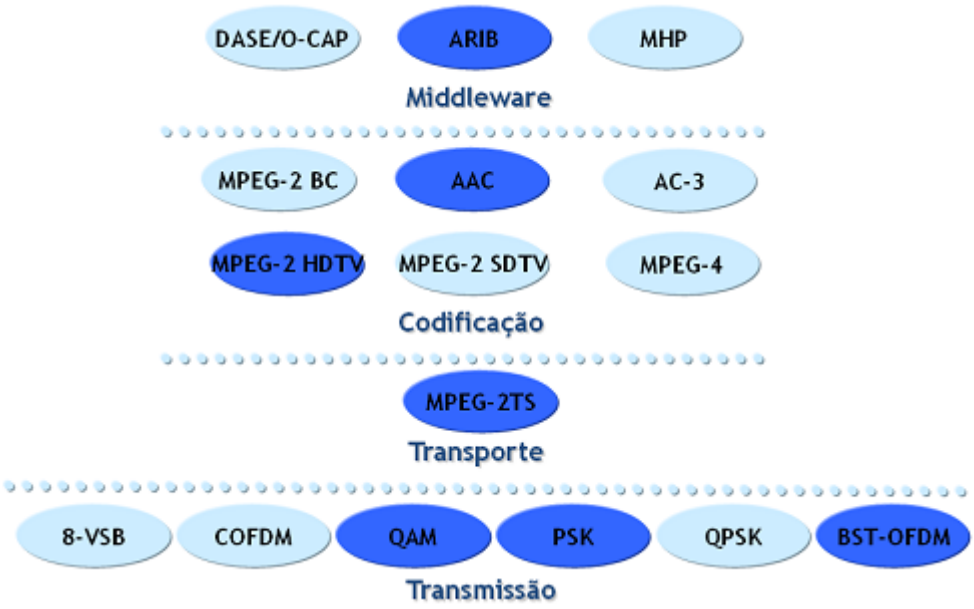


Figura 3: Arquitetura ISDB

então por uma solução híbrida, mesclando características do sistemas japonês e inovações propostas para o sistema brasileiro, principalmente, na camada de software, para permitir maior interatividade (SANTOS JUNIOR et al., 2009). A transmissão do sistema brasileiro iniciou-se em São Paulo em 2 de dezembro de 2007.

O ISDTV-T recomenda a mesma tecnologia usada no ISDB-T para codificação e modulação de sinais da televisão digital, pelo padrão BST-OFDM com largura de banda de 6 MHz. Esse padrão permite que sinais sejam transmitidos com a técnica de banda segmentada de transmissão (BST) e multiplexação de divisão de frequência ortogonal (OFDM), possibilitando flexibilidade e mobilidade na transmissão para receptores fixos e móveis. Nos testes feitos no Brasil, esse padrão de modulação obteve melhores resultados do que em outros sistemas testados (ATSC e DVB) (BEDICKS et al., 2006).

A codificação de áudio do sistema brasileiro é feita pelo padrão MPEG-2 AAC, com transmissões estéreo e multicanais 5.1 simultaneamente. Esse padrão é considerado o estado da arte em termos de codificação de áudio de alta qualidade. Como pode ser visto em (SOARES; BARBOSA, 2008), existem técnicas de melhoramento do áudio para o padrão AAC, tais como SBR (*Spectral Band Replication*) e PS (*Parametric Stereo*). SBR permite a mesma qualidade de som com a metade da taxa de *bits* e PS aumenta a eficiência de codificação. A combinação do AAC com SBR é chamada de HE-AAC (*High Efficiency - AAC*) e a combinação do AAC com SBR e PS é chamada de HE-AAC versão 2. As variações da codificação de áudio são utilizadas de acordo com o perfil de uso do telespectador.

Na codificação de vídeo, o sistema brasileiro apresenta uma de suas inovações, utilizando o padrão H.264, também conhecido como MPEG-4 Parte 10 ou *Advanced Video Coding*. O padrão H.264 representa uma das maiores vantagens em compressão de vídeo pois fornece uma redução no número de *bits* em relação aos outros padrões. Para altos níveis de qualidade de vídeo, o padrão H.264 diminui em até a metade a taxa de *bits* do MPEG-2. Esse padrão se divide em perfis e níveis, e para o sistema brasileiro de TV Digital serão usados o perfil alto (HiP) para receptores fixos e móveis, e o perfil base (BP) para receptores móveis.

A forma com que os fluxos de áudio, vídeo e dados devem ser multiplexados está padronizada nos sistemas europeu, americano, japonês e brasileiro, com a utilização do MPEG-2 Sistemas. Outra parte significativa da inovação tecnológica brasileira está na camada de software. O *middleware* adotado pelo ISDTV é chamado Ginga que é o resultado das pesquisas brasileiras em relação a plataformas de desenvolvimento procedimental e declarativo, originando o Ginga-J (SOUZA; LEITE; BATISTA, 2007) e Ginga-NCL (SOARES; RODRIGUES; MORENO, 2007). O *middleware* Ginga será descrito em detalhes na Seção 3.1.7. Os padrões do sistema ISDTV podem ser visualizados no esquema apresentado na Figura 4.

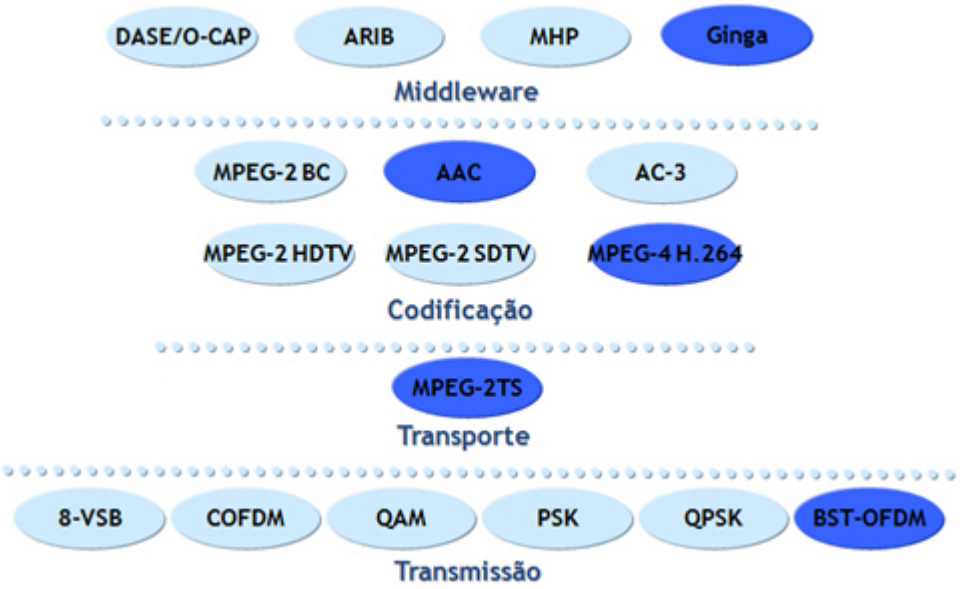
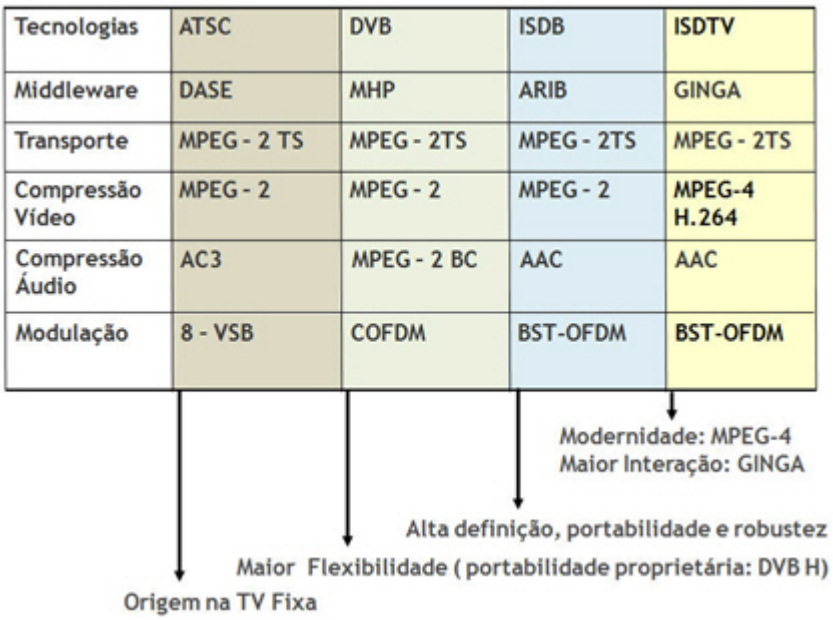


Figura 4: Arquitetura ISDTV

A Tabela 1 resume, por meio da arquitetura em camadas, a estratégia de padronização de cada um dos sistemas apresentados. Todos os sistemas são divididos nas mesmas camadas e cada um deles optou por adotar padrões específicos de acordo com seus objetivos particulares.

Tabela 1: Comparação entre os sistemas de TV Digital



2.2 Terminal de Acesso

Em um sistema de TV Digital, o dispositivo de acesso, também chamado de receptor, terminal de acesso, *set-top-box*, *home-hub*, dentre outros (CARVALHO et al., 2007), é o componente responsável, basicamente, por decodificar o sinal de áudio e vídeo, permitindo a apresentação no televisor. Com a maior largura de banda de transmissão e a compactação dos dados transmitidos pela TV Digital, há espaço disponível para envio de informação agregada ao áudio e vídeo (SOARES; BARBOSA, 2008). O receptor, assim, passou a possuir poder de processamento computacional, com processador, memória e sistema operacional, com o objetivo de manipular e executar as aplicações envidadas na transmissão de TV Digital, possibilitando interatividade com o telespectador (SOARES; BARBOSA, 2008) (BECKER; MONTEZ, 2004).

Os terminais de acesso são compostos por componentes físicos e de software, e passaram a conter tecnologias semelhantes aos computadores, possuindo, inclusive, sistemas operacionais que ficam armazenados em memória não volátil (ROM). Os sistemas operacionais dos receptores são bem mais simples que os sistemas operacionais para computadores pessoais. É pelo sistema operacional e por um *middleware* que o receptor abstrai as características de *hardware* do equipamento e permite que qualquer aplicação possa ser executada no ambiente do terminal do telespectador.

Os componentes físicos são representados por circuitos eletrônicos que são organizados

em núcleos responsáveis pelo tratamento da rádio frequência e posterior tratamento do conteúdo digital (CARVALHO et al., 2007). O receptor pode estar embutido no dispositivo exibidor (televisor, celular ou PDA) ou ser um aparelho separado. Junto a ele, sempre deve haver um dispositivo para captura do sinal difundido. Uma antena específica deve ser usada para captar sinais de radiodifusão ou difusão por satélite, ou a captura do sinal pode ser via cabo. Depois de demodulado, o sinal é de-multiplexado e repassado para os decodificadores de áudio e vídeo e à *Central Processing Unit* (CPU) para processamento dos dados (SOARES; BARBOSA, 2008). Os terminais ainda possuem memória, podendo ser volátil (RAM) ou permanente (ROM), para armazenar e manipular os dados de programas, e interfaces físicas para integração com diversos dispositivos externos como *modem*, interface USB, 10BASE-T, PCMCIA II e leitores de *SmartCards*. A Figura 5 representa graficamente um terminal de acesso.

De acordo com o ponto de vista de serviço e execução de aplicações, definem-se três perfis de configuração para os terminais de acesso. Os perfis abrangem funcionalmente a flexibilidade tecnológica permitida nos diversos sistemas de TV Digital, e podem existir simultaneamente no mercado (CARVALHO et al., 2007). São eles :

- *Perfil de Radiodifusão (Baixo Custo)* - Essa categoria considera um dispositivo de

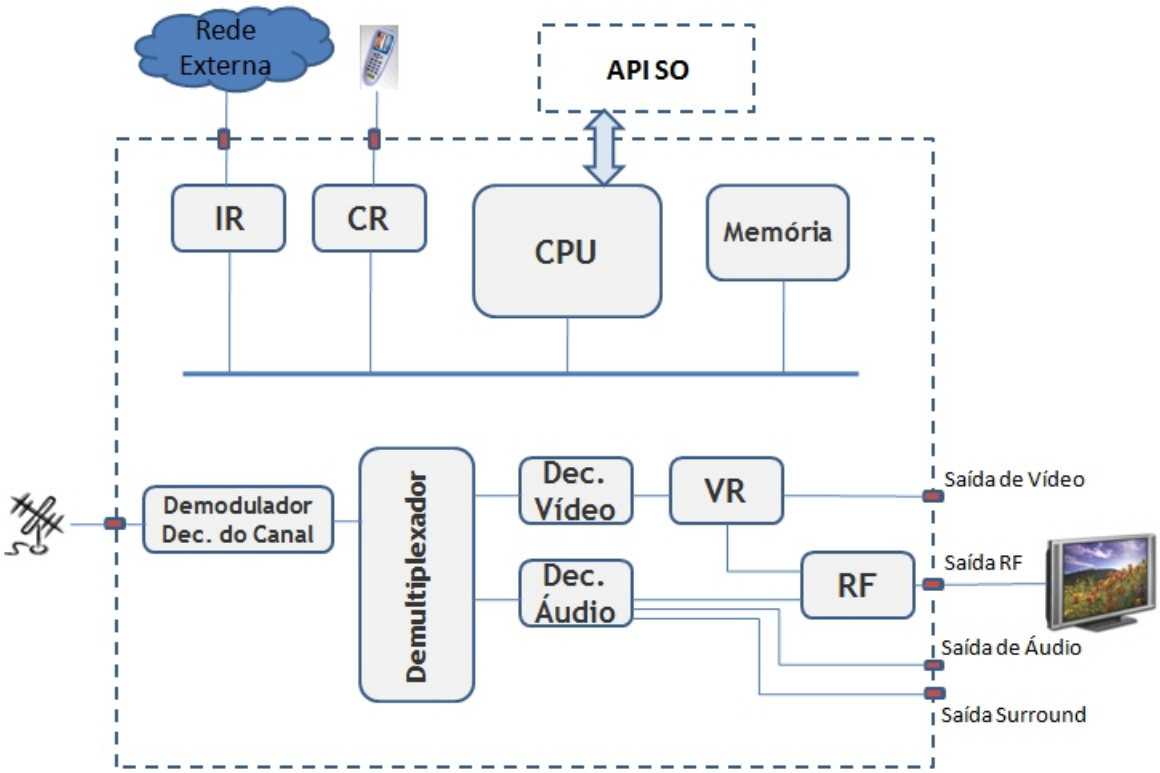


Figura 5: Terminal de Acesso de TV Digital

custo mais baixo e com tecnologia mais simples, oferecendo serviços essenciais de áudio e vídeo. É possível que se faça *download* de aplicações que serão executadas com interatividade local, sem o uso de um canal de retorno;

- *Perfil de Radiodifusão Interativa (Inclusão Digital)* - Essa categoria possibilita o uso de um conjunto de serviços interativos associados ao serviço de difusão. É requerido um canal de retorno;

- *Perfil de Acesso Internet (Maior valor agregado)* - Nessa categoria é possível o uso de serviços de Internet. O terminal de acesso pode conter funcionalidade avançadas como *Personal Video Recorder* (PVR) e suporte a jogos em tempo real.

Em um outra perspectiva, o perfil de baixo custo é dividido em outros dois perfis intermediários. O primeiro é um perfil em que o receptor somente é capaz de renderizar conteúdo digital de áudio e vídeo, e no segundo perfil, o receptor é capaz de executar aplicações e possuir algum serviço básico de interatividade (CESAR; VUORIMAA; VIERINEN, 2006). Esses mesmos autores também subdividem o perfil de maior valor agregado em outros dois perfis, um perfil que contempla o uso de conectividade pela Internet, com canal de retorno mais rápido, e um outro perfil para receptores com poder mais avançado de processamento multimídia com suporte a gráficos em três dimensões para jogos.

2.3 Carrossel de Dados e a Representação da Informação

O padrão MPEG-2 é responsável por transportar os dados, junto com o conteúdo de áudio e vídeo, da central difusora até o receptor do usuário. Quando o usuário muda de canal e escolhe alguma aplicação para ser executada, pode haver um problema, pois, caso o usuário comece a assistir a um programa após o seu início, a aplicação já deveria ter sido transmitida e o usuário ficaria sem poder executá-la, já que não teria sido carregada adequadamente no receptor. Para resolver esse problema, existe o carrossel, que mantém a mesma aplicação sendo enviada repetidamente por radiodifusão, fazendo com que o receptor possa capturar os vários pacotes de dados e organizá-los para construir uma aplicação completa quando necessário. Assim, um determinado canal pode ser sintonizado a qualquer momento, que não ocorrerá falta de sincronismos entre a programação de áudio e vídeo e as aplicações interativas.

Na TV Digital, prevaleceu a utilização do protocolo carrossel de dados e protocolo carrossel de objetos, especificados pelo padrão *Digital Storage Media - Command and*

Control (DSM-CC), para suportar a transmissão cíclica de dados (BECKER; MONTEZ, 2004), como ilustrado na Figura 6, sem o uso do canal de retorno. O DSM-CC é uma extensão do padrão MPEG-2 Sistemas que representa um conjunto de protocolos para controle e operação de fluxos gerenciados pelo MPEG2.

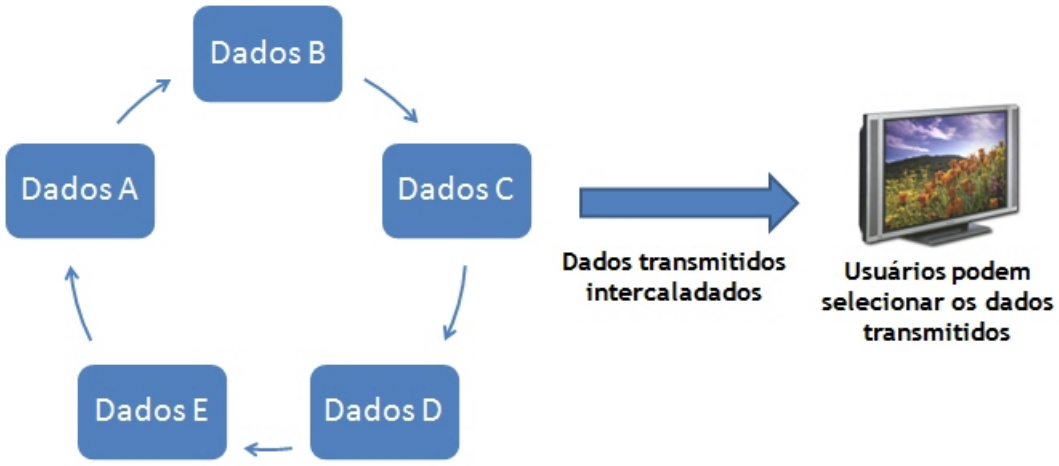


Figura 6: Carrossel

O carrossel de dados é um mecanismo de transporte do DSM-CC que permite a transmissão de grande volume de dados não estruturados para os terminais de acesso, repetindo o conteúdo a cada ciclo. Como os dados não estão estruturados, cabe ao sistema de TV Digital organizá-los.

O carrossel de objetos permite o envio de grupos estruturados de objetos. Quatro tipo de objetos podem ser encapsulados no carrossel de objetos: *ServiceGateway*, *Directory*, *File* e *Stream* (LIN; SUN, 2007). O objeto *ServiceGateway* representa o diretório raiz do sistema de arquivos e precisa ser recebido antes de qualquer outro objeto do sistema. Os objetos do tipo *Directory* são as informações estruturais do sistema de arquivos. Os objetos do tipo *File* são os dados dos arquivos. E os objetos *Stream* são apontamentos para outros fluxos elementares ou descritores de fluxos. O carrossel de objetos é mais completo que o carrossel de dados. Enquanto o carrossel de dados permite somente o envio de pedaços monolíticos de dados, o carrossel de objetos separa os dados em partes individualizadas que podem ser identificadas claramente pelo receptor e representar informações específicas como imagens, arquivos de dados e programas. Tanto o sistema brasileiro de TV quando o americano e europeu utilizam o carrossel de objetos (SOARES; BARBOSA, 2008).

Os dados a serem enviados pelo carrossel são divididos logicamente em módulos que,

por sua vez, são divididos em blocos de mesmo tamanho. A divisão em módulos facilita o processo de transmissão por radiodifusão (BATISTA et al., 2007). Cada módulo pode ser associado a uma priorização de retransmissão. Assim, os módulos com maior prioridade serão transmitidos com maior frequência (FERNANDES; LEMOS; SILVEIRA, 2004).

O sistema de arquivos de um carrossel se apresenta como um *drive* montado no *set-top-box*. Todos os arquivos estão visíveis e acessíveis, mas somente para leitura. De acordo com o ponto de vista do protocolo do carrossel de objetos, uma aplicação para TV Digital consiste de um conjunto de arquivos Java e outros arquivos em linguagem não procedimental. O *middleware* é responsável por executar esses arquivos, depois de serem montados de acordo com o protocolo do carrossel (LIN; SUN, 2007). Outro ponto relevante é que o DSM-CC não determina como as aplicações no receptor irão trabalhar com os dados do carrossel, deixando essa responsabilidade para o *middleware* (BECKER; MONTEZ, 2004).

As informações transmitidas por radiodifusão da emissora para o receptor do usuário podem ser de vários tipos de dados digitais. O serviço de envio de dados, conhecido como *datacasting*, pode ser classificado em: dados fortemente acoplados, dados fracamente acoplados e dados desacoplados (BECKER; MONTEZ, 2004). Quando os dados têm relacionamento temporal com a programação de áudio e vídeo, são chamados de fortemente acoplados. Os dados fracamente acoplados estão relacionados com a programação da TV mas podem ser executados a qualquer momento de acordo com a vontade do telespectador. E os dados que não possuem vínculo temporal com a programação são chamados de desacoplados. Em Soares e Barbosa (2008), uma abordagem ligeiramente diferente é apresentada. Os dados fracamente acoplados são chamados de síncronos e, como na abordagem de Becker e Montez (2004), não estão relacionados à temporalidade do fluxo de áudio e vídeo, mas estão, de alguma forma, sincronizados entre si, deixando de representar serviços que podem ser executados a qualquer momento. Os dados fortemente acoplados são chamados de sincronizados. E os dados desacoplados são chamados de assíncronos. Os dados assíncronos continuam representando as informações que não têm relação com a temporalidade dos fluxos de áudio e vídeo, e na abordagem de (SOARES; BARBOSA, 2008), incluem também as aplicações que podem ser executadas em tempo aleatório e determinado de acordo com a vontade do usuário.

Para que seja possível sincronizar a execução de determinada aplicação interativa com a programação da TV, o DSM-CC utiliza uma funcionalidade chamada de eventos de sincronismo DSM-CC (ou simplesmente eventos) (SOARES; BARBOSA, 2008). Uma

aplicação interativa tem que estar sincronizada com o programa corrente para assegurar que a exibição de um determinado conteúdo esteja coerente com o que está sendo exibido em áudio e vídeo. Esses eventos podem ser executados imediatamente ao recebimento da aplicação. Ou a aplicação pode ser executada em um tempo programado, de acordo com um agendamento prévio do evento.

2.4 Canal de Retorno e Interatividade

A interatividade e os novos conteúdos disponíveis são o requisito mais significativo para alavancar a TV Digital em todo o mundo. A interatividade irá definir a diferença entre as transmissões de TV analógica e digital. Programas da TV interativa vão envolver os telespectadores de forma muito mais interessante, melhorando o relacionamento das pessoas com a programação e das pessoas entre si. Comerciais de TV possibilitarão informações adicionais de produtos sob demanda.

Serviços interativos podem necessitar de diferentes níveis de comunicação entre o usuário e o provedor de serviços. Pode-se então classificar um sistema de acordo com a sua forma de interação com o telespectador. É possível existir uma interação local em que o canal de retorno no terminal de usuário não é requerido, chamado de sistema Pseudo-Interativo (SANTOS JUNIOR et al., 2009). Nessa forma de interação, o telespectador pode interagir com as aplicações mas não há retorno de informação para um servidor externo, como pode ser ilustrado pela Figura 7.

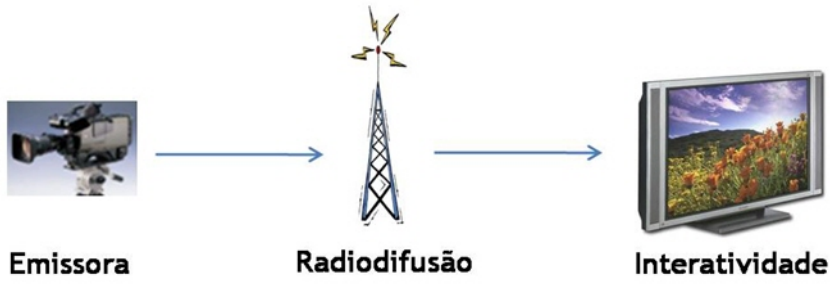


Figura 7: Sistema Pseudo-Interativo

Por outro lado, a necessidade do usuário telespectador em responder aos serviços na TV por meio de um canal de interação demanda a existência obrigatória de uma forma de comunicação com o meio externo, chamado de sistema Interativo Pleno (SANTOS JUNIOR et al., 2009). As informações geradas pelas aplicações podem ser transmitidas livremente para a emissora de televisão por um canal de comunicação (Figura 8). A resposta do usuário pode ser em forma de um simples comando de votação ou de compra de um

determinado produto que será entregue posteriormente na residência do telespectador ou a ação do usuário pode afetar diretamente a programação que está sendo assistida. Na outra ponta da comunicação, o Provedor de Serviços Interativos (PSI), vinculado à emissora de televisão, deve ser capaz de ouvir e reagir às respostas dos usuários. Segundo Soares e Barbosa (2008), o sistema interativo pleno ainda pode ser dividido em unidirecional e bidirecional. A aplicação unidirecional permite somente o envio de dados do telespectador para a emissora de televisão, enquanto a aplicação bidirecional permite tanto o envio de dados quando o *download* de dados utilizados pelos aplicativos.

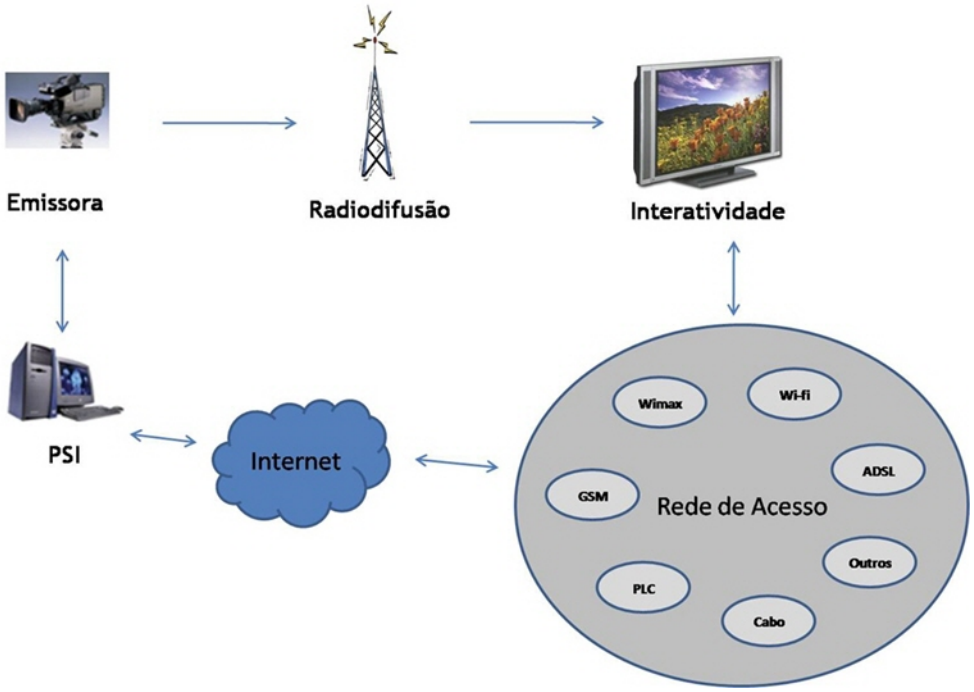


Figura 8: Sistema Interativo Pleno

O canal de retorno representa um dos instrumentos que possibilitam a interatividade na TV digital. É o meio de comunicação entre o terminal de acesso do telespectador e o provedor de serviços interativos. Sendo assim, por estar envolvido diretamente no processo comunicação para permitir a interação, o canal de retorno se torna uma das partes de maior relevância na arquitetura de um sistema de TV Digital (SANTOS JUNIOR et al., 2009).

A comunicação pode ocorrer por meio de uma conexão permanente como *Ethernet*, ADSL, modem a cabo ou outros, ou de uma conexão temporária, como uma conexão discada por modem, conforme pode ser percebido na Figura 8. Estudos estão sendo feitos quanto ao desempenho e custo/benefício dos diversos meios de comunicação possíveis para o canal de retorno.

O sistema europeu DVB padronizou o *DVB - Return Channel through Terrestrial* (DVBRCT) como seu canal de comunicação para transmissão terrestre. A especificação desse padrão de comunicação apresenta um canal de retorno sem fio pela banda VHF/UHF, com o objetivo de integrar o sistema de transmissão com receptor do usuário permitindo plena interação com a emissora de televisão. As aplicações, por esse meio, teriam acesso irrestrito para recepção de vídeos sob demanda e Internet. A vantagem desse sistema é uma alta taxa de transmissão, sem a necessidade de um meio físico, como cabo ou ADSL, eliminando o problema de custo de instalação (DVB Return Channel Terrestrial, 2009).

O sistema brasileiro de TV Digital não definiu uma tecnologia particular para o canal de retorno. Assim, os fabricantes de terminais de acesso estão liberados para construir receptores que estejam disponíveis para qualquer meio de comunicação. Ao contrário do que acontece em outros países, onde o canal de comunicação pode ser estabelecido utilizando redes por cabo já existentes, no Brasil a situação é mais complicada. Comunidades remotas no interior de um imenso país como o Brasil não atraem operadoras de telefonia fixa ou móvel por causa do alto custo de instalação e manutenção de equipamentos em comparação com a falta de perspectiva de retorno sobre o investimento nessas regiões (MARGALHO; FRANCêS; WEYL, 2007). Portanto, o país terá que implementar uma tecnologia de acesso que não prive qualquer parte da população do acesso à informação.

Cinco tecnologias foram selecionadas como passíveis de uso no sistema brasileiro de TV Digital (CAMPISTA et al., 2007). Três delas são com fio e outras duas sem fio, conforme ilustrado na Figura 9.

As possíveis tecnologias para o sistema brasileiro são:

- *Wi-Fi* - O padrão IEEE 802.11, também conhecido como *Ad Hoc*, é uma tecnologia de acesso sem fio que se diferencia das demais por não necessitar de uma estação de acesso fixa. O Wi-Fi elimina a infra-estrutura de estações de conectividade pois

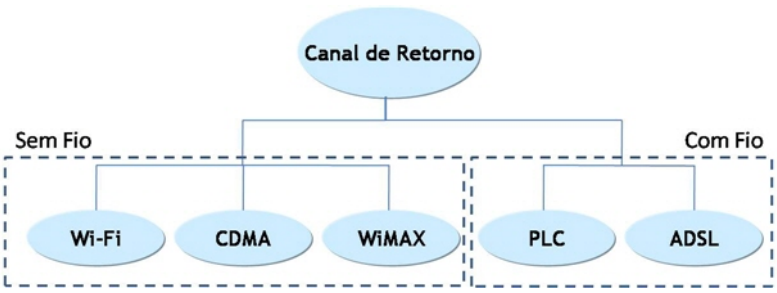


Figura 9: Tecnologias para o Canal de Retorno no ISDTV

cada receptor deve se comunicar com os outros receptores na rede. Esse formato mantém um custo baixo de infra-estrutura pois a rede é auto configurável, e o receptor vizinho colabora como um roteador, enviando os dados para os outros nodos da rede até que chegue ao destino. Entre outras vantagens, pode-se citar a escalabilidade, pois é possível aumentar a rede apenas adicionando novos receptores, e a independência quanto a assinaturas de planos para conexão. Apesar dessa vantagens, rede *Ad Hoc* possui baixa conectividade. Como cada receptor precisa dos outros para enviar informação até o destino, é imprescindível que um número mínimo de receptores estejam ligados para garantir que os dados cheguem ao provedor de serviços. O número de receptores que precisam estar ligados para garantir a conectividade depende da região e topografia em que a rede está localizada.

- *CDMA 1xEv-DO* - Essa é uma nova geração dos padrões CDMA2000 para redes de celulares, conhecida como terceira geração (3G). O *Code Division Multiple Access - Evolution - Data Optimized* (CDMA2000 1xEv-DO) é a evolução do padrão anterior com o objetivo de suportar aplicações de dados. O padrão define tanto envio quanto recebimento de dados, mas com taxas de transmissão assíncronas. Para *download*, alcança taxas de 2,4 Mbps e para *upload*, taxas de 156 kbps (CAMPISTA et al., 2007). O uso do CDMA como canal de retorno para TV Digital necessita de estações de transmissão para conectividade dos terminais de acesso. Assim, cada receptor na residência do telespectador deverá estar conectado diretamente a uma estação de transmissão, o que aumenta consideravelmente o custo de implantação. Além disso, esse tipo de serviço requer que o usuário tenha um plano de assinatura com a prestadora de telefonia celular.

- *WiMAX - World Interoperability for Microwave Access* ou IEEE 802.16 é uma tecnologia sem fio para redes em áreas metropolitanas. Foi desenvolvida com o objetivo de ser uma alternativa ao ADSL e conectividade por cabo, solucionando o problema de instalações até o ponto de acesso na residência do usuário final. O WiMAX demanda uma estação de conexão que permite acesso de vários usuários em velocidade de até 70 Mbps (CAMPISTA et al., 2007). O tráfego de informações fica centralizado na estação de conexão, aumentando os custos de infra-estrutura do sistema.

- *PLC - Power Line Communication* é o uso da rede de energia elétrica para transmissão de dados, concentrada em redes de baixa voltagem. Dessa forma, é possível obter alta capilaridade da rede, considerando que todas as casas com televisor estão conectadas à rede elétrica e assim, todas as casas estariam também conectadas à

rede de comunicação de dados. O custo de instalação é baixo para o usuário, por não haver necessidade de qualquer tipo de compra de serviços extra. A comunicação de dados pela rede elétrica é mais sensível a interferências que outros tipos de redes com fio. Assim, o ambiente da rede e a capacidade de transmissão podem estar limitados pelo tipo de usuário (residencial, industrial, comercial, etc) e pela densidade de usuários (POLO; HIRAKAWA; JUNIOR, 2007)(JUNIOR; HIRAKAWA; JATOBA, 2008). Um outro ponto preocupante quanto ao sistema PLC deve ser levado em consideração. A qualidade da rede elétrica é variável no Brasil. Em algumas regiões de população mais pobre, a infra-estrutura elétrica é precária e ainda inclui a possibilidade de instalações ilegais. A reconstrução ou mesmo reparo dessa rede teria um custo alto, inviabilizando o uso dessa tecnologia (CAMPISTA et al., 2007).

- *ADSL - Assymmetric Digital Subscriber Line* é atualmente a tecnologia mais usada para acesso à Internet no mundo. Esse sistema refere-se ao uso de serviços em linhas telefônicas para transmissão de dados. É assimétrico por apresentar diferentes taxas de transferência para envio e recebimento de dados. O uso de ADSL é tecnicamente viável para uso como canal de retorno na TV Digital, mas apresenta algumas desvantagens. O alcance da atual rede telefônica do país em regiões mais remotas é precário, e necessitaria de investimento em infra-estrutura por parte das companhias telefônicas. Outro ponto negativo é a necessidade de acesso pago ao serviço de comunicação de dados.

A Tabela 2 apresenta uma comparação entre as características das diversas tecnologias para canal de retorno para o sistema brasileiro ISDTV.

Considerando os resultados de análises apresentados por Farias, Carvalho e Alencar (2008), o WiMAX-700 é a tecnologia mais adequada para uso no sistema brasileiro ISDTV. WiMAX-700 é uma nova especificação do WiMAX. A especificação foi desenvolvida na Unicamp (Universidade Estadual de Campinas) para utilizar um perfil de frequências de

Tabela 2: Comparação entre as Tecnologias para Canal de Retorno no ISDTV

	IEEE 802.11 ad hoc	CDMA 1xEV-DO	WiMAX	PLC	ADSL
Plano de Assinatura	Não	Sim	Sim	Sim	Sim
Custo de Infra-Estrutura	Baixo	Alto	Médio	Alto	Alto
Disponibilidade da Rede	Alta	Alta	Alta	Alta	Muito Alta
Taxa de Transmissão (Mbps)	Até 54	Até 2,4 (download) e 0,153 (upload)	Até 70	Até 200	Até 24 (download) e 1 (upload)
Alcance	Centenas de metros	Dezenas de Quilômetros	Dezenas de Quilômetros	Centenas de metros	Poucos Quilômetros

400 MHz a 960 MHz, com o objetivo de adequar-se à TV Digital. Por sua transmissão de dados ser feita por uma rede sem fio, apresenta um custo menor quando comparada a outras tecnologias como ADSL ou acesso discado, que necessitam de uma infra-estrutura de cabeamento. Entre outras vantagens estão a melhoria na cobertura e sinal interno nas residências, o alto alcance de suas antenas que podem estar a até 65 km de distância dos terminais de acessos dos telespectadores, atingindo uma taxa de até 12 Mbps (MELONI, 2007).

2.5 Considerações Finais

Os sistemas internacionais de TV Digital estabeleceram um arcabouço de tecnologias que atendem às demandas de funcionamento da TV Digital interativa. O Brasil, no estudo para implantação de um sistema que atendesse às características do país, estabeleceu seu próprio sistema, denominado ISDTV, que pretende cumprir com essas exigências e ainda melhorar suas funcionalidades, principalmente na área de interatividade.

Neste capítulo, apresentou-se uma visão geral dos sistemas internacionais de TV Digital, abordando suas características e principais vantagens. O sistema brasileiro também foi detalhado, enfatizando os pontos de melhorias nas tecnologias envolvidas. A tecnologia de carrossel de dados e objetos foi descrita, salientando sua importância para o funcionamento de aplicações interativas no contexto da TV Digital. Existem vários meios de comunicação para o canal de retorno, todos com vantagens e desvantagens de escalabilidade e confiabilidade. Um detalhamento sobre a infraestrutura para desenvolvimento de aplicações interativas será mostrado no próximo capítulo, abordando os *middlewares* existentes e a padronização das linguagens de programação.

3 INFRAESTRUTURA DE *SOFTWARE* PARA APLICAÇÕES INTERATIVAS

Uma camada de *middleware* genérica tem o objetivo de diminuir a complexidade e heterogeneidade na comunicação entre diversos sistemas. Essa camada fornece uma plataforma definida de serviços que realizam a comunicação de forma transparente por meio de um conjunto de regras disponíveis em uma biblioteca, e garantem a interface entre as camadas inferior e superior. O *middleware* em um sistema de TV Digital Interativa é responsável por prover uma API genérica às aplicações, padronizando a forma de acesso aos diversos recursos de hardware e software dos terminais de acesso do telespectador. A camada superior (camada de aplicativos) se comunica com a camada inferior (camada de sistema operacional, protocolos, etc) pelos recursos do *middleware*, abstraindo diversidade de *hardware* e *software* dos diversos tipos de dispositivos de acesso, como mostrado na Figura 10. Uma aplicação de TV Digital poderia ser executada diretamente no sistema operacional do terminal de acesso, mas como os sistemas operacionais não estão preparados para fornecer suporte adequado para aplicações específicas de TV Digital, justifica-se a existência de um *middleware*.

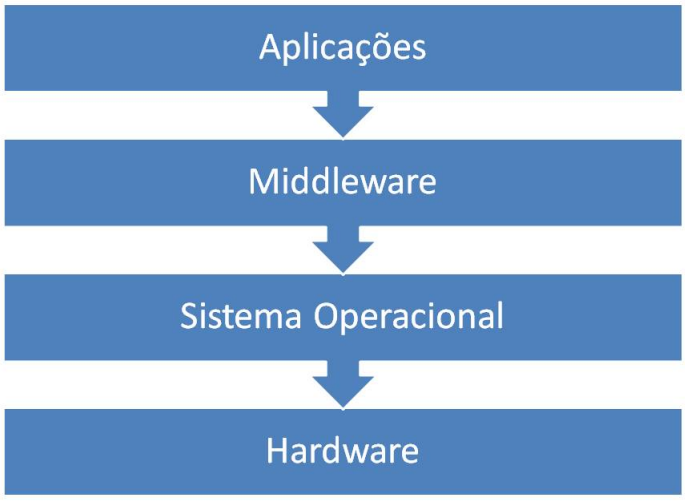


Figura 10: Arquitetura do Middleware

Para o desenvolvimento de aplicações para TV Digital, existem dois tipos de paradigmas de programação: declarativo e procedimental (SOARES; BARBOSA, 2008). A programação do tipo declarativa é aquela cujo objeto inicial de chamada é baseado em uma linguagem declarativa. Linguagens de programação declarativa possuem um alto nível de abstração, enfatizando a descrição declarativa ao invés de uma implementação algorítmica. Linguagens declarativas usualmente estão mais ligadas a um domínio ou objetivo específico, permitindo ao desenvolvedor fornecer instruções de mais alto nível sem se preocupar com a forma como a linguagem vai ser executada. Consequentemente, torna-se uma linguagem mais simples de ser escrita e resultando em programas de tamanho menor.

A programação do tipo procedimental é aquela cujo objeto inicial de chamada é do tipo procedimental. Uma linguagem procedimental deve ser implementada de forma algorítmica, podendo ser baseada em paradigma como programação estruturada ou orientada a objetos. Em um sistema com programação procedimental, o desenvolvedor tem mais poder sobre o código fonte, por ser capaz de fornecer instruções detalhadas de execução. Como será visto a seguir, praticamente todos os *middlewares* apresentam suporte para desenvolvimento declarativo e procedimental. Há também a possibilidade de mesclar os dois tipos de programação, criando-se uma aplicação com linguagem híbrida, sendo parte em programação declarativa e parte em programação procedimental.

São várias as camadas que compõem o processo de transmissão de TV: codificação de áudio e vídeo, sistema de transporte, modulação e camada de aplicação (SOARES; BARBOSA, 2008). Para a camada de aplicação, um dos primeiros padrões adotados nos sistemas de TV Digital foi definido em 1997 e conhecido como *Multimedia and Hypermedia Experts Group* (MHEG). Esse padrão usava notações para definir aplicações multimídia baseadas no relacionamento de objetos, com a linguagem declarativa *Nested Declarative Language* (NCL). O MHEG evoluiu juntamente com a portabilidade da linguagem Java. Os componentes em linguagem Java não foram implantados, mas a idéia serviu como base para padrão de TV Interativa (DAVIC, 2008), que teve várias de suas APIs adotadas pelo *Multimedia Home Platform* (MHP) (ETSI, 2003).

O primeiro padrão de camada de aplicação de *middleware*, puramente em Java, é o MHP. O MHP, padrão adotado pelo sistema europeu DVB, evoluiu para o *Globally Executable* MHP (GEM) (Digital Video Broadcasting, 2008). Aos poucos o ambiente procedimental Java foi adaptando-se também às linguagens declarativas de forma que linguagens procedimentais e declarativas pudessem funcionar juntas.

O sistema de TV Digital americano possui como seu *middleware* padrão o *Advanced*

Common Application Platform (ACAP) no qual Java é parte da solução de execução de aplicações. Tanto o MHP quanto ACAP incluem suporte para linguagens declarativas como XHTML e *scripts* como ECMAScript (ECMAScript, 2009). O *middleware* japonês, ARIB, e o americano, ACAP, recentemente adotaram o GEM como especificação base para desenvolvimento de aplicações em Java (BATISTA et al., 2007).

O *middleware* adotado pelo ISDTV-T é o Ginga (SOUZA; LEITE; BATISTA, 2007) (SOARES; RODRIGUES; MORENO, 2007). O *middleware* brasileiro Ginga suporta tanto aplicações com linguagens declarativas utilizando a linguagem NCL com o subsistema Ginga-NCL, quanto aplicações em linguagem Java com o subsistema Ginga-J. O Ginga-J incorpora muitas inovações e, recentemente, adotou a API JavaDTV (Sun Microsystems, 2009a) como padrão de implementação. A especificação JavaDTV foi desenvolvida pela *Sun Microsystems* com o apoio do Fórum SBTVD.

A plataforma *Java Interactive Television* (JiTV) (JiTV, 2008)(SANTOS JUNIOR et al., 2007) é uma forma de validação de requisitos do SBTVD-T. Desenvolvida no ambiente do Laboratório de Televisão Digital Interativa da PUC Minas em Poços de Caldas, essa plataforma promove o desenvolvimento de aplicações para TV Digital Interativa, com ferramentas de autoria, *middleware* de distribuição de mídia e tocadores multimídias para visualização e interação com o usuário.

A seguir, serão detalhados os *middlewares* presentes nos principais sistemas de TV Digital do mundo, abordando, principalmente, as suas funcionalidades quanto ao desenvolvimento de aplicações interativas com uso do canal de retorno. Também serão apresentadas as tecnologias envolvidas e o processo de desenvolvimento de aplicações para TV Digital.

3.1 Padrões de *Middleware*

3.1.1 MHP

O *middleware Multimedia Home Platform* (MHP) foi desenvolvido para o DVB, sistema europeu de TV Digital, com o objetivo de padronizar os elementos da plataforma de receptores, como a arquitetura do terminal de acesso, protocolos de transmissão, máquina virtual para execução de aplicações e suas APIs (YANG et al., 2008). MHP define uma interface genérica entre as aplicações interativas e os terminais de acesso em que são executados. Essa interface permite que fornecedores de aplicações alcancem todos os tipos de terminais de acesso, desde o mais simples até o mais sofisticado, garantindo o sucesso

de implantação da plataforma. O MHP atinge todos os padrões de difusão permitidos pelo DVB, como satélite, cabo ou terrestre (VRBA; CVRK; SYKORA, 2006).

O MHP suporta modelos de aplicações baseados em linguagem procedimental e declarativa. Para as aplicações procedimentais, adota a execução de aplicações Java TV (Sun Microsystems, 2009c), com a nomenclatura de DVB-J. Para as aplicações em modelo declarativo, o MHP opta pelas aplicações em HTML, com o DVB-HTML. A tecnologia Java foi adotada como ambiente de programação procedimental com a API Java TV e a especificação MHP inclui ainda outras bibliotecas como a *Home Audio Video Interoperability* (HAVi) (HAVi, Inc., 2008) para interface com usuário e o padrão *Digital Audio Visual Council* (DAVIC) (DAVIC, 2008), que está relacionado a funções de mais baixo nível no terminal de acesso, como sintonia do fluxo de transporte ou acesso condicional. As APIs MHP suportam os seguintes pacotes Java (LIN; SUN, 2007): *Personal Java Application Environment* (PJAE), *Java Media Framework* (JMF), *Java TV*, *HAVi User Interface* (HAVi-UI), *DAVIC Media*, *W3C DOM* (DOM2) e *OpenCard Framework* (OCF).

Com o objetivo de flexibilizar a produção dos terminais de acesso, permitindo aos fabricantes de equipamentos desenvolver equipamentos com diferentes capacidades e custos, o MHP define perfis de terminais de acesso de acordo com as funcionalidades habilitadas, que podem atuar em diferentes segmentos do mercado ou diferentes redes de operação (VRBA; CVRK; SYKORA, 2006). Os três perfis são:

- *Enhanced Broadcast Profile* - direcionado para terminais de acesso de baixo custo, com acesso de interatividade limitado, com canal de retorno por meio, geralmente, de linha telefônica e somente permitindo envio de dados;
- *Interactive Broadcast Profile* - apresenta suporte padrão para o canal de retorno, em que aplicações possam trafegar dados em ambas as direções (envio e recebimento);
- *Internet Access Profile* - permite o uso de aplicações gerais de acesso à Internet como *e-mail*, navegador e aplicações por IP.

Assim como as aplicações interativas podem ser inicializadas automaticamente quando um canal de TV é sintonizado, o *middleware* MHP possibilita a inclusão de aplicações que podem ser executadas posteriormente, de acordo com a vontade do telespectador. Alguns exemplos do uso dessa funcionalidade são a apresentação de um determinado item de compra durante a sua exibição em um filme, em que se deve executar a aplicação imediatamente à aparição do produto, ou um conjunto de jogos que podem ser executados

quando o telespectador desejar. As aplicações interativas podem ser armazenadas para serem executadas posteriormente por uma ação direta do usuário ou pode também ser executada por outra aplicação.

Outra relevante característica em relação ao carregamento de aplicações pelo *middleware* MHP é a possibilidade de descarga e sinalização de aplicações através do canal de retorno usando o protocolo HTTP e não somente através da difusão (PIESING, 2006). Para a interface de rede com o canal de retorno, os protocolos obrigatórios definidos na especificação são: IP, TCP, UDP, DNS e HTTP.

As aplicações no ambiente no *middleware* MHP podem ser executadas por meio de *plug-ins*. Um *plug-in* representa uma estratégia de execução e interpretação de aplicações e formatos de conteúdo que não estão especificados no padrão MHP. Assim, as aplicações podem ser desenvolvidas em DVB-J, DVB-HTML, ou em qualquer outra linguagem ou código nativo que possam ser executados por algum *plug-in*. As aplicações que não necessitam de *plug-in* para executar são chamadas de MHP compatíveis.

Atualmente a especificação do MHP está padronizada e disponível no mercado em muitos países da Europa. A estratégia de evolução do *middleware* está voltada para área de aplicações, com melhorias de suas implementações por parte de terceiros como a *Sun Microsystems*, que deseja construir aplicações mais rápidas (MIRABELLA; BRISCHETTO; RAUCEA, 2008), e também para a interoperabilidade de comunicação com outros *middlewares* com o GEM, que será detalhado na Seção 3.1.6.

3.1.2 DASE

O *middleware DTV Application Software Environment* (DASE) foi desenvolvido nos Estados Unidos pelo grupo *Advanced Television Systems Comittee* (ATSC) para ser a referência para transmissão terrestre. O DASE, em sua primeira versão, contemplava somente interatividade local, pois o sistema ATSC não previa o canal de retorno. Com a evolução do padrão norte-americano e a inclusão da utilização do canal de comunicação, ficou estabelecida a integração da TV com a Internet.

O DASE é composto por ambientes de execução independentes chamados *Procedural Application Environment* (PAE), que é composto por uma máquina virtual Java, e *Declarative Application Environment* (DAE), que permite a utilização de linguagens declarativas como HTML e JavaScript (ATSC, 2003). O ambiente de aplicação declarativa corresponde ao sistema que interpreta documentos XHTML, folhas de estilo CSS, *scripts*

ECMAScript e suporte a DOM, possibilitando que uma aplicação declarativa tenha acesso aos recursos do terminal de acesso.

A especificação do ECMAScript possibilita às aplicações declarativas um aumento de poder de construção por permitir que essas aplicações compartilhem recursos do ambiente procedimental, definindo uma ponte de comunicação entre os dois ambientes. Assim, uma aplicação declarativa DASE pode utilizar de recursos de uma aplicação procedimental, caracterizando a possibilidade de aplicações de ambiente híbrido.

A especificação do *middleware* DASE não define o conceito de *plug-in*. Dessa forma, é possível que o ambiente declarativo funcione sem a necessidade de coexistência do ambiente procedimental. Por outro lado, é possível a difusão de uma aplicação procedimental que simule o ambiente declarativo, em terminais que não possuam o ambiente declarativo nativamente.

3.1.3 OCAP

O *middleware OpenCable Application Platform* (OCAP) é um derivado do DVB-MHP produzido pela *CableLabs* para TV a cabo nos Estados Unidos e Canadá. No nível técnico, várias tecnologias do padrão Europeu DVB foram removidas ou substituídas por equivalentes funcionais no padrão americano (PIESING, 2006). Mesmo o OCAP sendo baseado no MHP, existem diferenças significativas entre eles. A primeira versão do OCAP foi desenvolvida baseada no MHP 1.0.2. A versão mais atual é baseada num subconjunto do MHP, chamado *Globally Executable MHP* (GEM), que será detalhado na Seção 3.1.6.

As aplicações desenvolvidas para OCAP não podem ser consideradas completamente compatíveis com aplicações MHP. Uma importante diferença é a remoção e substituição de algumas funcionalidades. Outra consideração relevante é o número de extensões e restrições impostas às aplicações OCAP. Para diferenciar as aplicações OCAP de aplicações MHP, foi definido na especificação do *middleware* os termos OCAP-J e OCAP-HTML para se referir a aplicações Java e HTML respectivamente. Mesmo que as aplicações OCAP e MHP compartilhem grande parte das funcionalidades de APIs, elas não são idênticas.

Como herança do MHP, o *middleware* OCAP, no seu ambiente procedimental, utiliza a API Java TV e, além disso, adiciona outras diversas instruções para controle de funções específicas do terminal de acesso, incluindo fornecimento de acesso aos dados de difusão e outros dados do fluxo de informação, decodificação e exibição de mídias. Para o ambiente

declarativo, a especificação do OCAP suporta aplicações DVB-HTML que inclui XHTML, CSS, ECMAScript e outras extensões definidas pelo DVB, permitindo a interpretação de aplicações declarativas de uma forma mais flexível.

OCAP tem suporte para execução de aplicações que não estão vinculadas a qualquer serviço de TV. Com isso, aplicações podem continuar executando mesmo que o telespectador troque o canal do televisor. Esse tipo de aplicação usualmente está relacionado a serviços de gravação de programação, jogos ou serviços genéricos de interação.

3.1.4 ACAP

Mesmo com a existência de dois *middlewares* e a tendência de padronização proposta pelo GEM, o ATSC ainda introduziu uma nova especificação de *middleware* denominada *Advanced Common Application Platform* (ACAP). O propósito desse terceiro sistema é ser uma base comum para todos os sistemas de TV interativa nos Estados Unidos, independente da forma de difusão, seja cabo, satélite ou terrestre. O ACAP é também baseado no GEM, herança obtida do OCAP, de quem incorpora outras funcionalidades que são específicas do padrão americano com o objetivo de aumentar o grau de interoperabilidade com o padrão ATSC . A essência do *middleware* ACAP está na unificação dos seus antecessores DASE e OCAP, promovendo a especificação de uma única plataforma comum. Essa plataforma tem o objetivo de ter uma arquitetura bem definida e modelos de execução e programação bem claros, proporcionando aos desenvolvedores de aplicações a segurança de que seus programas executem nos vários modelos de terminais de acesso (RICHER et al., 2006).

O *middleware* ACAP permite a execução de aplicações no modelo procedimental com o sistema ACAP-J e de aplicações no modelo declarativo com o sistema ACAP-X, além de permitir aplicações híbridas. As especificações do modelo procedimental são baseadas nos padrões GEM e OCAP, possibilitando que a execução dos mesmos arquivos de Java implementados para o MHP, além de algumas APIs de extensão. O ambiente declarativo ACAP-X, cuja implementação é opcional, é baseado no *middleware* DASE, com algumas adaptações para manter compatibilidade com o padrão GEM.

3.1.5 ARIB

O *Association of Radio Industries and Businesses* (ARIB) é o *middleware* do padrão japonês de TV Digital (ISDB). Esse *middleware* foi inicialmente concebido contendo ape-

nas um ambiente declarativo. Posteriormente, foi definido no padrão ARIB um sistema procedimental (ARIB, 2004). As definições dos ambientes declarativo e procedimental são instituídas pelos padrões *Data Coding and Transmission Specification for Digital Broadcasting* (ARIB STD-B24) e *Application Execution Engine Platform for Digital Broadcasting* (ARIB STB-B23), respectivamente. O ambiente declarativo define como linguagem padrão a *Broadcast Markup Language* (BML) e o ambiente procedimental, que é baseado no padrão europeu (MHP) com extensão do GEM, permite a execução de aplicações interativas baseadas em Java por meio de uma máquina virtual com uma biblioteca genérica de acesso aos recursos dos terminais.

A linguagem BML, assim como a linguagem DVB-HTML, é baseada em XHTML, com suporte a CSS e ECMAScript. A linguagem ECMAScript, no contexto do *middleware* ARIB foi definida com o objetivo de permitir alterações no conteúdo dos documentos BML, sem a possibilidade de acessar aplicações procedimentais como ocorre no DVB-HTML e DASE.

Os ambientes declarativo e procedimental no *middleware* ARIB são completamente independentes, mas também podem funcionar compartilhando recursos, apesar de não haver uma definição ou restrição explícita sobre a comunicação dos dois ambientes. É possível ainda o desenvolvimento de *plug-ins* com o objetivo de interpretar formatos de conteúdos ou linguagens que não são suportados pela plataforma.

3.1.6 GEM

O grupo DVB propôs uma especificação unificada de *middlewares* para TV Digital, chamada *Globally Executable MHP* (GEM). O objetivo era permitir a execução de aplicações MHP em outras plataformas, incluindo funcionalidades que não são específicas do sistema DVB (SOUZA; LEITE; BATISTA, 2007). Dessa forma, os principais padrões mundiais de TV Digital aderiram a essa integração. O sistema ATSC propôs o padrão ACAP, incluindo o GEM em sua definição, com o objetivo de relacionar os padrões MHP, DASE e OCAP. Da mesma forma, o sistema japonês incorporou o GEM ao seu *middleware* ARIB STB-B23. A especificação GEM se concentra na parte comum dos diversos *middlewares*, como protocolos de rede para canais de retorno, formatos de conteúdo de exibição e uso de máquina virtual Java para aplicações interativas. O ambiente declarativo não é relevante para o GEM, cada *middleware* fica liberado para definir sua própria arquitetura para esse ambiente.

O GEM é baseado na especificação MHP e inclui a maioria de seus elementos básicos.

Diferente do MHP, no GEM são definidos apenas dois perfis: o *Enhanced Broadcast* e o *Interactive Broadcast*. Foi adicionada à especificação uma estratégia de substituir elementos GEM que conflitam com as funcionalidades de um padrão específico. Alguns elementos podem ser removidos por incompatibilidade técnica ou por razões comerciais, garantindo assim a harmonização entre os diversos padrões mundiais. Por exemplo, o uso do carrossel de dados do MHP é incompatível tecnicamente com a língua japonesa e precisou ser substituído para utilização no ARIB. De forma semelhante, para o *OCAP*, que trabalha especificamente com operadoras de TV a cabo, foi necessário substituir o navegador de aplicações que possui influência sobre a política de liberação das operadoras que desejam ter mais controle sobre o receptor de TV. A Figura 11 ilustra a integração entre os diversos *middlewares* mundiais e o GEM

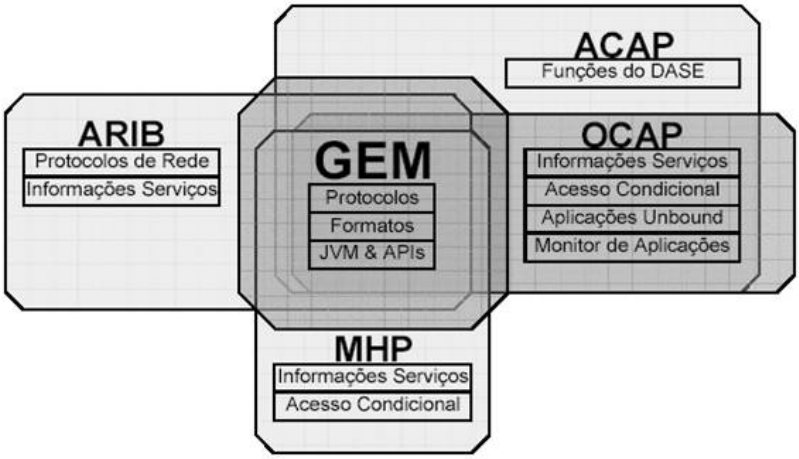


Figura 11: Relação entre GEM e outros *middlewares* (LEITE et al., 2005)

A forma de integração com o GEM é por meio de uma lista de equivalentes funcionais. Essa lista define quais elementos têm seu equivalente funcional definido ou quais são opcionais. Um extensa tabela de equivalentes funcionais se encontra em ETSI (2005). O GEM está definido com um núcleo comum que é representado pelo conjunto de APIs e garantias funcionais que deve estar presente em todos os receptores GEM. E pode ser estendido com o uso de APIs específicas para cada ambiente. Existem ainda algumas considerações de portabilidade que desenvolvedores deverão estar atentos para maximizar a compatibilidade na implementação de aplicações para o terminal de acesso. Na verdade, GEM é um *framework* a partir do qual se pode implementar um *middleware* para o terminal de acesso. Sua especificação não define completamente um terminal de acesso, mas simplesmente expõe a forma de se adaptar para obter uma conformidade que garanta a execução global de aplicações (LEITE et al., 2005).

3.1.7 Ginga

Ginga é o *middleware* do Sistema Brasileiro de TV Digital (SBTVD). Como ocorre nos demais *middlewares* internacionais, as aplicações para serem executadas no Ginga também se classificam entre as categorias procedimental e declarativa. Para o Ginga, as aplicações procedimentais são aquelas escritas em linguagem Java e as aplicações declarativas são as escritas em linguagem NCL. A especificação Ginga define dois ambientes distintos para atender cada uma das categorias separadamente, sendo Ginga-J para aplicações procedimentais e Ginga-NCL para aplicações declarativas (SOUZA; LEITE; BATISTA, 2007) (SOARES; RODRIGUES; MORENO, 2007). O Ginga-J foi originalmente desenvolvido no Laboratório de Aplicações em Vídeo Digital (Lavid) da Universidade Federal da Paraíba (UFPB) , com o nome de Flex-TV, que ao unir-se ao Maestro, formatador da linguagem NCL criado no laboratório Telemídia da PUC-Rio, deu origem ao Ginga.

O *middleware* Ginga também possibilita a execução de aplicações que contêm objetos tanto do ambiente declarativo quanto do ambiente procedimental. Essas aplicações são denominadas híbridas e podem usufruir de funcionalidades de ambos os ambientes. Para isso, o *middleware* Ginga definiu uma estratégia de comunicação entre os dois ambientes por meio de uma ponte bidirecional (*bridge*). Por um caminho, essa ponte possibilita que aplicações declarativas façam chamadas às aplicações procedimentais em Java. Pelo caminho inverso, as aplicações em Java podem monitorar qualquer evento de linguagem declarativa e ainda alterar aspectos de marcadores dessas aplicações (SOARES; FILHO, 2007) (SOARES; RODRIGUES; MORENO, 2007).

Algumas funcionalidades do *middleware* que atendem tanto o ambiente declarativo quanto o ambiente procedimental são atendidas pelo *Ginga Common Core*. Esse módulo é composto por decodificadores de conteúdo comum para decodificação e apresentação de tipos de conteúdos como PNG, JPEG, MPEG e outros formatos, além de funcionalidades para transporte de conteúdos por fluxo MPEG-2 e através do canal de retorno (SOARES; FILHO, 2007) (SOARES; RODRIGUES; MORENO, 2007) (SOUZA; LEITE; BATISTA, 2007). A arquitetura de alto nível do do *middleware* Ginga é apresentada na Figura 12.

O subsistema Ginga-NCL interpreta documentos NCL, que foi escolhida como linguagem a ser adotada pelo *middleware* brasileiro por possuir uma separação entre conteúdo e estrutura, tornando-se uma linguagem bastante flexível. Assim, NCL não está restrita somente a determinados tipos de mídia, podendo representar imagens, objetos de vídeo, objetos de áudio e objetos de execução como aplicações Java e outras, dependendo apenas que existam tocadores de mídia específicos embarcados no formatador

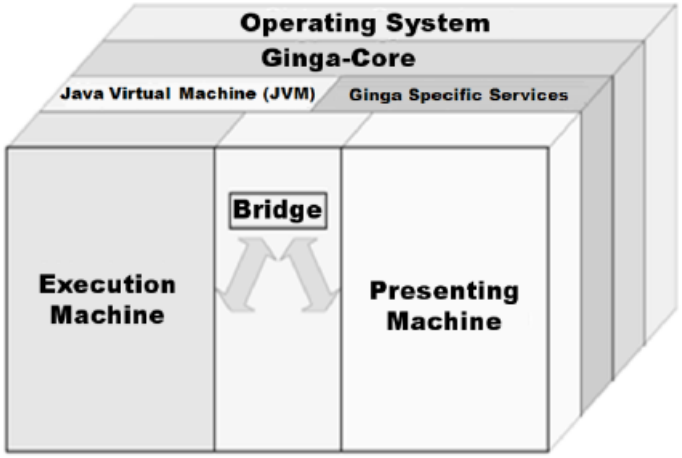


Figura 12: Arquitetura Ginga (SOUZA; LEITE; BATISTA, 2007)

NCL (SOARES; FILHO, 2007). Além disso, a linguagem NCL permite o relacionamento de sincronismo com outros objetos do documento. Outros importantes módulos do Ginga-NCL são o navegador XHTML-*based*, que inclui interpretadores de CSS e ECMScript, e a máquina LUA que é responsável pela interpretação de *scripts* LUA (SOARES; RODRIGUES; MORENO, 2007). Ainda é possível ter navegadores BML, DVB-HTML e ACAP-X embarcados no tocador de documentos NCL, por meio de interpretadores enviados por difusão como Java *plug-in*. Algumas funcionalidades da linguagem *Synchronized Multimedia Integration Language* (SMIL) também estão incorporadas pela linguagem NCL (SOARES; RODRIGUES; MORENO, 2007).

O Ginga-J é a parte do *middleware* responsável por execução de aplicações em linguagem Java. Inicialmente, o Ginga-J era baseado no padrão Europeu, usando como núcleo principal o GEM. Mesmo o GEM usando estruturalmente o Java TV da *Sun Microsystems*, as extensões feitas na Europa (HAVi, Davic, DVB *Extensions*) são controladas por patentes, gerando a cobrança de *royalties* pelo simples uso da especificação. Em junho de 2008, foi formada uma comissão para definição do Ginga-J com a estratégia de criação de um projeto conjunto entre a *Sun Microsystems* e o Fórum SBTVD para a especificação do padrão JavaDTV, que seria funcionalmente equivalente ao GEM. Em dezembro de 2008, a especificação JavaDTV foi liberada com direitos de copropriedade do Fórum SBTVD e da *Sun Microsystems*, sem a cobrança de *royalties* para todos os implementadores. Essa nova especificação mantém a mesma base de desenvolvimento (Java TV e JMF) usada nos *middlewares* internacionais. Em maio de 2009, o Conselho Deliberativo do Fórum SBTVD aprovou o JavaDTV como linguagem para o Ginga-J (Fórum SBTVD, 2009b).

A adoção do GEM pelo Ginga-J garantia a compatibilidade com a maioria dos *middlewares* atuais de TV Digital, mas o valor pago em *royalties* seria muito elevado. JavaDTV possui as mesmas funcionalidades do GEM, livre de *royalties*, e, como desvantagem, a necessidade de maior tempo para o desenvolvimento e penetração em outros mercados. A definição da especificação atualmente está em consulta pública (Fórum SBTVD, 2009a), e não tem valor normativo. Assim que aprovado, o Ginga-J será a única especificação mundial sem cobrança de *royalties*. Outro benefício da plataforma JavaDTV é a possibilidade de produzir aparelhos de TV e/ou receptores por um custo muito mais baixo, já que o preço final do *software* será menor.

O JavaDTV é um conjunto de APIs que fornecem possibilidade de portabilidade entre aparelhos utilizando recursos de uma máquina virtual Java. Suporta as principais APIs da *Sun Microsystems* para dispositivos de recursos limitados: *Connected Device Configuration* (CDC) 1.1, *Foundation Profile* (FP) 1.1 e *Personal Basis Profile* (PBP) 1.1. Dessa maneira é possível a apresentação de aplicações em diversos dispositivos independentemente de sua interface gráfica. O JavaDTV ainda define parcialmente a especificação do *LightWeight User Interface Toolkit* (LWUIT). A arquitetura detalhada do Ginga é mostrada na Figura 13. As funcionalidades dessas APIs estão detalhadas na Seção 3.3.

O Ginga é capaz de executar aplicações interativas persistentes (embarcadas pelo fabricante diretamente na memória do receptor) ou transmitidas via sinal digital pelas redes de TV ou mesmo pelo canal de retorno (SOUZA; LEITE; BATISTA, 2007). Outra

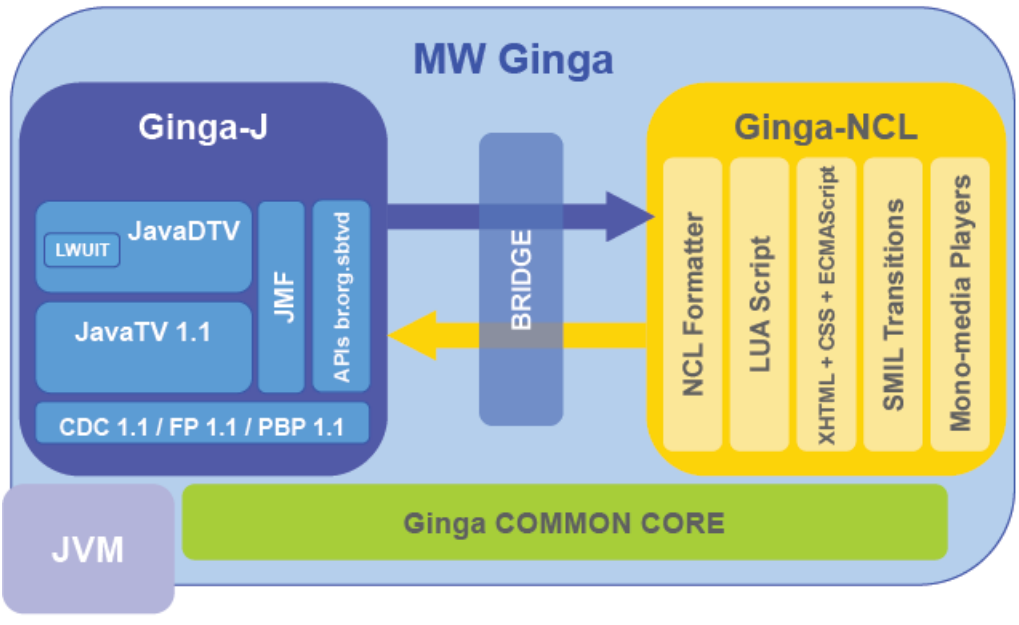


Figura 13: Arquitetura Ginga Detalhada

inovação é a possibilidade de execução/interpretação de aplicações não relacionadas ao serviço (canal) de TV, permitindo a existência de aplicações de serviços genéricos e suporte a múltiplas aplicações executando simultaneamente (Fórum SBTVD, 2009a). É possível também carregar aplicações provenientes do canal de retorno. Para validar a especificação Ginga-J, o Laboratório de Aplicações em Vídeo Digital (Lavid), da Universidade Federal da Paraíba (UFPB) está desenvolvendo uma implementação de referência com código aberto chamada OpenGinga.

Os terminais de acesso do telespectador, no ambiente Ginga, podem ser classificados entre duas categorias: *Full-seg* e *One-seg*. Os receptores *Full-seg* representam dispositivos de maior capacidade de processamento e memória como televisores e terminais externos à TV. E os receptores *One-seg*, por outro lado, são os receptores com menor capacidade, como *hand-helds* e aparelhos celulares. O ambiente Ginga-NCL é mandatório para todos receptores, já o ambiente Ginga-J é mandatório somente para receptores *Full-seg*. O canal de retorno é opcional para os receptores *Full-seg* e obrigatório para os receptores *One-seg*.

3.1.8 Comparação entre os middlewares

Os dois paradigmas de programação, procedimental e declarativo, são utilizados por praticamente todos os *middlewares* mundiais de TV Digital (SOARES; BARBOSA, 2008). Porém, alguns *middlewares* disponibilizam somente o ambiente declarativo de forma pura ou híbrida e outros disponibilizam somente o ambiente procedimental. A Tabela 3 ilustra essa comparação.

Os ambiente declarativos dos *middlewares* dos sistemas de TV digital europeu (DVB-HTML), americano (ACAP-X) e japonês (BML) apresentam máquinas de apresentação para controle de suas aplicações declarativas com navegadores XHTML. XHTML é uma linguagem projetada para navegação de informações textuais, privilegiando a interatividade em detrimento à sincronização e adaptabilidade do conteúdo. A sincronização e adaptabilidade de conteúdo podem ser alcançadas com o uso da linguagem ECMAScript. O *middleware* brasileiro Ginga tenta solucionar o problema da linguagem XHTML com a linguagem NCL que é capaz de obter a sincronização e adaptabilidade de conteúdo sem a necessidade de *scripts* auxiliares. Isso porque a linguagem NCL define uma separação entre conteúdo e estrutura do aplicativo (SOARES; BARBOSA, 2008).

Existem algumas divergências na forma como os diversos *middlewares* possibilitam a incorporação de novas funcionalidades. O *middleware* DASE permite a adição de novos interpretadores para diferentes aplicações. Esses interpretadores podem ser executados

Tabela 3: Comparação entre *middlewares* (SOARES; BARBOSA, 2008)

Middleware	Sistema de TVD	Ambiente Declarativo	Ambiente Procedural
ACAP	Americano/ATSC	ACAP-X [ATSC A-101 2005] (linguagem declarativa = XHTML like; linguagem não-declarativa = ECMAScript)	ACAP-J [ATSC A-101 2005] (linguagem procedural = Java)
MHP	Europeu/DVB-T	DVB-HTML [ETSI TS 102 8121 V1.2.2, 2006] (linguagem declarativa = XHTML like; linguagem não-declarativa = ECMAScript)	MHP [ETSI TS 102 812 V1.2.2, 2006] (linguagem procedural = Java)
ARIB-BML	Japonês/ISDB-T	ARIB – BML [ARIB B-24 2004] (linguagem declarativa = BML XHTML like; linguagem não-declativa= ECMAScript)	Opcional (GEM [ETSI TS 102 819 V1.3.1 2005] like); não implementado)
Ginga	Brasileiro/SBTV	Ginga-NCL [ABNT NBR 15606-2 2007] (linguagem declarativa = NCL; linguagem não declarativa = Lua)	Ginga-J (Linguagem procedural = Java)

tanto no ambiente declarativo quanto procedimental, dependendo do tipo de interpretador. O *middleware* MHP suporta o conceito de *plug-ins*, que também podem ser alocados em diferentes ambientes. O *middleware* ARIB permite *plug-ins* somente no ambiente procedimental, fornecendo ao ambiente declarativo somente a possibilidade de utilização da linguagem BML.

Outro ponto de comparação entre os *middlewares* é a definição do perfil de interatividade. O *Enhanced Broadcast* do MHP, que não caracteriza um canal de retorno, se assemelha ao DASE nível 1. O *Interactive Broadcast* do MHP apresenta suporte padrão para o canal de retorno enquanto o DASE nível 2 define pequenas condições de interatividade. O DASE nível 3 e o perfil Internet Access do MHP são idênticos quanto à interatividade e integração com a Internet. O *middleware* ARIB não define perfis em suas especificações, criando conformidades que se vinculam aos tipos de funcionalidades de acordo com a classe do *hardware* dos terminais de acesso. O ACAP define os perfis ACAP-X *only*, ACAP-X e ACAP-J, que, em harmonização com o GEM, são equivalentes funcionais aos perfis do MHP.

3.2 Plataforma JiTV

A plataforma *Java Interactive Television* (JiTV) (SANTOS JUNIOR et al., 2008)(SANTOS JUNIOR et al., 2007) disponibiliza um ambiente para construção de conteúdo multimídia em formato digital, especialmente protótipos de aplicações interativas para TV

Digital. Está em desenvolvimento, desde 2002, no Laboratório de Televisão Digital Interativa da PUC Minas. A plataforma fornece um ambiente de validação de requisitos do sistema brasileiro de TV Digital (SBTVD), permitindo o desenvolvimento de aplicações interativas para TV Digital com um conjunto de ferramentas para autoria, distribuição e tocadores multimídia para visualização e interação com o telespectador. Entre as principais funcionalidades da plataforma JiTV estão a descrição do conteúdo baseado em esquemas XML e a implementação de aplicações interativas usando linguagem Java.

Na concepção da plataforma JiTV, foram consideradas as características de desenvolvimento do *middleware* Ginga que fornece aspectos inovadores que contribuíram para a evolução dessa plataforma. Novos cenários baseados no *middleware* brasileiro são possíveis devido a essa integração. A Figura 14 ilustra a harmonia entre a plataforma JiTV e o *middleware* Ginga.

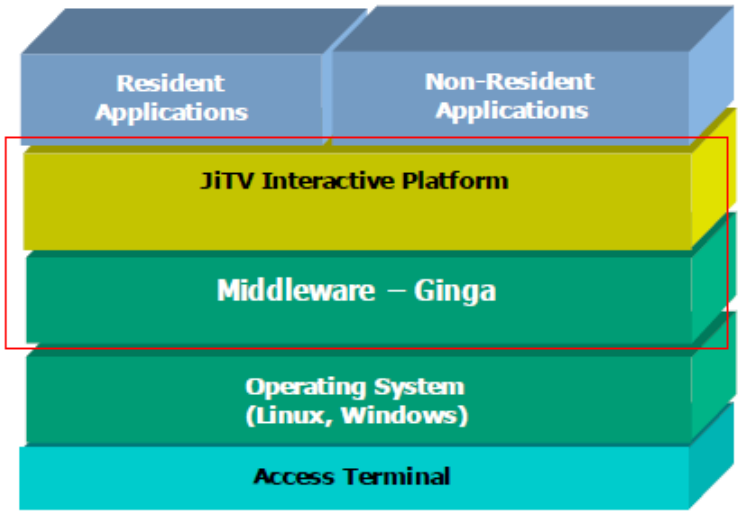


Figura 14: Arquitetura JiTV (SANTOS JUNIOR et al., 2007)

A plataforma JiTV, com seus componentes e módulos de *software*, permeia várias fases do ciclo de vida de uma aplicação interativa, iniciando na produção e distribuição pelo carrossel de dados, chegando à formatação e exibição de diversos conteúdos no terminal de acesso, e interação local ou através do canal de retorno (SANTOS JUNIOR et al., 2009). De um modo geral, a plataforma permite ações de autoria suficientes para desenvolver uma aplicação interativa e formatar sua distribuição para os telespectadores. A formatação do conteúdo é feita de acordo com o terminal de recepção do telespectador, podendo se adequar a diversos dispositivos como televisores, celulares e computadores pessoais.

Relacionando as principais características da plataforma JiTV, pode-se então defini-la como uma plataforma de *software* multi-meio (ambientes IP e *broadcasting*) para desen-

volvimento integrado de aplicações com a possibilidade de integração entre diferentes sistemas de *middleware*, sistemas de acesso condicional e servidores de mídia, e, incluindo, do ponto de vista do receptor, interfaces para navegação entre as diversas aplicações disponíveis pelo telespectador.

A plataforma é composta de duas suítes principais (SANTOS JUNIOR et al., 2009):

- *JiTV Application Development Kit* (JADK) - possui suporte para aplicações Java, suporte para comunicação externa (infra-vermelho, serial, USB, *bluetooth*) para uso de controle remoto e manipulação de objetos de mídia. Compõe-se dos recursos para geração de carrossel de dados, seleção de modo de programação (mono e multi-programação), geração de metadados e geração de fluxo de transporte como pode ser observado na Figura 15.

- *JiTV Application Suite* (JAS) - possui suporte para apresentação e interação com o telespectador, com tocador multi-dispositivo para apresentação de objetos multimídia e interação com o telespectador, além de ferramenta para exibição de Guia Eletrônico de Programação, ferramenta para geração de aplicações interativas para votação e entretenimento, chamada JiTV *QuizComposer*, ferramenta para envio e recebimento de *e-mails* e *framework* que suporta apresentação e execução de aplicações e serviços genéricos (SANTOS JUNIOR et al., 2007).

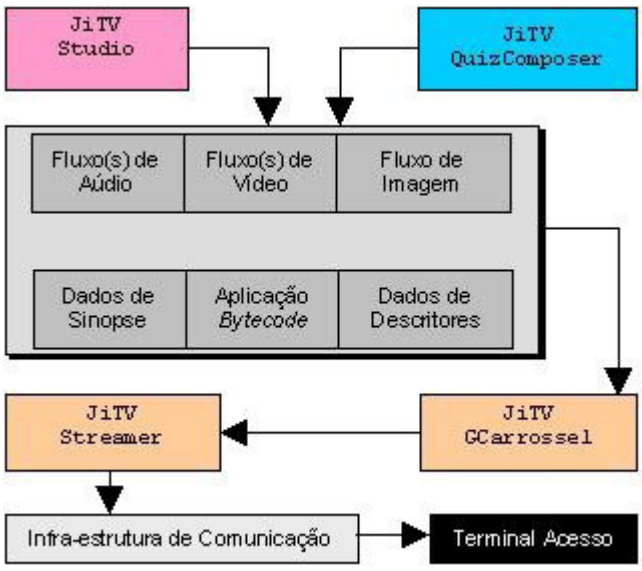


Figura 15: Arquitetura JiTV - produção e distribuição (SANTOS JUNIOR et al., 2008)

Na plataforma JiTV, o módulo JiTVPSI *Studio* foi desenvolvido para suportar as formas de comunicação entre: (1) da difusora para o telespectador, (2) do telespectador

para o PSI (provedor de serviços interativos), (3) da difusora para o PSI e (4) do PSI para a difusora, conforme mostra Figura 16. Para suportar essas formas de comunicação, o módulo fornece funcionalidades de receber estruturas de dados, persistir essas estruturas e enviar informação para o PSI. Para operações com a difusora, JiTVPSI *Studio* oferece funcionalidades para filtrar dados nas bases de dados do PSI.

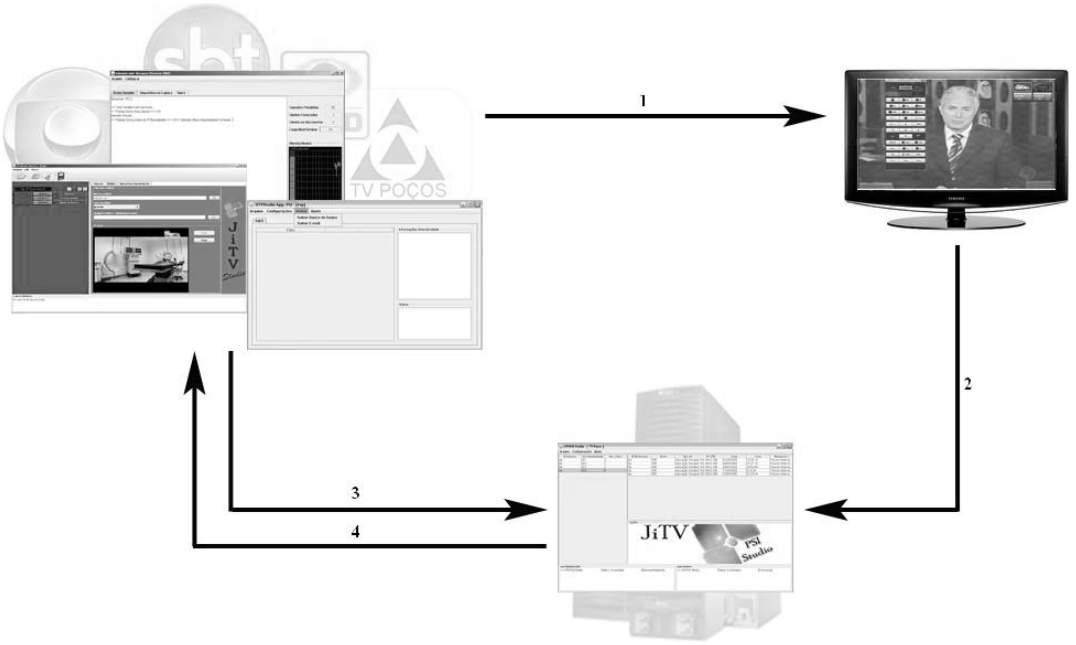


Figura 16: Relacionamento entre entidades usando o JiTVPSI *Studio* (SANTOS JUNIOR et al., 2010)

Para validar os aspectos de interação de aplicações para TV Digital, especialmente quanto ao JiTVPSI *Studio*, três aplicações foram desenvolvidas.

A aplicação *RecommenderTV* foi desenvolvida com o propósito de interação local com o telespectador. É um sistema de recomendação de programação para o telespectador que, ao contrário de Guias Eletrônicos de Programação que se tornaram não atrativos por exigir muito tempo para se encontrar algo entre as várias opções disponíveis, possibilita ao telespectador pesquisar sobre uma lista de recomendações e escolher o programa desejado. Essa aplicação não será testada em conjunto com o projeto apresentado neste trabalho por não possuir interação com servidores remotos, não justificando, assim, a utilização do sistema de comunicação.

Um outra aplicação é chamada JiTVDengue, que foi desenvolvida para coletar informação sobre possibilidade de ocorrência de dengue no ambiente residencial. A aplicação JiTVDengue foi criada com o objetivo de investigar a possibilidade de ocorrência de dengue dentro de residências, permitindo ao telespectador responder quatro questões

sobre a causa da doença. Assim que o usuário responde às questões, os dados são armazenados em formato XML no terminal de acesso e enviados ao JiTVPSI *Studio*, que armazena os dados para análise posterior.

A terceira aplicação, chamada JiTVEleicao, foi desenvolvida para suportar um sistema de votação eletrônica (SANTOS JUNIOR et al., 2010). Essa aplicação simula um sistema para eleições políticas em uma cidade, estado ou país, permitindo o telespectador interagir com o sistema para selecionar o candidato em que deseja votar. Da mesma forma que na aplicação JiTVDengue, os dados são armazenados em XML e posteriormente enviados ao sistema JiTVPSI *Studio* para armazenamento.

3.3 Tecnologias para Desenvolvimento de Aplicações e Serviços em TV Digital

3.3.1 Ambiente de Execução Java

A *Java Runtime Environment*(JRE) é a implementação da especificação Java para uma plataforma específica que garante a abstração da máquina virtual para execução de uma aplicação em qualquer ambiente. É a JRE que garante a portabilidade da plataforma Java, fazendo com que aplicações executem nos mais diversos ambientes e sistemas operacionais, inclusive dispositivos com poucos recursos como aparelhos celulares e receptores de TV. A tecnologia Java é dividida entre três grandes áreas: Java SE (*Standard Edition*) que representa o núcleo da plataforma, recomendada para a maioria do usuários, Java EE (*Enterprise Edition*) que corresponde ao segmento direcionado para uso em recursos avançados no mercado empresarial e Java ME (*Micro Edition*) que é o segmento para utilização em dispositivos com recursos limitados.

A tecnologia Java ME, por ser direcionada a dispositivos de recursos limitados, está baseada em uma configuração que fornece o conjunto básico de bibliotecas e funcionalidades da máquina virtual Java para se adequar ao maior número de dispositivos, focando em estreitar a adequação aos dispositivos por meio de pacotes opcionais para APIs específicas de cada aparelho (Sun Microsystems, 2009b). A Figura 17 ilustra essa arquitetura.

A *Connected Device Configuration* (CDC) é a configuração mais abrangente do Java ME, que fornece uma implementação completa da linguagem Java 1.4. O CDC pode se tornar completamente na API Java SE por meio da adição de bibliotecas complementares. A *Connected Limited Device Configuration* (CLDC) é uma configuração direcionada para telefones celulares e aparelhos de pequeno porte e possui um subconjunto limitado da API

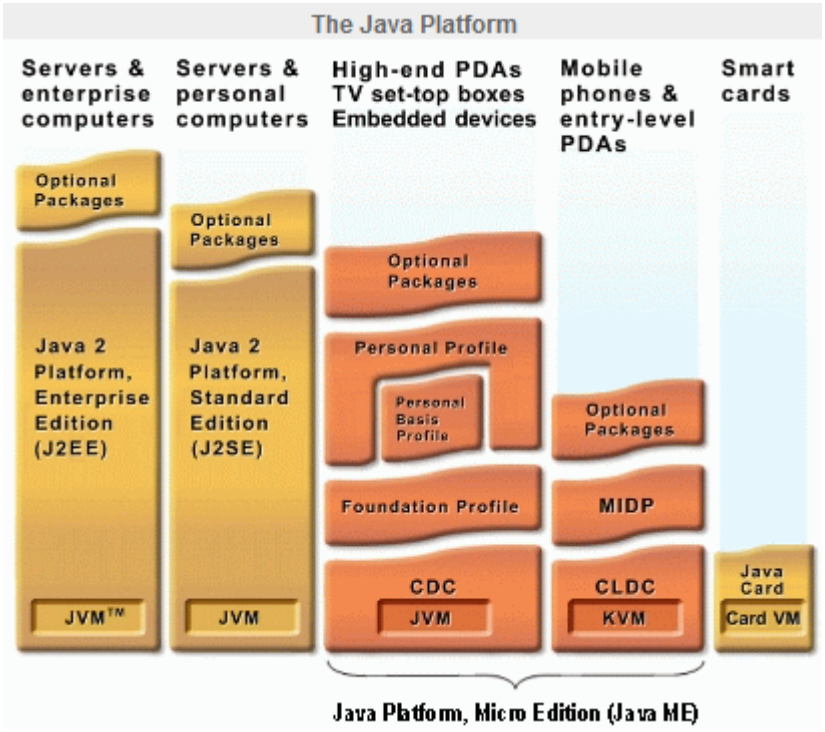


Figura 17: Plataforma Java (Sun Microsystems, 2009b)

Java SE, sendo possível ser executado com apenas 256KB de memória.

A configuração CDC possui a definição de três perfis. Cada perfil adiciona uma quantidade de APIs à configuração original. O *Foundation Profile* (FP) é o perfil básico que incorpora bibliotecas básicas como rede e entrada/saída, representados pelos pacotes *io*, *security*, *network*, *utility*. O *Personal Basis Profile* (PBP) é um perfil intermediário que complementa o perfil anterior adicionando a construção de interfaces por meio de um subconjunto limitado do *Abstract Window Toolkit* (AWT). E, por fim, o *Personal Profile* (PP) é o perfil mais avançado, suportando integralmente o AWT e *applets*.

O ambiente Java para TV Digital para o *middleware* Ginga apresenta a configuração CDC, o perfil PBP ou FP e algumas bibliotecas opcionais (Fórum SBTVD, 2009a). Essa combinação mantém a compatibilidade com a configuração PersonalJava, descontinuada pela *Sun Microsystems*, que está prevista no GEM.

3.3.2 Principais APIs para Desenvolvimento de Aplicações Interativas em TV Digital

Dentre as principais APIs em Java para desenvolvimento de aplicações interativas para TV Digital, pode-se enumerar:

1. Java TV

   Java TV é a API da *Sun Microsystems* para plataforma de TV Digital que define elementos comuns necessários para esse ambiente (Sun Microsystems, 2009c). A API introduz diversos elementos específicos para o ambiente de TV Digital interativa, incluindo a localização de conteúdos, controle sobre o modelo e ciclo de vida de aplicações, acesso a informações e serviços disponibilizados por difusão (seja pela própria API ou pelo *Java Media Framework*). As aplicações que utilizam essa API são também denominados Xlets, que serão vistos em detalhe na Seção 3.3.3. Java TV é uma especificação neutra em relação a definições específicas e, por isso, não está vinculado a nenhum padrão de TV Digital. Assim, Java TV é a API base para grande parte dos *middlewares* de TV Digital no mundo (FERNANDES; LEMOS; SILVEIRA, 2004).

2. JMF

   O *Java Media Framework* (JMF) é uma API que possibilita a utilização e interação de aplicações Java com objetos de mídia, como áudio e vídeo, além de tratar elementos de mídia contínua. JMF é utilizado para capturar, gerenciar fluxo contínuo, processar e apresentar alguns tipos de mídias contínuas.

3. HAVi

   Home Audio Video Interoperability (HAVi) é uma API que apresenta elementos de aplicação relacionados à interface com o usuário (HAVi, Inc., 2008). Essa API fornece uma extensão ao pacote *java.awt*, adicionando suporte a controle remoto de aplicações (em que funcionalidades de um dispositivo podem ser utilizadas e controladas a partir de um outro dispositivo), controle sobre transparência em várias camadas de elementos de tela sobrepostos, e outros itens de interface. Foi originalmente criada para fornecer um padrão de interoperabilidade entre dispositivos, e logo foi adotada pelos principais padrões de TV Digital, por disponibilizar funcionalidades específicas para TV, como componentes de botões, campos de textos, campos de entrada de textos, dentre outros, que facilitam o desenvolvimento das interfaces interativas das aplicações (FERNANDES; LEMOS; SILVEIRA, 2004).

4. DAVIC

   Digital Audio Visual Council (DAVIC) é uma API que provê acesso a funcionalidades específicas para o ambiente de TV Digital, permitindo controle sobre os recursos limitados do terminal de acesso, como componentes de *hardware*. O objetivo da API

é fornecer funcionalidades para manter a interoperabilidade fim-a-fim de informações audiovisuais e por difusão (FERNANDES; LEMOS; SILVEIRA, 2004), com especificação de formatos de conteúdos para diversos tipos de objetos e também controle de língua do áudio e legenda.

5. JavaDTV

   A especificação JavaDTV foi desenvolvida pela *Sun Microsystems* com o apoio do Fórum SBTVD para substituir as funcionalidades do *middleware* GEM. A JavaDTV possui a base de desenvolvimento do Java TV e JMF usada em *middlewares* internacionais e é a API padrão para o Ginga-J (Fórum SBTVD, 2009a). Suporta as principais APIs da *Sun Microsystems* para dispositivos de recursos limitados: *Connected Device Configuration* (CDC) 1.1, *Foundation Profile* (FP) 1.1 e *Personal Basis Profile* (PBP) 1.1. O JavaDTV ainda define parcialmente a especificação do *LightWeight User Interface Toolkit* (LWUIT) para interface com usuário. A API ainda define outras funcionalidades como: informações de serviço, sintonia e fluxo de transporte, tocadores multimídia, controle das mídias: áudio, vídeo e legendas, propriedades de perfil do usuário, persistência, canal de retorno, comunicação entre *Xlets* e segurança. Dessa forma, a API JavaDTV substitui funcionalmente os componentes do GEM, como DAVIC, HAVi e PersonalJava.

6. LWUIT

   LightWeight User Interface Toolkit (LWUIT) é a API que trata de elementos relacionados à interface de usuário, incorporada ao JavaDTV. Essa API permite o controle de componentes gráficos de alto nível, com o tratamento de eventos hierárquicos com o uso dos contêineres e componentes definidos na especificação. Permite componentes animados e personalização por meio de estilos.

7. Extensões brasileiras

   O sistema Ginga-J ainda possui extensões com o objetivo de complementar e estender o JavaDTV para melhor adaptação ao cenário brasileiro. Dessa forma, novas funcionalidades estão sendo previstas como: API de informações de serviço dependente de protocolo, acesso a informações dos serviços de baixo nível específicos do padrão brasileiro, API de extensões para sintonia de canais, API da ponte com linguagem NCL, API adicional de acesso ao canal de retorno, envio de mensagens, API de acesso de baixo nível aos planos gráficos e API de integração de dispositivos externos (Fórum SBTVD, 2009a).

3.3.3 Ciclo de Vida de um Programa de TVDI

O modelo de aplicações para TV Digital, de um modo geral, deve estar de acordo com o modelo de aplicações definido pelo Java TV. Assim, toda aplicação para TV Digital, seja interativa ou não, segue a especificação de uma interface denominada *Xlet*. Tanto o padrão GEM quanto o *middleware* brasileiro Ginga utilizam essa padronização definida pela API da *Sun Microsystems*.

Um *Xlet* é conceitualmente semelhante a um *applet*, que se trata de uma aplicação Java que pode ser carregada de uma aplicação Web e gerenciada por um navegador de Internet. Os *Xlets* se referem a aplicações Java que podem ser carregadas para os terminais de acesso do telespectador (por difusão ou pelo canal de retorno dependendo do *middleware*) e serem controladas pelo gerenciador de aplicações global presente nos *middlewares*. A interface *Xlet* é encontrada no pacote *javax.microedition.xlet* que é parte do Java ME PBP no ambiente Ginga-J (Fórum SBTVD, 2009a), ou no pacote *javax.tv.xlet* que é parte do Java TV (Sun Microsystems, 2009c).

Os *Xlets* podem ser carregados, ativados, interrompidos ou destruídos de acordo com o controle de estados do sistema ou da própria aplicação (FERNANDES; LEMOS; SILVEIRA, 2004). A interface Xlet provê o ciclo de vida para a aplicação por meio da implementação de seus métodos. Pelos métodos da interface, é possível controlar a transição dos estados da aplicação. Os métodos da interface a serem implementados são: *initXlet*, *startXlet*, *pauseXlet* e *destroyXlet* (BECKER et al., 2005). A Figura 18 ilustra as mudanças de estado possíveis em um *Xlet*.

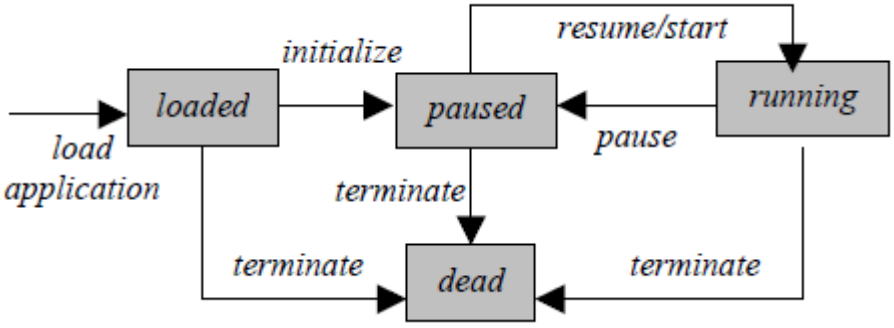


Figura 18: Ciclo de Vida de um *Xlet* (PENG; VUORIMAA, 2001)

O módulo que controla o ciclo de vida das aplicações dentro do *middleware* e informa a situação de cada aplicação para o restante do sistema é chamado Gerente de Aplicações. Há também o módulo Navegador, uma aplicação especial residente no receptor, que é responsável por mostrar a lista de aplicações disponíveis para os usuários e interagir com

o Gerente de Aplicações.

As informações sobre as aplicações carregadas são armazenadas em uma tabela de dados chamada *Application Information Table* (AIT). Essa tabela com informações sobre as aplicações é multiplexada e enviada ao terminal de acesso, juntamente com os outros elementos do fluxo de transporte MPEG-2 (áudio, vídeo e dados). Ao chegar ao receptor, o fluxo é desmembrado, e a tabela AIT é utilizada pelo Gerente de Aplicações para que identifique as informações sobre localização e estado inicial das aplicações (PENG; VUORIMAA, 2001) e possa executá-las da forma adequada.

As aplicações não podem ser executadas sem o Gerente de Aplicações. Dessa forma existe uma estratégia de comunicação entre a aplicação e o gerenciador para que consigam trafegar informação entre si. Essa comunicação é feita por um elemento chamado *XletContext*. O *XletContext* permite que o *Xlet* possa descobrir informações sobre o ambiente de execução em que está envolvido por suas propriedades. E o Gerenciador de Aplicações é capaz de controlar as trocas de estado do *Xlet* e terminá-lo por meio da interface *XletContext* passada à aplicação (PENG; VUORIMAA, 2001).

Além do ciclo de vida apresentado, é possível que haja aplicações residentes no terminal de acesso que podem ser implementadas usando funções não padronizadas pela especificação. Essas aplicações residentes podem utilizar API específica do sistema operacional ou de uma implementação proprietária do *middleware*. Essa funcionalidade está especificada no Ginga, como pode ser observado em Fórum SBTVD (2009a).

3.3.4 Desenvolvimento com Canal de Retorno

Para se conectar em algum provedor de serviços para interação, as aplicações de TV Digital podem utilizar a API *java.net* para estabelecer comunicação e intercambiar dados como em uma aplicação tradicional. Para o ambiente de TV Digital algumas diferenças são observadas. O canal de retorno pode não estar completamente disponível no momento da comunicação e os objetos da API *java.net* não conseguirão se comunicar por estarem preparados para funcionar somente com conexões permanentes. A instabilidade do canal de retorno pode ser explicada por existirem diversas opções de comunicação como WiMAX, Wi-Fi e ADSL (Seção 2.4). Isso torna necessário algum controle para manipular as conexões, desconexões e permissões sobre os dispositivos de rede. Com isso, as APIs dos *middlewares* dedicadas para comunicação com o canal de retorno devem estar preparadas para prevenir que as aplicações utilizem o canal de retorno sem permissão ou que utilizem desnecessariamente esse escasso recurso do terminal de acesso do telespectador.

O *middleware* europeu MHP possui a API do gerenciador de conexões do canal de retorno definida no pacote *org.dvb.net.rc*. A interface *RCInterface* representa o objeto relacionado ao canal de retorno, sem ainda o conceito de conexão. Essa interface estabelece conceitos básicos como tipo de dispositivo e sua taxa de transmissão. A interface *ConnectionRCInterface* é uma subclasse da interface anterior e representa um canal de retorno que não está permanentemente conectado. A *ConnectionRCInterface* possui métodos para manipular conexão e desconexão do canal de retorno (JONES, 2002). Para que a comunicação seja estabelecida, é necessário que se defina parâmetros de conexão do dispositivo de rede. A classe *ConnectionParameters* suporta parâmetros tais como número do telefone, usuário e senha para que a conexão seja obtida e autenticada. Alguns dispositivos não utilizam parametrização por usarem outros mecanismos de autenticação. Nesses casos, os parâmetros podem ser omitidos.

O acesso ao canal de retorno deve ser feito por intermédio da classe *RCInterfaceManager*. Essa classe possui os métodos necessários para se obter uma interface do canal de comunicação. Esse métodos podem receber como parâmetros os objetos *URLConnection*, *Socket* ou *InetAddress* do pacote *java.net*. A interface do canal de retorno deve ser reservada pela aplicação antes que conexão seja efetivamente requisitada. Esse procedimento evita que várias aplicações concorram pela conexão simultaneamente. A Figura 19 representa um trecho de código, adaptado de Jones (2002), com os passos para estabelecimento da comunicação com o canal de retorno pela API GEM/MHP.

O desenvolvedor deve implementar qualquer protocolo adicional que seja necessário para se comunicar com o meio externo, seja para um acesso a banco de dados ou para registrar dados em uma aplicação Web ou mesmo enviar um *e-mail*. Na verdade, os *middlewares* disponibilizam um ambiente flexível para prover o mecanismo de comunicação para um vasta gama de possíveis aplicações sem se preocupar com qual protocolo de troca de informações será estabelecido.

A API de desenvolvimento para o canal de retorno do *middleware* Ginga é baseada no *MHP*, com algumas alterações e complementações. Uma novidade na API Ginga é a possibilidade de envio de mensagens assíncronas através do canal de retorno do receptor. Essas mensagens podem ser enviadas após a conexão estabelecida ou em um horário pré-estabelecido (Fórum SBTVD, 2009a).

A especificação da API Ginga define o pacote *br.org.sbtvd.net.rc* como o caminho para suas classes. A interface *ReturnChannel* representa o objeto relacionado ao canal de retorno, sem ainda o conceito de conexão. O conceito de conexão é definido pela

```
import java.io.*;
import java.net.*;
import org.dvb.net.rc.*

public class RCInterfaceManagerExample implements ConnectionListener{

  public void stabilishConnection() {
    // recupera a instância do gerenciador do canal de retonor
    RCInterfaceManager rcManager = RCInterfaceManager.getInstance();
    // cria um objeto com o endereço do canal de retorno
    InetAddress inet = InetAddress.getByName("jitv.pucminas.br");
    // recupera uma interface para comunicar com o servidor
    RCInterface rcInterface = rcManager.getInterface(inet);
    // verifica se é uma interface conectável
    if(rcInterface instanceof ConnectionRCInterface){
      ConnectionRCInterface connection =
                                    (ConnectionRCInterface)rcInterface;
      // verifica se está conectado
      if(connection.isConnected()){
        // sim - proceder com o uso da API java.net
      } else {
        // não - fazer conexão
        // parâmetros para conexão
        ConnectionParameters cp = new ConnectionParameters
                                      ("123456789", "user", "password");
        connection.setTarget(cp);
        connection.connect();
      }
    } else {
      // Não é uma interface conectável - assumir que está conectado
    }
  }
}
```

Figura 19: Exemplo de comunicação com o canal de retorno com API GEM/MHP.

classe *ConnectionReturnChannel* e pelo controlador *ConnectionRCController* que possui os métodos para conectar, desconectar, reservar e monitorar a situação da conexão. A classe *ReturnChannelManager* representa o produtor e gerenciador das interfaces de canal de retorno fornecendo acesso aos dispositivos por meio dos objetos do pacote *java.net* semelhante ao MHP.

Adicionalmente ao MHP, a API Ginga define a possibilidade de envio de mensagens assíncronas pelo canal de retorno. Essas mensagens são representadas pela classe *RCMessage* que recebe um objeto serializável e um caminho de localização para o servidor remoto. É com essa classe que a mensagem é enviada. Uma outra classe denominada *RCAsynchronous* possui recursos mais sofisticados para estipular uma data de envio da mensagem e a data de expiração para descarte da mensagem. A classe *AsynchronousMessageTable* define um objeto que contém a lista de mensagens a serem enviadas assincro-

namente. As mensagens assíncronas *RCAsynchronous* são adicionadas à lista para serem enviadas quando determinado.

3.4 Trabalhos Relacionados

Alguns autores abordam a nova forma de desenvolvimento de aplicações para o receptor de acesso, com o intuito de descrever o processo de desenvolvimento, mas sem se preocupar com o tipo de aplicação que resultará do processo. Como exemplo, cita-se Jones (2002), que apresenta os passos detalhados para a criação e publicação de sistemas, dando ênfase para as fases de desenvolvimento e abordando minucias do sistema (ciclo de vida, controle de acesso de arquivos e canal de comunicação externo). Sua metodologia envolve o desenvolvimento de uma aplicação e o uso de emuladores para testes.

As aplicações iniciais desenvolvidas para o ambiente de TV Digital apresentam uma tendência de trazer para o contexto da TV as mesmas características encontradas no ambiente de Internet. Ferretti, Roccetti e Andrich (2006) propõem a simplificação das páginas em linguagem HTML para uma linguagem chamada XHTML *Basic* para possibilitar a visualização no televisor, seguindo a linha evolutiva de Gil et al. (2002), que descreve as funcionalidades existentes no *middleware* MHP referentes à visualização de HTML. Segundo Gil et al. (2002), o DVB-HTML traz bastante flexibilidade do HTML tradicional para o ambiente do terminal receptor, apesar dos ajustes necessários para a adaptação da linguagem para esses novos dispositivos. Outros autores (MESQUITA; SOUZA, 2006) tratam o uso de ferramenta de *e-mail* no ambiente de TV Digital. Mesquita e Souza (2006) criaram um protótipo de cliente de *e-mail* para funcionamento no receptor com a comunicação com um servidor de *e-mail* específico. Esse servidor de *e-mail* converte as mensagens no formato de Internet para um padrão criado para comunicação com o terminal de acesso.

Como um dos principais objetivos da TV Digital no Brasil é a inclusão digital (MARGALHO; FRANCêS; COSTA, 2007), uma tendência bastante relevante é o crescimento dos estudos em aplicações relacionadas à educação à distância no contexto da TV Digital. Vários autores abordam esse tema e propõem aplicações para o terminal de acesso com o objetivo de ensino pelos canais de televisão. As aplicações de ensino à distância são chamadas de *t-learning*, ao contrário do *e-learning* para as aplicações de ensino utilizadas na Internet. Um comparativo entre *t-learning* e *e-learning* é mostrado em (SANTOS; VALE; MELONI, 2006), salientando as diferenças entre os ambiente de TV e Internet, e detalhando as dificuldades e impedimentos encontrados no desenvolvimento de aplica-

tivos para o receptor de TV. Esse trabalho mostra também uma aplicação para estudo e testes do telespectador. O trabalho de Andreata (2006) mostra similaridades com esse anterior. Andreata aborda as viabilidades de implementação de aplicações de *t-learning* e as dificuldades encontradas no decorrer do desenvolvimento do protótipo apresentado por ele.

Seguindo a tendência de ensino à distância pelo televisor, Fuks et al. (2007) mostra a adaptação que está sendo construída no sistema chamado AulaNet para compatibilização com o ambiente de TV Digital. O sistema AulaNet é um ambiente gratuito para ensino-aprendizagem *online* pela Internet. E a proposta dos autores é que a camada de apresentação do sistema AulaNet seja funcional também no ambiente do receptor de TV Digital.

Os trabalhos relacionados anteriormente nesse texto mostram aplicações verticalizadas, isto é, específicas para um ramo de conhecimento. Podemos notar linhas de conhecimento claras como aplicações de Internet (HTML e *e-mail*) e *t-learning*. Mais recentemente, vem surgindo outros tipos de aplicações para TV Digital num âmbito mais genérico. Batista et al. (2007) propõem a utilização do poder de processamento dos receptores de TV como terminais processadores de tarefas paralelas. Pelo canal de comunicação ocioso, aplicações são transmitidas para serem executadas no receptor quando também estiver desocupado, formando assim uma rede de receptores trabalhando em tarefas complexas e sendo coordenados por um servidor remoto. Em outro trabalho de uma aplicação verticalizada é mostrado um *framework* para desenvolvimento de aplicações para TV Digital (VRBA; CVRK; SYKORA, 2006). Esse *framework* teria o objetivo de abstrair alguns pontos importantes da fase desenvolvimento, permitindo ao desenvolvedor construir aplicações complexas sem ter conhecimento profundo dos detalhes de implementação. A implementação do sistema seria feita com uso de XMLs que representariam funções importantes como validações de dados, segurança, suporte a multi-línguas, autorização e outros.

3.5 Considerações Finais

O desenvolvimento de aplicações interativas para TV Digital passou por algumas evoluções desde seu início. As primeiras aplicações estavam voltadas para apresentação de informações sobre a programação da TV. Paulatinamente a possibilidade de interação com o telespectador aumentou. No início, observou-se a interação local, em que o usuário não conseguia enviar dados para o ambiente externo. E evolutivamente, o telespectador

passou a participar ativamente das programação, enviando informações em tempo real para servidores remotos.

Neste capítulo, apresentou-se um detalhamento sobre os diversos *middlewares* para TV Digital no mundo e como cada um deles possibilita o desenvolvimento de aplicações. Duas tendências mundias são observadas quanto à escolha de tecnologias para os *middlewares*. Uma voltada para o GEM, em que se encontram o padrões DVB, ATSC e ISDB. E outra voltada para o JavaDTV, incluindo o Brasil, com o ISDTV-B. O próximo capítulo mostrará em detalhes o Projeto JiTVPSI-CommService. Esse projeto está integrado à plataforma JiTV e tem a proposta de desenvolver uma camada de software relacionada à comunicação de dados de aplicações através do canal de retorno.

4 SISTEMA JITVPSI-COMMSERVICE

O sistema desenvolvido como resultado da proposta apresentada nesta dissertação aborda problemas teóricos e, principalmente, busca a validação desses aspectos pelas implementações de sistemas computacionais. Implementou-se um conjunto de produtos e serviços para suportar o desenvolvimento de programas para Televisão Digital Interativa, contemplando os aspectos de comunicação de aplicações desenvolvidas em um *middleware* genérico. A camada de software desenvolvida tem a capacidade de se comunicar com um servidor externo para prover os mais diversos serviços de interação. Aspectos inovadores foram analisados durante o desenvolvimento, sobretudo em relação à criação de uma estratégia de comunicação e sua integração com o provedor de serviços remoto.

As estratégias de implementação do sistema exploram os requisitos exigidos pelo SBTVD-T e as especificações de *middleware* propostas pelo Ginga (SOUZA; LEITE; BATISTA, 2007) (SOARES; RODRIGUES; MORENO, 2007), tendo em vista também as especificações do *middleware* GEM para comunicação com canal de retorno.

Como resultado do trabalho realizado, criou-se um módulo de sistema denominado JiTVPSI-CommService. O JiTVPSI-CommService, que está integrado à plataforma JiTV, é um sistema que visa explorar os padrões envolvidos na TV Digital Brasileira, implementando uma camada de software relacionada à comunicação de dados de aplicações pelo canal de retorno, para prover serviços de interação. A Figura 20 mostra, em destaque, as partes do ambiente de TVDI que estão cobertas pelo sistema.

A intenção de um sistema exclusivo para comunicação de dados é evitar que cada aplicação tenha que implementar sua própria forma de negociação com o Provedor de Serviços Interativos. JiTVPSI-CommService possui a capacidade de se comunicar com um servidor externo para prover os mais diversos serviços de interação.

Neste capítulo são apresentadas as tecnologias e questões técnicas envolvidas na implementação do JiTVPSI-CommService. A arquitetura do sistema será descrita em detalhes. E em seguida, cada um dos seus módulos será também detalhado, apresentando

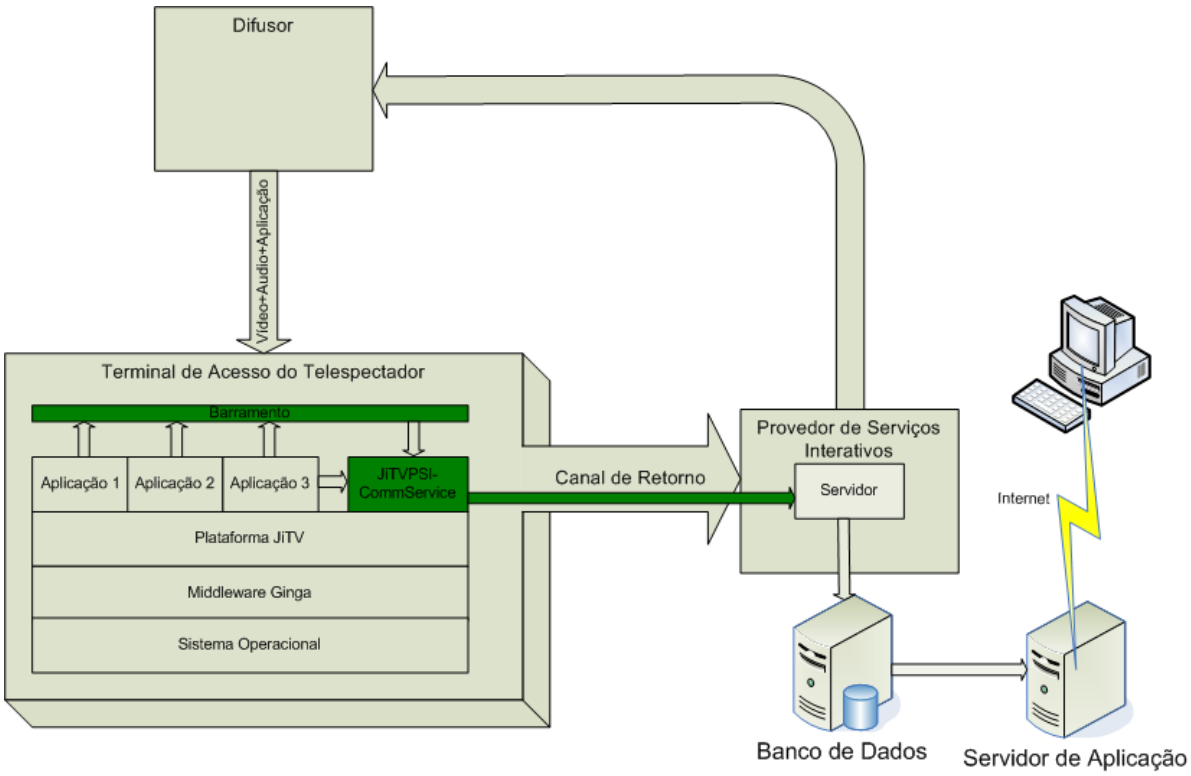


Figura 20: Diagrama que ilustra em destaque os objetivos do projeto

as ferramentas e padrões utilizados.

4.1 Arquitetura do Sistema

A arquitetura do projeto JiTVPSI-CommService foi elaborada de forma modular. Essa arquitetura está ilustrada no Diagrama de Componentes representado na Figura 21. Entre os principais módulos que fazem parte dessa arquitetura estão: Monitor de Barramento, Monitor de Lista de Envio, Monitor de Banco de Dados e a API de Interatividade.

JiTVPSI-CommService foi desenvolvido de forma genérica com o objetivo de se adequar a qualquer sistema de televisão digital. Assim, é possível estendê-lo a *frameworks* específicos, como é o caso de JavaDTV e do Ginga-J. O módulo API de Interatividade foi implementado baseado na especificação da API de canal de interatividade do Ginga-J (no pacote *br.org.sbtvd.net.rc*) conforme descrito em Fórum SBTVD (2009a). O Diagrama de Classes completo do sistema encontra-se no Apêndice A.

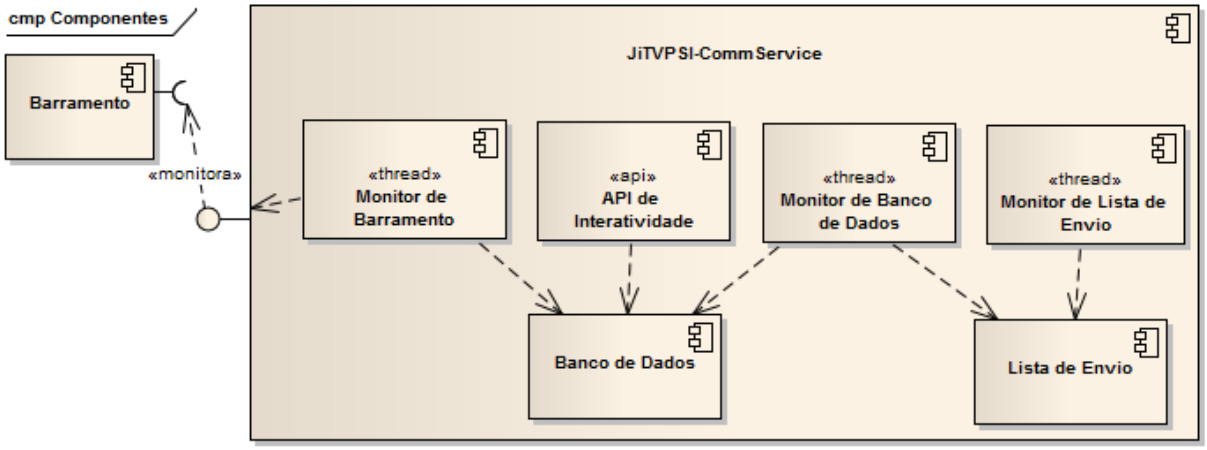


Figura 21: Diagrama de Componentes do sistema JiTVPSI-CommService

4.1.1 Módulo Monitor do Barramento de Comunicação

O barramento de comunicação é a denominação da estrutura lógica que permite a interligação ou troca de informações entre as aplicações de terceiros e o sistema JiTVPSI-CommService. O principal objetivo do módulo Monitor de Barramento de Comunicação consiste em controlar um barramento de comunicação em busca de arquivos no formato XML que representem mensagens a serem enviadas pelo canal de retorno. O barramento é materializado como um diretório específico no sistema operacional do terminal de acesso. As mensagens são coletadas do barramento, tratadas por um analisador estrutural, devidamente representadas de acordo com o serviço de comunicação e, finalmente, são armazenadas para serem enviadas quando possível. A Figura 22 representa o Diagrama de Sequência do processo de monitoração do barramento. Esse monitor executa a cada 1 segundo, dessa forma, uma mensagem adicionada ao barramento levará no máximo esse tempo para ser capturada pelo sistema e iniciar o seu processo de envio.

A primeira tarefa do monitor de barramento é capturar todos os arquivos no formato XML que estão presentes em um determinado diretório do sistema operacional do receptor. Cada um dos arquivos de mensagens devem estar em formato específico para que a aplicação JiTVPSI-CommService consiga interpretá-los e entregar a mensagem corretamente ao seu destino. O arquivo apresenta duas partes principais com os marcadores de transporte e de mensagem. A primeira parte, representada pelo marcador *transport*, corresponde à configuração necessária para envio da mensagem, como endereço de servidor, parâmetros, usuário e senha de acesso. E a segunda parte, representada pelo marcador *message*, é o conteúdo da mensagem a ser enviada. Os tipos de transportes permitidos

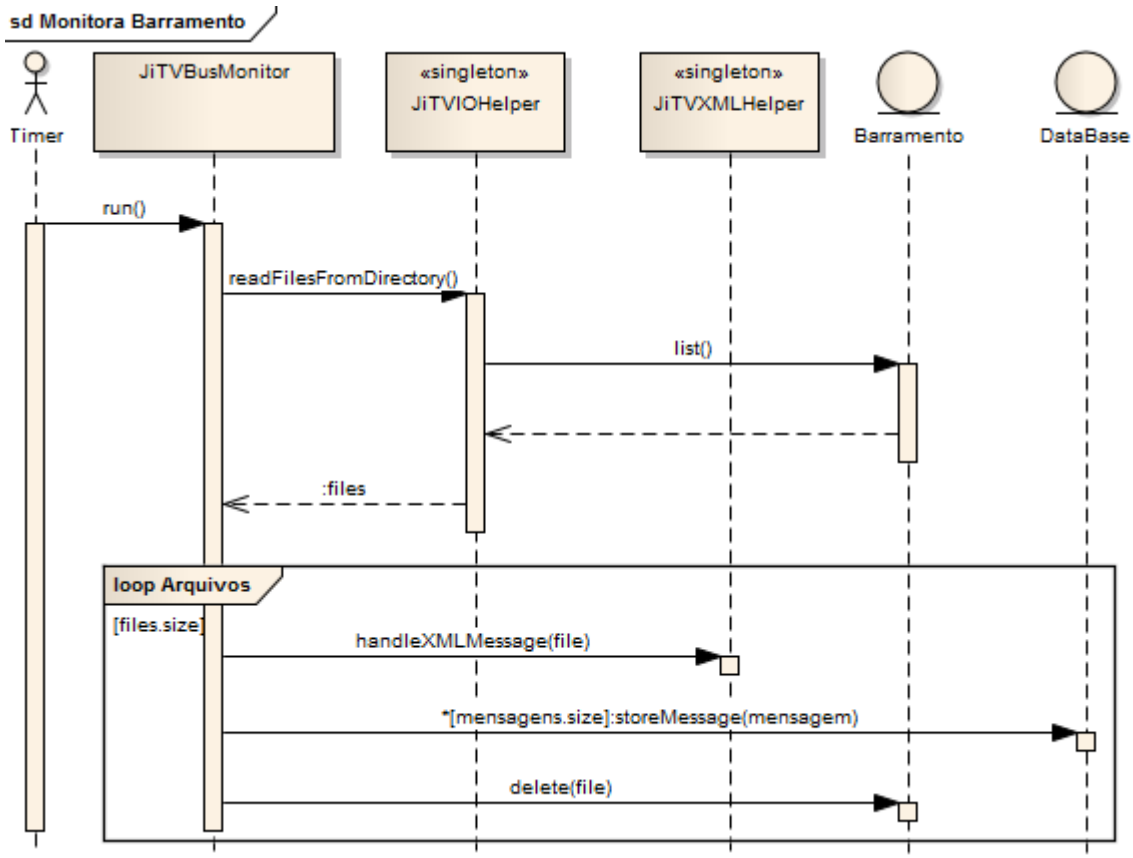


Figura 22: Diagrama de Sequência de processo que monitora o barramento

pelo sistema são: *e-mail*, *socket*, HTTP (postagem em formulário), *Web-Service* e *File Transfer Protocol* (FTP). Cada um desses tipos será detalhado na Seção 4.2. A Figura 23 mostra um exemplo do arquivo de mensagem.

As mensagens são devidamente tratadas pelo manipulador de arquivos XML, representado pela classe *JiTVXMLHelper*. A classe *JiTVXMLHelper* é um *singleton* que analisa os arquivos XML, transformando-os em objetos do tipo *JiTVMessage* de acordo com os marcadores de transporte em cada arquivo. Os objetos de mensagens possuem uma hierarquia de classes correspondente ao serviço de comunicação permitido pelo sistema. A Figura 24 representa o Diagrama de Classes dessa estrutura. As APIs utilizadas para tratamento de arquivos e tratamento do formato XML pertencem à máquina virtual Java 1.4.2 e não foi necessário adicionar qualquer biblioteca auxiliar para tratar essas informações.

Cada uma dessas mensagens coletadas do barramento de comunicação é armazenada no banco de dados da aplicação para ser enviada pelo módulo que monitora a lista de envio. A estratégia de armazenamento no banco de dados e a estrutura dos dados persistidos serão discutidos na Seção 4.1.2.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<commservice>
  <transport>
   <email>
     <host>localhost</host>
     <port>25</port>
     <to>rogerio.baldini@gmail.com</to>
     <from>rogerio.baldini@gmail.com</from>
     <subject>Quiz Dengue</subject>
   </email>
  </transport>
  <message><![CDATA[<?xml version="1.0" encoding="iso-8859-1"?>
<Quiz type="Dengue">
  <etiqueta>
    <idemissora>glb</idemissora>
    <idinteracao>001</idinteracao>
    <ip>192.168.0.100</ip>
    <data>24/06/2009</data>
    <hora>21:54:20</hora>
    <pergunta1>Sim</pergunta1>
    <pergunta2>Sim</pergunta2>
    <pergunta3>Sim</pergunta3>
    <pergunta4>Sim</pergunta4>
    <observacao>
    Há um grande risco de possuir dengue em sua residência !!!
    </observacao>
  </etiqueta>
</Quiz>
  ]]>
  </message>
</commservice>
```

Figura 23: Exemplo de XML para mensagem a ser transmitida.

4.1.2 Módulo Monitor do Banco de Dados

O monitor do banco de dados é representado pelo módulo JiTVDataBaseMonitor. Esse módulo é responsável por recuperar as mensagens que foram armazenadas em banco de dados e recolocá-las na lista para envio para que o procedimento de comunicação seja realizado em cada uma delas.

As mensagens ficam armazenadas em uma estrutura de banco que contém somente uma tabela. A tabela tem o nome de JITV_MESSAGE e está modelada de acordo com a Figura 25. A chave primária da tabela é o atributo chamado *index* e é preenchido por incremento automático pelo próprio gerenciador do banco de dados. A tabela possui outros quatro atributos do tipo inteiro longo para armazenar valores de tempo. Esses tempos são representados pela quantidade de segundos passados desde janeiro de 1970. O atributo *timestamp* representa o momento em que a mensagem foi criada, o atributo

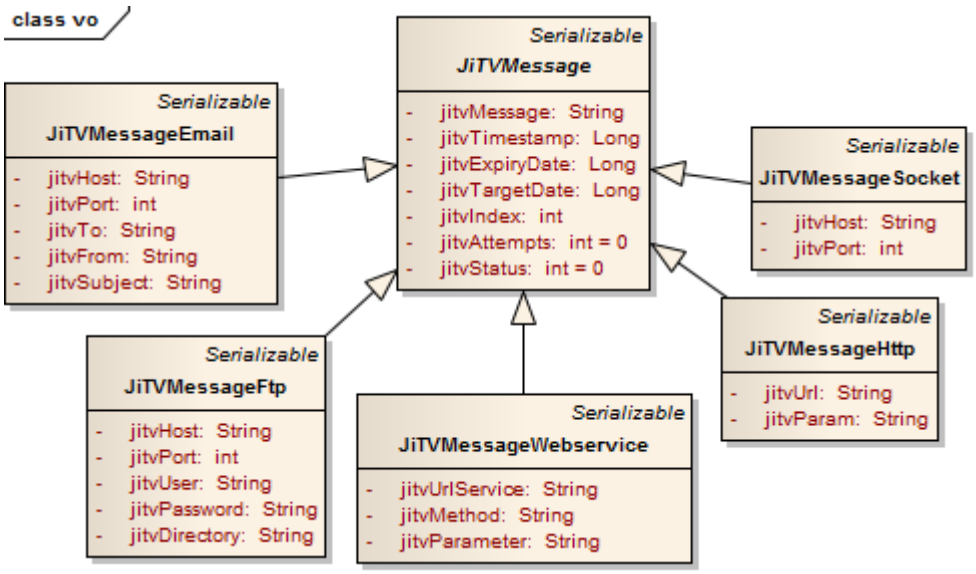


Figura 24: Diagrama de Classes da hierarquia JiTVMessage

targetDate representa o momento em que a mensagem deverá ser enviada, o atributo *expiryDate* representa o momento em que a mensagem deverá ser expirada (eliminada) e o atributo *nextAttempt* representa quando poderá ser tentado um novo envio.

O atributo de tempo para a próxima tentativa foi adicionado para que a mesma mensagem não tente ser enviada novamente num período muito curto de tempo desde a última tentativa. Foi levado em consideração o fato que, se a mensagem não pode ser enviada nesse momento, a probabilidade de conseguir sucesso nos próximos dez ou quinze segundos será mínima, daí é adicionado um tempo de espera de dez minutos com o objetivo de aumentar essa possibilidade. É armazenado também o número de tentativas de envio já feitas para cada mensagem. Esse número é limitado em cinco tentativas para não permitir que uma mesma mensagem tenha vários insucessos de envio e fique consumindo recurso

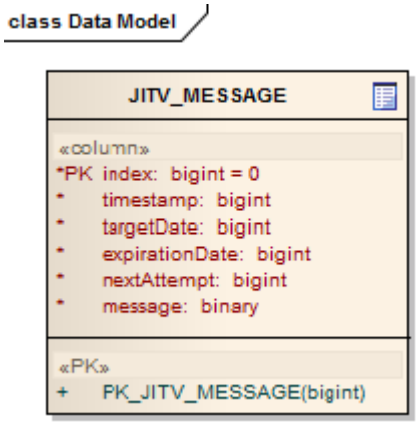


Figura 25: Tabela do banco de dados para armazenar as mensagens

do terminal de acesso. O tempo de espera para reenvio das mensagens foi determinado de forma empírica, apenas para não reenviar a mensagem num momento imediatamente posterior. Nenhuma análise mais detalhada foi realizada sobre esse tempo.

A mensagem propriamente dita é armazenada no banco de dados no atributo *message*. Esse atributo tem formato binário para facilitar a conversão do objeto Java em um fluxo de dados para ser persistido e vice-versa. Dessa forma, no momento de inserção dos dados na tabela do banco, o objeto Java é serializado e armazenado no atributo binário, e no momento de recuperação, o atributo binário é revertido ao mesmo objeto Java. Essa escolha foi tomada somente por facilidade de implementação e diminuição de linhas de código do sistema.

Assim, as mensagens, para serem selecionadas pelo monitor de banco de dados, devem apresentar as seguintes condições:

- terem alcançado o tempo de envio;
- não estarem expiradas;
- estarem dentro do tempo para a próxima tentativa;
- e terem o número de tentativas menor que cinco.

Em cada execução do monitor de banco de dados, também é feita a exclusão de mensagens expiradas, eliminando todas que estejam com o atributo de expiração com o tempo alcançado.

As bibliotecas para criação da estrutura do banco de dados e para sua manipulação estão empacotadas na aplicação JiTVPSI-CommService. Ao iniciar o sistema pela primeira vez, é feita uma verificação para se certificar que o gerenciador do banco de dados foi iniciado com sucesso e que a tabela foi criada e está funcional. Caso contrário, são executados os *scripts* para criação da tabela de acordo com o modelo de dados definido no sistema.

O banco de dados utilizado para armazenamento das informações é o Hyper SQL Database (HSQLDB) (HSQLDB, 2009). O HSQLDB é um banco de código aberto, escrito totalmente em linguagem Java. Esse banco é uma solução simples e de tamanho pequeno, o que é uma vantagem pelo projeto se tratar de um sistema para ser executado em dispositivos com recursos restritos. Em relação ao banco de dados, a escolha foi baseada em fatores funcionais e pelo tamanho do arquivo físico da biblioteca. Era requerido um

sistema de banco de dados que suportasse um máquina de gerenciamento de dados com interpretação de comandos SQL para facilitar a inserção e recuperação de dados, além de ser em Java para ficar embutido com o restante da aplicação. Outro fator relevante era o tamanho do arquivo da biblioteca com o banco de dados, pois era imprescindível um arquivo pequeno para que o tamanho total do sistema também fosse pequeno. Outros gerenciadores de bancos de dados, com comandos SQL, em Java e de código livre foram analisados, como o Apache Derby e o Oracle Berkeley DB Java Edition. O HSQLDB foi escolhido por ser o de menor tamanho - 691 KB, contra 2,4MB do Apache Derby e 2,1MB do Oracle Berkeley DB Java Edition - já que, funcionalmente, todos eles apresentam características semelhantes.

4.1.3 Módulo Monitor da Lista de Envio

O monitor da lista de envio é representado pelo módulo JiTVSendListMonitor. Esse módulo monitora a lista de mensagens a serem enviadas. O sistema tenta enviar cada uma das mensagens da lista pelo canal de retorno de acordo com os parâmetros em cada uma delas. As mensagens que se encontram na lista são mensagens novas ou mensagens cujas tentativas de envio anteriores falharam. Após a tentativa de envio de cada uma das mensagens, obtendo sucesso ou não, a situação das mensagens é atualizada no banco de dados. As mensagens que não consigam ser enviadas, seja por falha no canal de retorno, por erro de parametrização da mensagem ou falta de comunicação do serviço remoto, terão o seu número de tentativas incrementado, a situação colocada com o resultado de insucesso e o tempo para a próxima tentativa adicionado de dez minutos. As mensagens com sucesso também terão sua situação atualizada. A Figura 26 representa o Diagrama de Sequência do processo de monitoração da lista de mensagens.

Para que seja possível o envio de todas as mensagens para os seus respectivos destinos, é necessário que haja uma conexão estabelecida por algum dispositivo de rede que permita acesso ao meio externo. A classe *JiTVConnectionRC* é responsável por estabelecer a conexão com uma interface de rede. No seu método *establishConnection*, essa classe utiliza a API de interatividade e tenta conectar o receptor a um dispositivo de rede disponível. Como os métodos *connect* e *disconnect* da classe *ConnectionRCController* são assíncronos, o monitor da lista de envio deve aguardar até que a conexão retorne com sucesso ou seja abortada por *timeout*. Com a conexão feita, as mensagens podem então ser enviadas. A API de interatividade será detalhada na Seção 4.1.4.

Como cada mensagem pode utilizar um serviço de comunicação para acessar ao meio

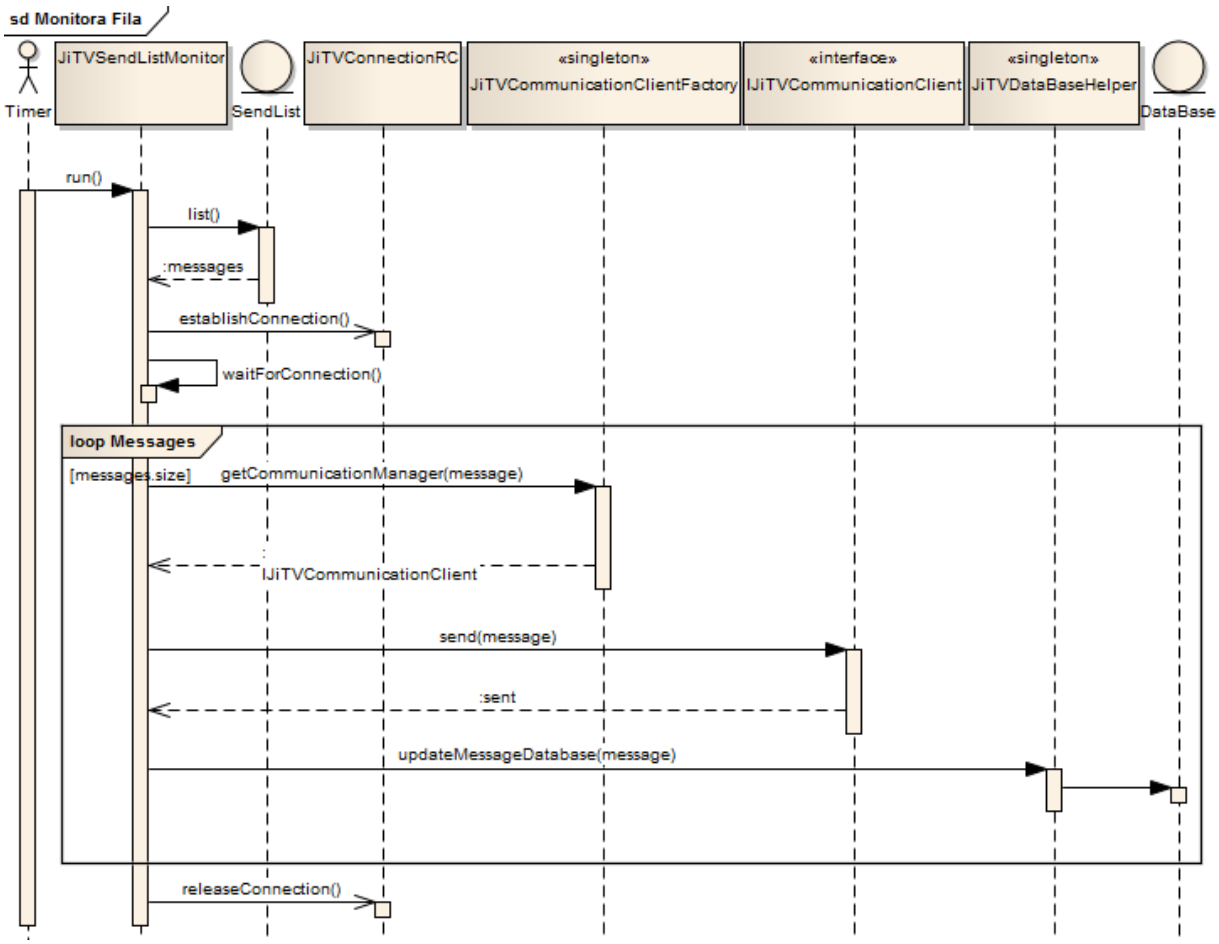


Figura 26: Diagrama de Sequência de processo que monitora lista de envio

externo pelo canal de retorno, foi implementada uma fábrica produtora de objetos de comunicação, classe *JiTVCommunicationClientFactory*, de acordo com a configuração de cada uma das mensagens. Essa fábrica analisa o tipo da mensagem e retorna a instância de uma classe da interface *IJiTVCommunicationClient*. As subclasses da interface *IJiTV-CommunicationClient* são responsáveis por concretizar a comunicação de dados no seu determinado serviço. A estrutura das classes responsáveis pela comunicação está ilustrada no Diagrama de Classes da Figura 27. A fábrica faz uso de uma classe *singleton* denominada *JiTVClientLocator*, que mantém todos as classes responsáveis pela comunicação em memória para não necessitar de uma nova busca para cada chamada na fábrica.

Assim, para cada mensagem, é feita uma chamada à fábrica de objetos de comunicação. Essa fábrica retorna o responsável pela comunicação do serviço, e, enfim, o sistema aciona o envio da mensagem. De acordo com o resultado do envio, as informações do banco de dados são atualizadas e a conexão com o dispositivo de rede é liberada.

Cada um dos três monitores do sistema JiTVPSI-CommService são *threads* (proces-

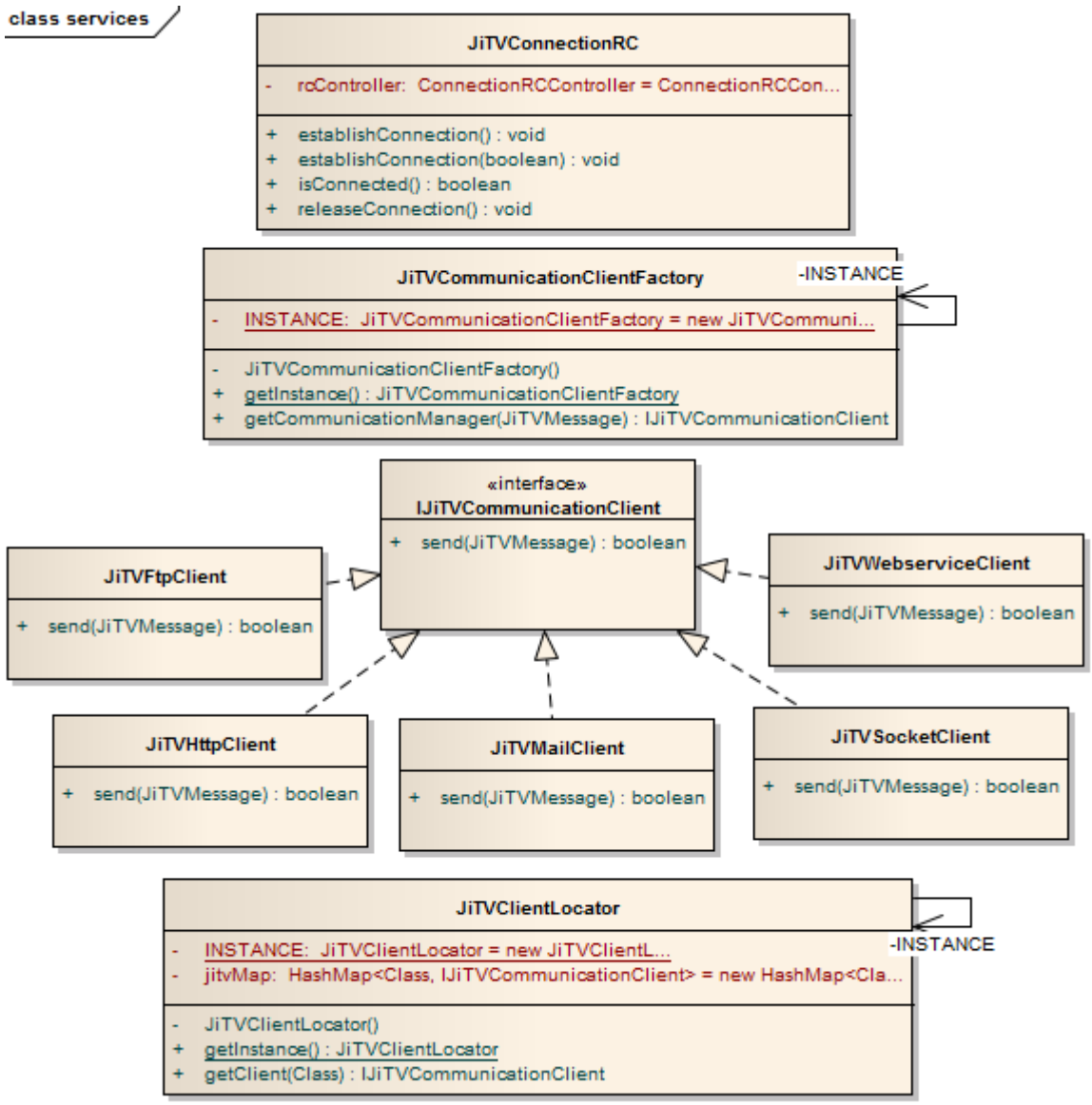


Figura 27: Diagrama de Classes do pacote de serviços de comunicação

sos) independentes que executam de forma recorrente de acordo com um tempo pré-determinado que está configurado para 1 segundo no caso dos monitores de barramento e de banco de dados, e 5 segundos para o monitor de lista de envio. A justificativa para a escolha dos tempos está embasada na necessidade de agilidade em cada um dos monitores. O monitor de barramento precisa ser executado em ciclos curtos para que as mensagens sejam retiradas o mais rápido possível do diretório e encaminhadas para envio. O monitor de banco de dados também deve ser executado em ciclos curtos para recuperar as mensagens do banco assim que estiver na hora de serem enviadas. Valores menores que 1 segundo poderiam causar execuções intensivas e levar a um nível alto de processamento desnecessariamente. Para o monitor da lista de envio foi considerado 5 segundos por causa do tempo necessário para criação da conexão com a interface de rede. Esse valores

deverão ser revistos em trabalhos futuros quando os testes forem feitos em ambientes de conexão reais, como está sendo proposto na Seção 6.1.1.

Para que não haja problema de concorrência entre as diversas *threads* sobre a lista de mensagens, um sistema de sincronismo foi implementado na classe *JiTVHandleConcurrence* para controlar os acessos à lista. Esse sistema possui dispositivos de travamento e destravamento para leitura e gravação na lista. Somente um processo de cada vez poderá fazer acesso de leitura ou gravação à lista.

4.1.4 Módulo API de Interatividade

O módulo API de Interatividade é a implementação da especificação da API canal de interatividade proposta pelo Ginga em seu documento Fórum SBTVD (2009a). Com esse módulo, é possível a manipulação dos dispositivos de interatividade disponíveis na plataforma, com comandos para reserva, conexão e desconexão das interfaces de rede. E esse módulo permite também o envio de objetos serializáveis a qualquer endereço remoto por meio de classes definidas para encapsulamento de mensagens e envio assíncrono.

A API de interatividade foi implementada seguindo rigorosamente o padrão definido pela proposta do Ginga-J que se adere ao JavaDTV Fórum SBTVD (2009a). Todas as classes e interfaces do padrão JavaDTV estão implementadas no sistema JiTVPSI-CommService conforme demonstra a Tabela 4, que ilustra um comparativo de classes e interfaces entre a API de Interatividade do Ginga-J (JavaDTV) e do sistema de comunicação JiTVPSI-CommService. O padrão JavaDTV foi escolhido como referência do sistema JiTVPSI-CommService por ser a base de implementação recentemente adotada pelo Sistema Brasileiro de TV Digital representada pelo módulo Ginga-J, e pelo fato de ser ainda uma especificação, não havendo implementação de referência.

A API de interatividade tem dois objetivos distintos. O primeiro deles é a manipulação dos dispositivos de rede do receptor. A Figura 28 ilustra as principais classes da API para manipulação dos dispositivos de rede. Como foram baseadas na API de canal de interatividade do Ginga, essas classes foram descritas na Seção 3.3.4. A implementação das classes *ReturnChannelManager*, que lista as interfaces de rede, e *ConnectionRCController*, que estabelece a conexão com uma interface, foi feita de forma simulada por não ser possível ter acesso real aos dispositivos de rede no ambiente de testes. Algumas classes que não estão representadas no diagrama na figura se relacionam ao disparo de eventos (*ConnectionRCEvent*, *ConnectionEstablishedEvent*, *ConnectionFailedEvent* e *ConnectionTerminatedEvent*) e no controle de exceções (*ReturnChannelException*, *NoFreeInterface-*

Tabela 4: Comparação com API de Interatividade do Ginga-J (JavaDTV)

Classes Ginga-J (JavaDTV)	Classes JiTVPSI-CommService
br.org.sbtvd.net.rc.ReturnChannel	br.pucminas.jitv.net.rc.ReturnChannel
br.org.sbtvd.net.rc.ReturnChannelManager	br.pucminas.jitv.net.rc.ReturnChannelManager
br.org.sbtvd.net.rc.ConnectionReturnChannel	br.pucminas.jitv.net.rc.ConnectionReturnChannel
br.org.sbtvd.net.rc.ConnectionRCController	br.pucminas.jitv.net.rc.ConnectionRCController
br.org.sbtvd.net.rc.ConnectionParameters	br.pucminas.jitv.net.rc.ConnectionParameters
br.org.sbtvd.net.rc.ConnectionListener	br.pucminas.jitv.net.rc.ConnectionListener
br.org.sbtvd.net.rc.ConectionRCEvent	br.pucminas.jitv.net.rc.ConectionRCEvent
br.org.sbtvd.net.rc.ConnectionEstablishedEvent	br.pucminas.jitv.net.rc.ConnectionEstablishedEvent
br.org.sbtvd.net.rc.ConnectionFailedEvent	br.pucminas.jitv.net.rc.ConnectionFailedEvent
br.org.sbtvd.net.rc.ConnectionTerminatedEvent	br.pucminas.jitv.net.rc.ConnectionTerminatedEvent
br.org.sbtvd.net.rc.ReturnChannelException	br.pucminas.jitv.net.rc.ReturnChannelException
br.org.sbtvd.net.rc.IncompleteTargetException	br.pucminas.jitv.net.rc.IncompleteTargetException
br.org.sbtvd.net.rc.NoFreeInterfaceException	br.pucminas.jitv.net.rc.NoFreeInterfaceException
br.org.sbtvd.net.rc.NotOwnerException	br.pucminas.jitv.net.rc.NotOwnerException
br.org.sbtvd.net.rc.RCMessage	br.pucminas.jitv.net.rc.RCMessage
br.org.sbtvd.net.rc.RCAsynchronous	br.pucminas.jitv.net.rc.RCAsynchronous
br.org.sbtvd.net.rc.AsynchronousMessageTable	br.pucminas.jitv.net.rc.AsynchronousMessageTable

Exception, *IncompleteTargetException* e *ReturnChannelException*). As classes de manipulação dos dispositivos de rede foram implementadas utilizando o conceito de *mocks* que simulam o comportamento de objetos reais de forma controlada. Essa implementação foi feita de forma simulada pelo fato de o Ginga-J ainda não ter uma implementação de referência completa.

E o segundo objetivo da API de interatividade corresponde ao envio de mensagens síncronas e assíncronas para o meio externo através do canal de retorno. A classe *RCMessage*, para envio de mensagens síncronas, é responsável por encapsular um objeto serializável, que representa a mensagem a ser enviada, e um localizador, que contém o serviço de envio e endereço de destino da mensagem. O construtor da classe *RCMessage* recebe como parâmetros a mensagem a ser enviada como um objeto *Serializable* do Java e um objeto *java.net.URL* que é o localizador. A implementação dessa classe, no sistema JiTVPSI-CommService, permite o recebimento do localizador com os seguintes serviços:

- email://host:porta/to/from/subject
- socket://host:porta

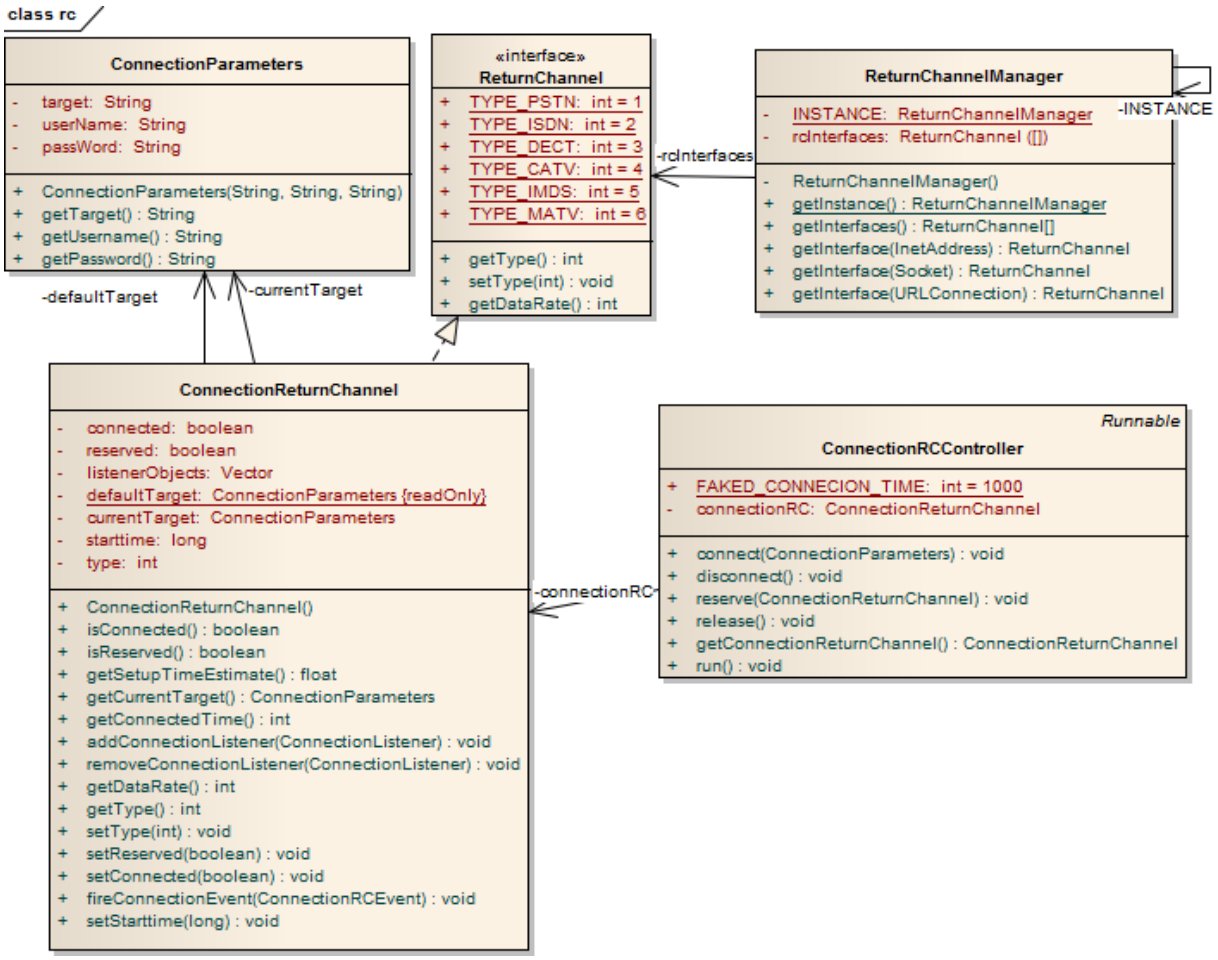


Figura 28: Classes da API de Interatividade para manipulação dos dispositivos de rede

- http://url#param
- webservice://urlService#method.parameter
- ftp://userid:password@host:porta/directory

Dessa forma, caso se deseje enviar uma mensagem por *e-mail*, basta instanciar o objeto *RCMessage* passando como parâmetros a mensagem e uma URL, por exemplo, com email://smtp.pucminas.br:25/destino@pucminas.br/origem@pucminas.br/Assunto. E de forma análoga é possível enviar mensagens pelos outros serviços, de acordo com cada padrão de localizador. Caso um objeto URL com um serviço não implementado seja passado como parâmetro, é retornada uma exceção. Cada um dos serviços será detalhado na Seção 4.2.

Na implementação da classe *RCMessage*, o construtor converte os parâmetros recebidos em uma classe do tipo *JiTVMessage* e o método *sendMessage* utiliza a classe *JiTV-*

CommunicationClientFactory para recuperar o cliente de comunicação e efetivamente enviar a mensagem ao seu destino. O processo de envio da mensagem também manipula os controladores de conexão para estabelecer a comunicação com o dispositivo de rede e liberá-la ao final do envio. Caso a mensagem não seja enviada, uma exceção é retornada. Essa mensagem não será armazenada no banco de dados para novas tentativas de envio, pois essa mensagens são síncronas e não se justifica seu envio posterior.

Para encapsulamento de mensagens assíncronas, a classe *RCAsynchronous* é definida. Com essa classe, é possível a configuração para o envio dos dados em uma data específica ou quando a conexão com o canal de retorno é estabelecida. O seu construtor recebe como parâmetros a data para envio da mensagem, a data para expiração da mensagem, e a mensagem pelo objeto do tipo *RCMessage*. Se a data de envio for nula, a mensagem é enviada assim que a conexão é estabelecida. Se a data de expiração for nula, é considerado um tempo padrão para o descarte da mensagem. Esse tempo é definido no sistema com uma hora. O tempo de uma hora foi estabelecido por ser considerado um período razoável para descartar as mensagens do banco. Um tempo maior poderia levar ao inconveniente de armazenar mensagens desnecessariamente no banco de dados. Caso seja necessário um valor maior, o desenvolvedor pode defini-lo diretamente nas configurações da mensagem.

A classe *AsynchronousMessageTable* é a responsável por administrar a tabela de mensagens assíncronas. Por essa classe, é possível registrar objetos *RCAsynchronous* para serem enviados pelo canal de retorno do receptor. A classe *AsynchronousMessageTable* é um *singleton* que permite o cadastro e a exclusão de mensagens na tabela do banco de dados. Pelo método *registerRCAsynchronous*, que recebe um objeto *RCAsynchronous* como parâmetro, é possível inserir uma nova mensagem no banco de dados. Pelo método *unregisterRCAsynchronous*, que recebe um valor de índice que é único para cada mensagem, é possível excluir uma mensagem. Outros métodos possibilitam coletar informações das mensagens, como o método *getRCAsynchronous* que retorna um objeto do tipo *RCAsynchronous* referente ao índice recebido como parâmetro e o método *getStatus* que retorna a situação da mensagem de acordo com o índice. As situações possíveis são: sem sucesso (valor -1), não processada (valor 0), e enviada (valor 1).

A Figura 29 mostra o Diagrama de Classes da API de Interatividade para envio de mensagens.

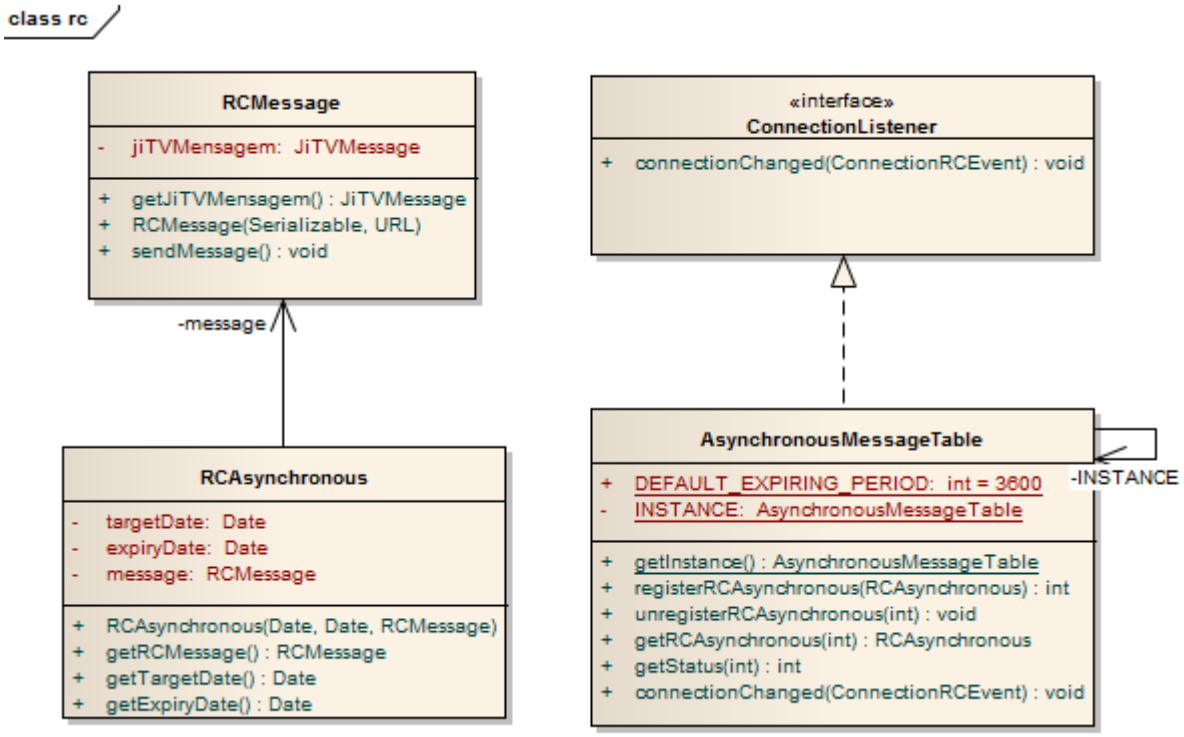


Figura 29: Classes da API de Interatividade para envio de mensagens

4.2 Ciclo de Vida da Informação

As diversas aplicações em execução em um terminal de acesso podem utilizar os serviços do JiTVPSI-CommService, bastando adicionar ao barramento um arquivo XML correspondendo à mensagem a ser enviada ou usar as classes *RCMessage* ou *AsynchronousMessagetable* da API de Interatividade.

As mensagem síncronas são enviadas pela classe *RCMessage* e não necessitam ser armazenadas no banco de dados por não demandarem novas tentativas de envio. Dessa forma, possuem um ciclo de vida muito simples pois somente existem dentro do escopo do envio. E deixam de existir para o sistema, caso sejam enviadas ou não.

As mensagens assíncronas podem entrar no sistema por um arquivo XML adicionado ao barramento de comunicação ou pela utilização da classe *AsynchronousMessagetable* com o registro de um objeto *RCAsynchronous*. Independente da forma de entrada no sistema, as mensagens assíncronas seguem o mesmo ciclo de vida por todo o sistema. O ciclo de vida das mensagens assíncronas está mostrado na Figura 30. Assim que as mensagens são criadas, são armazenadas no banco de dados. Essas mensagens somente serão apagadas do banco quanto tiverem sua data de expiração alcançada. Enquanto estiverem no banco de dados, e ainda não estiverem sido enviadas, as mensagens podem

ir para a lista de envio para que uma nova tentativa seja feita. Sendo enviadas ou não, as informações sobre as mensagens são atualizadas no banco de dados.

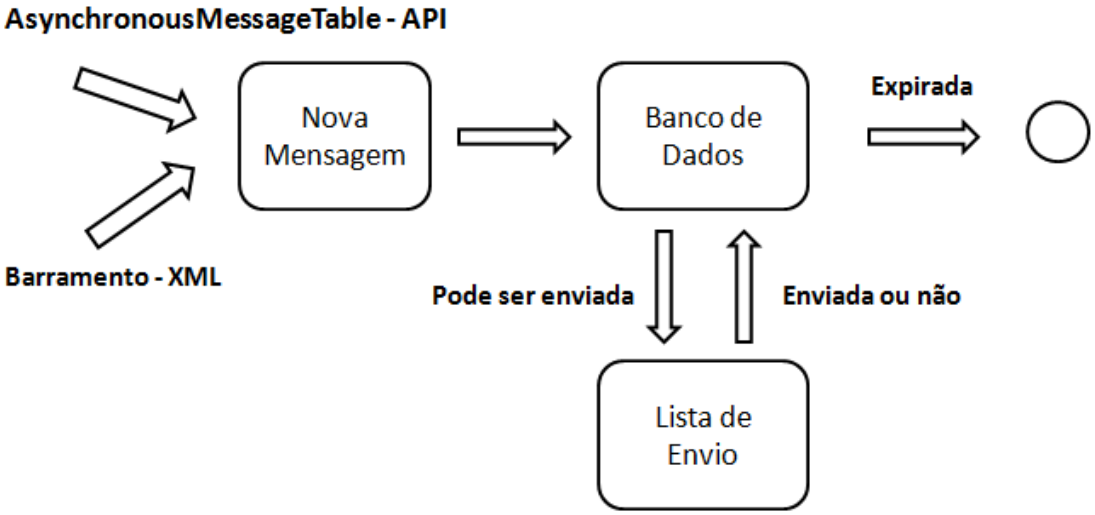


Figura 30: Ciclo de vida das mensagens assíncronas

As mensagens originadas da API possuem configuração de data de envio e de data de expiração. Já as mensagens originadas do barramento não permitem esses valores. As mensagens adicionadas ao barramento são, por padrão do sistema, enviadas ao seu destino assim que possível.

Os tipos de serviços de transporte permitidos pelo sistema JiTVPSI-CommService são: *e-mail*, *socket* (TCP), HTTP (postagem em formulário), *Web-Service* e FTP.

Uma mensagem pode ser enviada por *e-mail* por meio do serviço de SMTP. As configurações necessárias para o envio de email são: endereço do servidor e porta de acesso, endereço de *e-mail* do destinatário e do remetente e o assunto. O sistema utiliza uma biblioteca específica chamada Java *Mail* para o envio da mensagem. O sistema JiTVPSI-CommService também possibilita o envio de mensagens para um servidor de *socket*, sendo necessário informar o endereço do servidor e porta de acesso. Para essa forma de comunicação, é utilizado o objeto *java.net.Socket*. E o sistema transmite o conteúdo da mensagem como um fluxo de dados.

Para o envio de mensagens por HTTP, é necessário que o servidor remoto esteja preparado para receber a postagem de um formulário que contenha um parâmetro de entrada de dados. Dessa forma, com a utilização do objeto *java.net.URL*, o sistema JiTVPSI-CommService submete uma requisição com o método POST para um endereço remoto, passando o conteúdo da mensagem como parâmetro. Esse processo simula a

submissão de um formulário por um navegador Web. O mesmo raciocínio foi utilizado para o envio de mensagens para servidores de *Web-Service*. O conceito de *Web-Service* permite que se faça uma requisição a um servidor remoto por meio de trocas de informação por conteúdo XML. Para a comunicação por *Web-Services*, o sistema JiTVPSI-CommService utiliza uma biblioteca específica chamada kSOAP (KSOAP, 2008). Com essa biblioteca, é possível fazer uma requisição abstraindo os conceitos do serviço e da transmissão de informações por XML. A requisição é feita num elemento (método) do serviço, passando a mensagem como parâmetro. O envio de comunicação por *Web-Service* implementado pelo sistema permite o consumo de elementos remotos que tenham um e somente um parâmetro do tipo *string*, não sendo possível o uso de elementos complexos (objetos encadeados) ou com mais de um parâmetro.

Para o protocolo FTP, o sistema conecta-se ao servidor de FTP na porta de acesso e com um usuário e senha passados na configuração da mensagem. O conteúdo da mensagem é então enviado ao servidor na forma de uma arquivo que é armazenado em um diretório especifico também passado na configuração da mensagem. O nome do arquivo criado é o valor do *timestamp* da mensagem. Na implementação, foi utilizada uma biblioteca chamada *Jakarta Commons Net* que possui uma API para o tratamento do protocolo FTP.

4.3 Bibliotecas Utilizadas

No desenvolvimento do sistema JiTVPSI-CommService, foram utilizadas algumas bibliotecas específicas para aplicações em dispositivos com recursos limitados. O conjunto das bibliotecas e mais a própria aplicação precisam ter um tamanho relativamente pequeno para ser transportada por difusão ou pelo canal de retorno sem oferecer problemas com desempenho de tempo de transmissão. O sistema JiTVPSI-CommService possui um tamanho total, incluindo suas bibliotecas, de 1307 KB.

As seguintes bibliotecas de terceiros foram utilizadas:

- HSQLDB

  HyperSQL Database (HSQLDB) é um banco de dados relacional escrito em Java. Ele oferece um pequeno e rápido banco de dados que suporta tabelas armazenadas em memória ou em disco e que pode ser embarcado em outros sistemas ou funcionar em modo servidor. Na implementação do JiTVPSI-CommService, foi utilizada a

versão 1.8.0.10 que é representada pelo arquivo hsqldb.jar com tamanho de 691 KB. Possui licença de uso e distribuição completamente livre (HSQLDB, 2009).

- JavaMail API

  A JavaMail API é uma biblioteca fornecida pela Sun que proporciona um *framework* para se construir aplicações para envio de *e-mail*. Foi utilizada a versão 1.4 que é representada pelo arquivo mail-1.4.jar e possui tamanho de 380 KB.

- kSOAP 2

  kSOAP é uma biblioteca para implementação de clientes de serviços *Web-Service* SOAP em ambientes Java limitados como *applets* ou aplicações JME (CLDC / CDC / MIDP). Na implementação do JiTVPSI-CommService, foi utilizada a versão 2.1.2 que é representada pelo arquivo ksoap2-j2se-full-2.1.2.jar com tamanho de 96 KB.

- KXML 2

  KXML é uma pequena biblioteca para manipular XML especialmente desenhada para ambientes limitados. Foi desenvolvida para acessar, analisar, e exibir arquivos XML em dispositivos de pequeno porte em aplicações JME. No sistema JiTVPSI-CommService, foi necessária para executar os funcionalidades do kSOAP. Foi utilizada na versão 2.3.0 e é representada pelo arquivo kxml2-2.3.0.jar que tem tamanho de 43 KB.

- *Jakarta Commons Net*

  Jakarta Commons Net implementa o lado cliente de muitos protocolos de Internet. O propósito dessa biblioteca é fornecer acesso fundamental nos protocolos. Na implementação do JiTVPSI-CommService, foi utilizada somente parte da biblioteca que corresponde ao protocolo FTP. Foi incluída a versão 2.0 que é representada pelo arquivo commons-net-ftp-2.0.jar de 90 KB.

4.4 Considerações Finais

Neste capítulo, foi apresentada uma visão geral de sistema JiTVPSI-CommService, com suas características e detalhes de implementação. O sistema foi desenvolvido baseado em uma arquitetura modular de acordo com uma coesão funcional, em que cada uma dos módulos possui funcionalidades específicas. O módulo Monitor do Barramento de Comunicação possui a tarefa de coletar os arquivos XML de diretório determinado, tratar cada um desses arquivos como mensagens e adicioná-las em uma tabela do banco de

dados. Essa tabela armazena todas as mensagens ainda não expiradas do sistema. Um outro módulo, o Monitor do Banco de Dados, é responsável por buscar mensagens do banco de dados e inseri-las num lista para serem enviadas para o seu destino. Enviar as mensagens para seu destino é a responsabilidade do Monitor da Lista de Envio. E ainda existe um último módulo que permite o acesso ao envio de mensagens e acesso às mensagens do banco de dados por uma API de Interatividade que se baseou na API de canal de interatividade do Ginga-J.

No próximo capítulo são apresentados os testes realizados com o sistema JiTVPSI-CommService. Serão abordados detalhes das adaptações feitas em aplicações já existentes na plataforma JiTV, assim como o resultado do envio de mensagens pelo sistema JiTVPSI-CommService em todos os serviços aceitos.

5 AVALIAÇÃO DOS TESTES REALIZADOS

Com o objetivo de comprovar a eficiência das principais funcionalidades implementadas no sistema JiTVPSI-CommService, foram desenvolvidos dois módulos de aplicação para simulação de eventos e serviços relacionados à comunicação de dados entre o receptor e o ambiente remoto. O primeiro módulo da aplicação de teste é responsável pelo o envio de mensagens pelo barramento e pela API de interatividade, simulando aplicações cliente no ambiente do receptor de TV Digital. O segundo módulo da aplicação de testes é responsável por prover os serviços disponíveis para comunicação, citando *e-mail*, *socket*, HTTP, *Web-Service* e FTP, simulando, assim, o ambiente do Provedor de Serviços Interativos (PSI). Com o sistema JiTVPSI-CommService em funcionamento em conjunto com os dois módulos de teste, foi possível reproduzir de forma fiel a estrutura de um ambiente real composto pelo receptor, onde são executadas as aplicações de interatividade e o serviço JiTVPSI-CommService, e o ambiente remoto, onde são executados os diversos serviços de comunicação.

Além do desenvolvimento dos dois módulos de teste, outras duas aplicações pertencentes à plataforma JiTV foram adaptadas para utilizar o sistema JiTVPSI-CommService como suporte para o envio de mensagens para provedores de serviços. As aplicações que sofreram essas adaptações são a JiTVDengue e a JiTVEleicao. A forma de comunicação tradicional, com o uso da API de comunicação fornecida pelo *middleware*, foi substituída pela utilização do barramento de comunicação na aplicação JiTVDengue e substituída pela utilização da API de interatividade na aplicação JiTVEleicao. A Figura 31 ilustra a arquitetura do ambiente de testes.

Os testes foram desenvolvidos baseados na metodologia de testes caixa-preta, em que a meta principal é verificar se a implementação está de acordo com as definições do sistema. Os testes validam as correções do sistema JiTVPSI-CommService, garantindo que seu funcionamento está como esperado. A avaliação de desempenho não é foco dos testes realizados visto que não é possível executar o sistema em uma ambiente de terminal de acesso real, por não existir uma implementação funcional do Ginga-J homologada para a

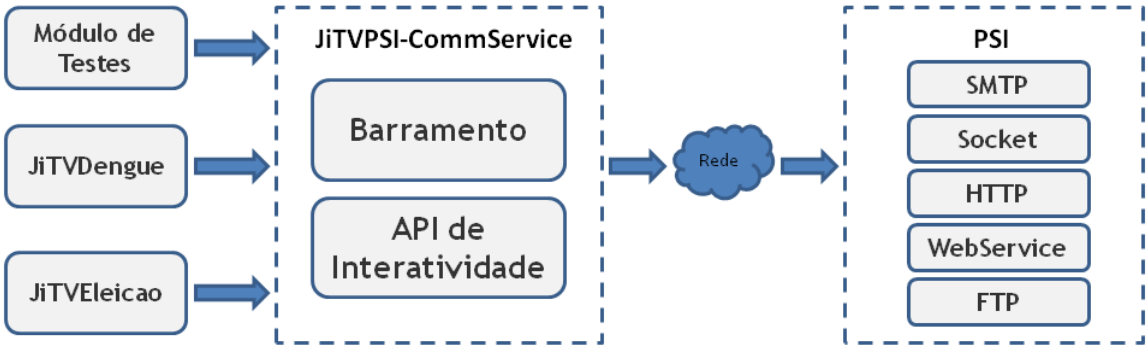


Figura 31: Arquitetura do ambiente de testes

especificação do JavaDTV. A questão de desempenho será realizada em trabalhos futuros.

Os testes com os módulos de aplicação desenvolvidos têm o objetivo de analisar a confiabilidade e eficiência do sistema em enviar as mensagens de forma adequada, no tempo programado, e suportando requisições simultâneas de envio. Os testes com as aplicações da plataforma JiTV, por outro lado, focam em verificar a viabilidade funcional do sistema, analisando as vantagens e facilidades de sua utilização.

5.1 Ambiente de Testes

O sistema JiTVPSI-CommService foi testado utilizando-se a plataforma JiTV como infraestrutura de execução dos módulos de teste. Para validar a aplicabilidade do sistema na prática, foram feitos vários experimentos no ambiente de simulação. Os testes foram focados na confiabilidade e facilidade de uso do sistema, o que é muito importante para saber sobre sua atratividade para os desenvolvedores de aplicações. As mensagens enviadas foram monitoradas em relação ao sucesso de transmissão e à adequação ao horário de envio. Observou-se também a situação do barramento e do banco de dados durante a execução dos testes.

O ambiente tanto para o JiTVPSI-CommService quando para o PSI é comum e consiste de:

- Notebook HP Pavilion dv2000 com Processador Intel Core 2 Duo T9300 - 2,53GHz, 4 GB de memória RAM sobre Windows 7 (64 bits)
- Java Standard Edition Development Kit 1.4.2
- JiTV

Por sua vez, o ambiente do PSI consiste adicionalmente de:

- Apache Tomcat 6.0.18 como contêiner Web
- Apache FTP Server 1.0.2
- Java Apache Mail Enterprise Server (JAMES) 2.3.2

Esse ambiente foi escolhido por não ser possível a execução do sistema em uma ambiente de terminal de acesso real, já que não existe uma implementação funcional do Ginga-J. Testes em ambientes reais serão realizados em trabalhos futuros, permitindo uma avaliação de desempenho em equipamento que represente de forma mais realística o ambiente do terminal de acesso (com restrições mais evidentes de processamento e memória).

Conforme já apresentado, a plataforma JiTV consiste de um conjunto de ferramentas para contemplar o ciclo de vida de um sistema interativo para TV Digital. Assim, é possível simular todas as etapas de uma aplicação, desde a sua construção e formatação, até a execução no receptor e possível interatividade com servidores remotos pelo canal de retorno. Vale ressaltar que os módulos desenvolvidos para testes e as aplicações JiTVDengue e a JiTVEleicao foram testados simulando estarem residentes no terminal de acesso do telespectador, ou seja, supondo que já se encontram instaladas no receptor e não precisam ser distribuídas e instaladas. A aplicação RecommenderTV possui somente interação local com o telespectador e não necessita da utilização do sistema JiTVPSI-CommService.

5.2 Módulos para Simulação de Testes

5.2.1 Aplicação para simulação de envio de mensagens

Um módulo de aplicação foi desenvolvido para viabilizar o envio de mensagens de teste em diversos protocolos e configurações de data de envio e expiração. Esse módulo é executado paralelamente ao sistema JiTVPSI-CommService e, aleatoriamente, escreve mensagens no barramento de comunicação e envia mensagens utilizando a API de Interatividade.

A aplicação desenvolvida para a simulação do envio de mensagens executa por um determinado período de tempo enviando mensagens sequenciais com um atraso de alguns segundos entre uma mensagem e outra. O período total de execução e o tempo de atraso são configuráveis. O objetivo é enviar mensagens indefinidamente ao sistema JiTVPSI-CommService, buscando por possíveis problemas grosseiros de desempenho ou insucesso

no envio das informações pelo canal de retorno. Esses testes não têm o objetivo de simular envios de mensagens diferentes e com conteúdo muito grande. A mensagem enviada é padronizada com um conteúdo XML recuperado da aplicação JiTVDengue.

Para tornar o envio das mensagens aleatório, foi usado o conceito matemático conhecido como módulo ou operador de resto. O operador retorna o resto da divisão entre dois números. O valor usado como numerador da divisão é o número de milissegundos desde janeiro de 1970. E o denominador da divisão é a quantidade de opções a serem escolhidas. A primeira escolha aleatória entre três opções corresponde ao tipo de mensagem a ser enviada, seja pelo barramento de comunicação, pela API de interatividade síncrona ou pela API de interatividade assíncrona. A segunda escolha aleatória entre cinco opções corresponde ao protocolo da mensagem a ser enviada, seja *e-mail*, *socket*, HTTP, FTP ou *Web-Service*. Com essa estratégia, é possível enviar mensagens nos diversos protocolos permitidos e utilizando as variadas formas de entrada das mensagens no sistema. O tempo de envio de cada mensagem também é informado aleatoriamente variando de um a sessenta minutos a partir da data de criação da mensagem. O trecho de código incompleto apresentado na Figura 32 demonstra como as mensagens são submetidas ao sistema de forma aleatória:

Esse módulo de testes também controla a situação dos diversos recursos utilizados pela aplicação, monitorando o banco de dados e o barramento de comunicação. Do banco de dados são coletadas as informações sobre a quantidade de mensagens total na tabela, e quantidade de mensagens enviadas e não enviadas que já alcançaram o horário de envio. Do barramento é coletada a informação da quantidade de arquivos que estão aguardando processamento.

5.2.2 Serviços disponíveis no PSI

Os serviços para o recebimento das mensagens pelo PSI foram disponibilizados de acordo com os protocolos de envio disponíveis pelo sistema JiTVPSI-CommService. Assim, para cada um dos protocolos, existe um serviço para recebimento e tratamento das mensagens. Esses serviços não executam nenhuma tarefa específica como armazenamento em banco de dados ou envio das informações tratadas para a emissora de TV ou algum outro interessado, apenas permanecem disponíveis para garantir a finalização da comunicação no determinado protocolo.

Para o recebimento de mensagens de *e-mail*, foi disponibilizado um servidor de SMTP, que é o serviço responsável por receber e redirecionar mensagens de *e-mail* aos desti-

```
while(currentTimeMillisInicio+TEMPO_EXECUCAO<System.currentTimeMillis()){
  long t = System.currentTimeMillis();
  if (t1%3==0) { // API assíncrona
    if (t%5==0) {
      // Submete mensagem assíncrona por email.
    }else if (t%5==1) {
      // Submete mensagem assíncrona por socket.
    }else if (t%5==2) {
      // Submete mensagem assíncrona por http.
    }else if (t%5==3) {
      // Submete mensagem assíncrona por FTP.
    }else if (t%5==4) {
      // Submete mensagem assíncrona por WebService.
    }
  } else if (t1%3==1) { // Barramento
    if (t%5==0) {
      // Submete mensagem assíncrona por email no barramento.
    }else if (t%5==1) {
      // Submete mensagem assíncrona por socket no barramento.
    }else if (t%5==2) {
      // Submete mensagem assíncrona por http no barramento.
    }else if (t%5==3) {
      // Submete mensagem assíncrona por FTP no barramento.
    }else if (t%5==4) {
      // Submete mensagem assíncrona por WebService no barramento.
    }
  } else { // API síncrona
    if (t%5==0) {
      // Submete mensagem assíncrona por email no barramento.
    }else if (t%5==1) {
      // Submete mensagem assíncrona por socket no barramento.
    }else if (t%5==2) {
      // Submete mensagem assíncrona por http no barramento.
    }else if (t%5==3) {
      // Submete mensagem assíncrona por FTP no barramento.
    }else if (t%5==4) {
      // Submete mensagem assíncrona por WebService no barramento.
    }
  }
  Thread.sleep(ATRASO);
}
```

Figura 32: Código para geração de mensagens aleatórias.

natários. O sistema utilizado foi o Apache JAMES que é uma ferramenta de código aberto, escrita em Java, que suporta o protocolo de mensagens de *e-mail*, dentre outros. De forma semelhante, o serviço de FTP foi disponibilizado por um servidor de código aberto também escrito em Java, chamado Apache FtpServer. O servidor de FTP recebe as mensagens em forma de um arquivo texto e armazena cada um dos arquivos de acordo com os parâmetros passados no protocolo.

Uma aplicação Web foi desenvolvida utilizando-se os assistentes da IDE Eclipse com o objetivo de receber mensagens no protocolo HTTP e no padrão *Web-Service*. Nessa aplicação, o serviço HTTP foi implementado com um arquivo *Java Server Page* (JSP) que recebe um determinado parâmetro e o exibe no arquivo de saída do servidor Web. Esse parâmetro será utilizado para receber o conteúdo da mensagem. Para o padrão *Web-Service*, foi implementado um serviço que possui um método que recebe um parâmetro do tipo *String*. Da mesma forma, esse parâmetro será utilizado para receber o conteúdo da

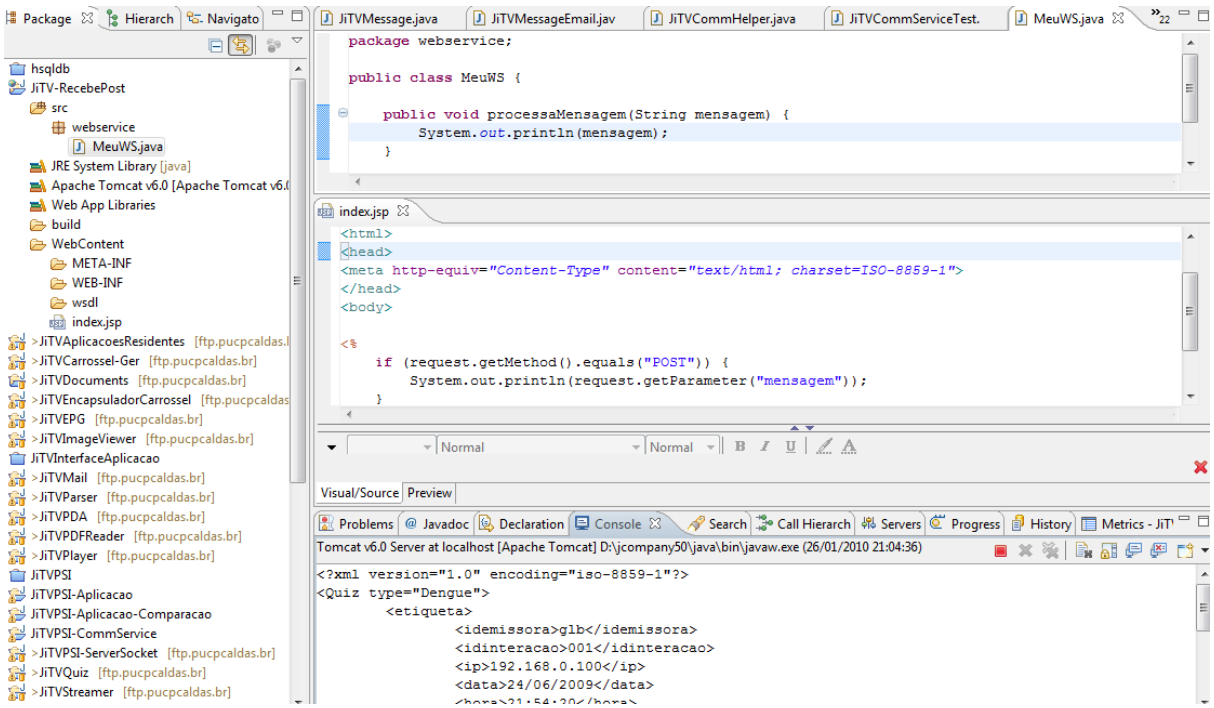


Figura 33: Demonstração da execução da aplicação Web para testes

mensagem e é exibido no arquivo de saída do servidor Web. A Figura 33 mostra a IDE Eclipse com perspectiva de desenvolvimento com a aplicação Web aberta à esquerda, e, nas visualizações à direita, a classe de *Web-Service*, o arquivo JSP e a saída do servidor Web com um mensagem recebida.

O serviço de *socket* (TCP) é proporcionado por uma aplicação chamada JiTVPSI-ServerSocket que pertence à plataforma JiTV. Essa aplicação é responsável por receber mensagens por *socket* na porta 1234 e armazená-las em arquivos no disco rígido do servidor. Além disso, apresenta um interface com as mensagens recebidas, como mostrado na Figura 34.

Os fatores de escolha dos sistemas de serviços apresentados nesta seção, assim como suas configurações, não são relevantes ao contexto deste trabalho, visto que, como o sistema JiTVPSI-CommService manipula os protocolos de acordo com suas especificações, qualquer sistema que implemente os devidos protocolos poderá ser utilizado.

5.2.3 Testes de Confiabilidade e Eficiência

A confiabilidade de um sistema pode ser atribuída à capacidade de executar e manter seu funcionamento em circunstâncias normais, bem como em circunstâncias inesperadas.

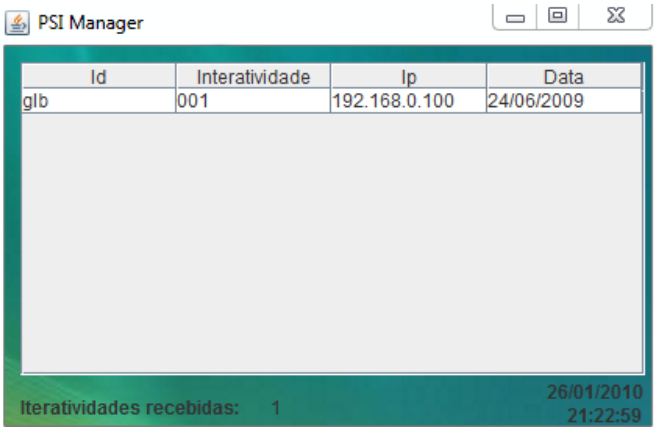


Figura 34: Interface da Aplicação JiTVPSI-ServerSocket

Já a eficiência de um sistema visa medir seu rendimento em relação aos resultados obtidos e aos recursos empregados. Com o objetivo de obter informações quanto à confiabilidade e à eficiência do sistema JiTVPSI-CommService, o módulo de testes para envio de mensagens foi executado durante duas horas de forma a submeter mensagens ao sistema JiTVPSI-CommService com um atraso de um segundo entre elas. O módulo de testes apresentou as informações de mensagens submetidas de acordo com a seguinte distribuição:

```
Mensagens submetidas:                    4708
Mensagens que retornaram exceção:           0

Mensagens por forma de envio
  Barramento de Comunicação:             1125
  API de Interatividade (assíncrono):    1777
  API de Interatividade (síncrono):      1806

Mensagens por tipo de protocolo
  e-mail:                                 465
  socket:                                1127
  FTP:                                   1263
  HTTP:                                   816
  Web-Service:                           1037
```

O sistema JiTVPSI-CommService gera uma saída de *logging* em arquivo com as informações de mensagens enviadas com o seu protocolo e seus tempos de criação, envio e expiração. Esse arquivo de saída foi tabulado e sumarizado para que as informações pudessem ser comparadas com as informações do módulo de testes. Diante das infor-

mações apresentadas pelo módulo de testes e pelo sistema JiTVPSI-CommService, alguns itens foram analisados com o objetivo de medir sua confiabilidade e eficiência. Esses itens são:

- Sucesso no envio das mensagens

  De acordo com o relatório apresentado pelo módulo de testes, todas as mensagens submetidas foram supostamente enviadas, visto que nenhuma delas apresentou um exceção ou erro em sua submissão. Mas somente isso não basta para garantir que todas as mensagens foram realmente enviadas, pois pode ocorrer algum erro no envio ou na criação do canal de comunicação com o servidor remoto. Para garantir o envio efetivo, é necessário analisar os relatórios apresentados pelo sistema JiTVPSI-CommService. O sistema JiTVPSI-CommService apresentou um total de mensagens enviadas igual ao total de mensagens submetidas pelo módulo de testes. Com essa informação, garante-se que 100% das mensagens foram enviadas ao seu destino.

- Eficácia no tempo de envio das mensagens

  Para analisar a eficácia no tempo de envio das mensagens, é preciso dividir as mensagens em síncronas e assíncronas. As mensagens síncronas são aquelas enviadas no mesmo momento de sua submissão no sistema. No sistema JiTVPSI-CommService, as mensagens síncronas somente são postadas por meio da API de Interatividade pela classe *RCMessage*, conforme detalhado na Seção 4.1.4. As mensagens assíncronas são as que podem ser enviadas em um momento posterior de acordo com um tempo definido ou quando o canal de comunicação estiver disponível.

  Para mensagens síncronas, obteve-se um tempo mínimo de envio de 1000 milissegundos e um tempo máximo de 2030 milissegundos, para as 1806 mensagens enviadas. Nas mensagens assíncronas, a variação de tempo pode ser maior por estarem sujeitas à concorrência de mensagens do banco da dados e ainda do tempo de execução dos monitores de banco de dados e de envio de mensagens. Assim, obteve-se um tempo mínimo de envio de 93 milissegundos e um tempo máximo de 29813 milissegundos, nas 2902 mensagens enviadas, com um tempo médio de envio de 4270 milissegundos.

  Com esses números é possível notar que a escolha dos tempos de execução dos monitores foi satisfatória já que os testes não focam em verificação de desempenho. Para aplicações interativas que estejam relacionadas a programas ao vivo, como aplicações de enquete interativas para programas ao vivo, é recomendável que usem o serviço de mensagens síncronas que em que o tempo máximo de envio esteve

na faixa de dois segundos. Para outros tipos de aplicações e serviços interativos como *t-commerce*, *t-learning* e vídeo sob demanda, em que não há vínculo com nenhum sistema que deva receber as mensagens com precisão de tempo, pode-se utilizar as mensagens assíncronas pelo barramento de comunicação ou pela API de interatividade.

- Situação do banco de dados

  O gráfico apresentado na Figura 35 demonstra a situação do banco de dados durante a execução dos testes. Nota-se que a quantidade máxima de mensagens armazenadas no banco de dados, 1903, está no tempo de 2 horas, que é o momento em que mensagens param de ser submetidas ao sistema JiTVPSI-CommService. No pico de 2 horas, existiam 1418 mensagens já enviadas aguardando expiração, 484 mensagens com tempo de envio ainda não alcançado e apenas 1 mensagem com tempo de envio alcançado e ainda não enviada. É possível verificar que as mensagens vão sendo eliminadas do banco de dados gradativamente de acordo com seu tempo de expiração. No momento de 2 horas, um total de 2902 mensagens assíncronas já haviam passado pelo sistema, e somente 1903 ainda permaneciam no banco de dados, mostrando que 999 mensagens já haviam sido excluídas. Depois do tempo de 2 horas, a quantidade de mensagens começa a diminuir, até o momento próximo de 4 horas quando não há mais mensagens no banco de dados.

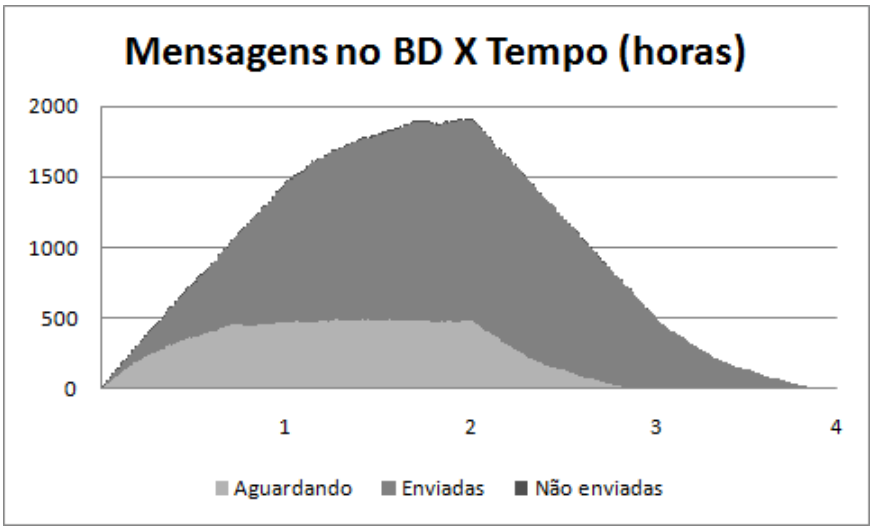


Figura 35: Gráfico com situação do banco de dados

- Eficiência na captura de mensagens no barramento de comunicação

  Outro ponto importante de manipulação do sistema JiTVPSI-CommService é o barramento de comunicação. Durante todo o tempo dos testes, o barramento de co-

municação foi monitorado para obtenção da quantidade de arquivos de mensagens existentes e verificação da eficiência do sistema em capturar adequadamente essas mensagens. A maior quantidade de arquivos encontrada no barramento de comunicação foi 1. Isso mostra que o sistema JiTVPSI-CommService estava buscando as mensagens em tempo hábil, visto que o módulo de testes inseriu 1125 mensagens no barramento durante todo o teste.

- Concorrência no envio de mensagens

  Como existem várias formas de submeter mensagens ao sistema JiTVPSI-CommService, um possível problema é a concorrência no envio de mensagens síncronas e assíncronas. Pode ocorrer de mensagens assíncronas estarem sendo enviadas e uma mensagem síncrona ser submetida. De forma inversa, pode também ocorrer de uma mensagem síncrona estar sendo enviada e o monitor de lista de envio requerer o envio de mensagens assíncronas. O sistema JiTVPSI-CommService deve tratar ambos os casos para não haver perdas de mensagens. Quando as mensagens assíncronas estiverem sendo enviadas e uma mensagem síncrona for submetida, a mensagem síncrona aguarda até que o canal de comunicação possa ser reservado para ela. Essa tarefa está sendo feita de forma correta visto que o módulo de testes não recebeu nenhuma exceção de impossibilidade de reserva da interface de comunicação. Quando uma mensagem síncrona estiver sendo enviada e a monitor de envios requerer o canal de comunicação, o sistema JiTVPSI-CommService aborta o envio daquele momento e aguarda até uma próxima tentativa. Esse evento ocorreu 345 vezes durante as duas horas de testes, e não causou perda de mensagens visto que todas as mensagens submetidas foram enviadas. O único efeito colateral dessa concorrência é um aumento no tempo de envio das mensagens que precisam aguardar até que o canal possa ser reservado. A concorrência no envio de mensagens deve ser averiguada de forma mais contundente em ambiente de testes simulando um terminal de acessos real e utilizando aplicações com maior grau de interatividade, que será realizado em trabalhos futuros.

5.3 Adaptação das Aplicações da Plataforma JiTV

5.3.1 Aplicação JiTVDengue

Testes foram realizados sobre a plataforma JiTV utilizando a aplicação JiTVDengue. Essa aplicação simula a interatividade com o telespectador por meio de uma pesquisa sobre

fatores ambientais na residência do indivíduo com o objetivo de detectar a existência de risco iminente de focos do mosquito da dengue no local. A aplicação é apresentada ao usuário em forma de um questionário com 4 perguntas com 3 possibilidades de resposta: Sim, Talvez e Não. O objetivo dessa aplicação é obter informações sobre a situação de suas residências quanto à chance de foco de mosquito da dengue e fazer com que essas informações cheguem às autoridades competentes para que elas tomem as decisões cabíveis.

Originalmente, na plataforma JiTV, a aplicação JiTVDengue se comunica diretamente pela comunicação por *socket* na porta 1234 com um serviço remoto representado pela aplicação JiTVPSI-ServerSocket. Neste contexto, a aplicação residente no terminal de acesso do telespectador tem que se responsabilizar pelo envio da resposta do usuário até o servidor externo, se preocupando com falhas de qualquer natureza na comunicação entre os dois pontos.

Para que fosse possível os testes no projeto JiTVPSI-CommService, uma alteração na aplicação JiTVDengue foi realizada para que não envie a resposta diretamente por *socket* para o servidor externo, e sim que utilize o barramento de comunicação existente no terminal de acesso, operacionalizado pelo sistema JiTVPSI-CommService. Dessa forma, o serviço no barramento se responsabiliza pela entrega da mensagem ao seu destino, independente de possíveis problemas ocorridos durante a tentativa de envio.

A diminuição da complexidade no desenvolvimento da aplicação que deseja enviar os dados para o meio externo pode ser vista no código fonte apresentado nas listagens apresentadas a seguir. Essas listagens mostram o trecho da aplicação em que é feita a substituição do código responsável pela comunicação dos dados com o servidor externo pelo código que envia a mensagem ao barramento de comunicação. Outros trecho de código foram alterados na aplicação, mas que são de menor relevância. A única tarefa da aplicação, neste novo contexto, é escrever um arquivo no formato XML no barramento de comunicação por meio de uma API própria do JiTVPSI-CommService. Além da diminuição da complexidade, a JiTVDengue também apresentou uma diminuição de cerca de 8% de seu tamanho, passando de 1460 para 1347 linhas efetivas de código.

Trecho de código original da aplicação JiTVDengue:

```
private void enviaDados(String xmlEnvio) {
  try {
    // conectando ao servidor remoto
```

```
    Socket socketCliente = new Socket("servidor.pucminas.br", 1234);
    System.out.println("Conectado ao servidor");

    // ler conteúdo do arquivo XML
    ByteArrayInputStream readXML = null;
    // enviar dados para o socket
    DataOutputStream writeXML = null;
    // armazenar dados do arquivo
    byte bufferFileXML[] = new byte[4096];

    readXML = new ByteArrayInputStream(xmlEnvio.getBytes());
    writeXML = new DataOutputStream(socketCliente.getOutputStream());

    readXML.read(bufferFileXML);
    writeXML.write(bufferFileXML);

    socketCliente.close();
    readXML.close();
    writeXML.close();
  } catch (IOException e) {
    System.out.println("Falha ao conectar ao servidor");
    e.printStackTrace();
  }
}
```

Novo trecho de código da aplicação JiTVDengue:

```
private void enviaDados(String xmlEnvio) {
  JiTVCommHelper.getInstance().
         sendSocketMessage("servidor.pucminas.br", 1234, xmlEnvio);
}
```

A Figura 36 mostra a aplicação JiTVDengue em execução no simulador, juntamente com o controle remoto virtual para entrada de dados similar ao controle remoto real de um televisor. Nesta figura, há também destaque para a aplicação JiTVPSI-ServerSocket que recebeu corretamente a informação supostamente enviada pelo telespectador.

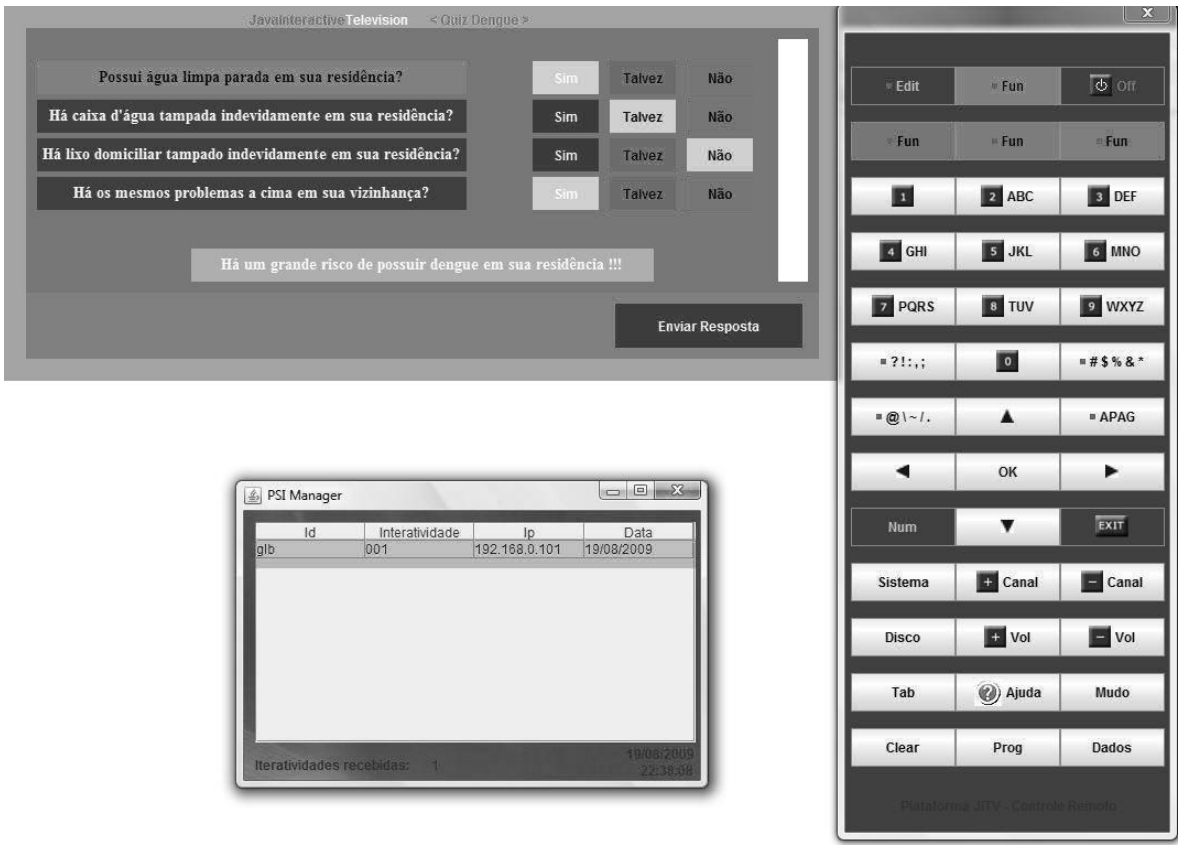


Figura 36: Interface das aplicações JiTVDengue e JiTVPSI-ServerSocket

5.3.2 *Aplicação JiTVEleicao*

Outros testes também foram realizados com a aplicação JiTVEleicao que faz parte da plataforma JiTV. Essa aplicação simula uma eleição presidencial, em que são escolhidos deputados estadual e federal, governador, senador e presidente da república, de forma similar ao sistema de votação com urnas eletrônicas existente no Brasil. A aplicação JiTVEleicao requisita ao eleitor que entre com seu número de CPF (Cadastro de Pessoa Física) e com seu número de título de eleitor que são enviados para um servidor externo para que seja validado. Após a validação, inicia-se o processo de votação. O eleitor deve digitar o número do seu candidato ou o número do partido, em caso de eleição de cargos que permitem o sistema proporcional (deputados e senadores), e apertar a tecla correspondente para confirmação. Após o preenchimento de todos os cargos, a confirmação submete o voto para o servidor externo para que seja devidamente computado na eleição. A Figura 37 ilustra a tela de votação para governador da aplicação JiTVEleicao, e, ao lado, o controle remoto de simulação.

O envio das informações com o CPF e título de eleitor e com o voto do eleitor, originalmente, eram submetidos pela aplicação JiTVEleicao pela comunicação por *socket*

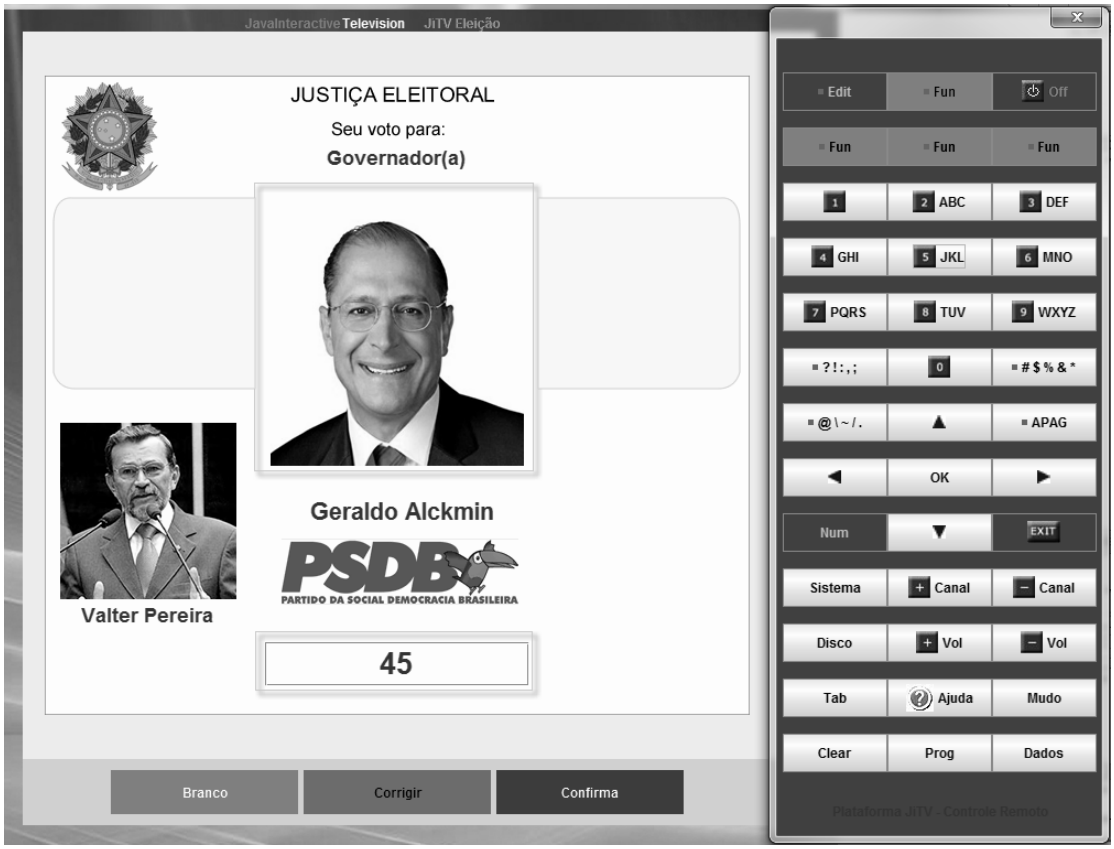


Figura 37: Interface da aplicação JiTVEleição

na porta 7600. Como também ocorria na aplicação JiTVDengue, toda a responsabilidade de conexão e envio da mensagem pelo canal de retorno está com a aplicação JiTVEleicao.

Para os testes com o sistema JiTVPSI-CommService, a aplicação JiTVEleicao foi alterada para que enviasse suas mensagens pela API de Interatividade existente no sistema de comunicação. As mensagens foram programadas para serem enviadas assincronamente para o servidor externo. Dessa forma, a aplicação JiTVEleicao não necessitou aguardar até o envio da mensagem, pois assim que a mensagem foi submetida ao sistema JiTVPSI-CommService, a mesma entrou no fluxo de envio para ser transportada assim que possível.

Trecho de código utilizado para envio das mensagens da aplicação JiTVEleicao:

```
try {
  RCMessage message = new RCMessage(this.mensagem.toString(),
              new URL(null, "socket://"+ipServidor+":"+portaServidor,
                                          new JiTVURLStreamHandler()));
  RCAsynchronous rca = new RCAsynchronous(null,null, message);
  AsynchronousMessageTable.getInstance().registerRCAsynchronous(rca);
} catch (MalformedURLException e) {
```

```
  e.printStackTrace();
}
```

Da mesma forma que ocorreu na aplicação JiTVDengue, na aplicação JiTVEleicao também houve diminuição na complexidade pela retirada da responsabilidade de garantia de entrega das mensagens. Além disso, houve uma diminuição do tamanho da aplicação de 3848 para 3810 linhas efetivas de código, cerca de 1

Seria relevante que os testes fossem realizados em aplicações com maior grau de interação com o ambiente externo, como um jogo *online*. Como a plataforma JiTV não possui aplicações com essa característica, e o Ginga-J ainda não possui implementação funcional disponível, testes com aplicações com maior interação serão deixados para trabalhos futuros.

5.4 Considerações Finais

Neste capítulo foi apresentado o ambiente de testes criado para simular o funcionamento do sistema JiTVPSIComm-Service. O ambiente é composto por um módulo de aplicação de testes desenvolvido especialmente para submeter mensagens para o JiTVPSI-Comm-Service. Do lado do PSI, foram instalados serviços para receber mensagens nos protocolos homologados. Durante duas horas uma variedade de mensagens, síncronas e assíncronas e nos vários protocolos aceitos, foram submetidas ao sistema JiTVPSIComm-Service. A aplicação JiTVPSI-CommService enviou mensagens de forma assíncrona, com uma espera de no máximo 30 segundos para que a mensagem alcançasse seu destino final, de acordo com testes realizados. O sistema enviou todas as mensagens submetidas. Além disso, a expiração de mensagens no banco de dados e a quantidade de mensagens no barramento de comunicação foram controladas adequadamente. Por fim, testes com as aplicação JiTVDengue e JiTVEleicao foram realizados, adaptando a forma original de comunicação, pela comunicação utilizando o barramento de comunicação e a API de Interatividade.

A seguir serão detalhadas as conclusões sobre o projeto assim como os próximos passos a serem seguidos para melhor desenvolvimento do sistema JiTVPSIComm-Service.

6 CONCLUSÕES E TRABALHOS FUTUROS

6.1 Conclusões

Este trabalho apresentou a arquitetura e aspectos da estratégia de implementação de um sistema para promover a interatividade em aplicações para TV Digital, com foco no ciclo de vida do envio de informações do terminal de acesso do telespectador para o provedor de serviços interativos. Como resultado do estudo, criou-se um sistema chamado JiTVPSIComm-Service que fornece todo o suporte necessário para a transmissão de dados para o PSI.

O projeto segue a tendência de aplicações de âmbito genérico e tem como objetivo principal propor um Sistema de Comunicação no ambiente do terminal de acesso do telespectador, em *Middleware* Genérico, para facilitar transmissão de dados para o Provedor de Serviços Interativos (PSI) através do canal de retorno, permitindo envio de informações ao difusor e ao banco de dados para armazenamento.

O sistema JiTVPSIComm-Service está integrado à plataforma JiTV, que, com sua estratégia de validação de requisitos do Sistema Brasileiro de TV Digital Terrestre (SBTVD-T), serviu de suporte para o desenvolvimento do sistema de comunicação e para os testes realizados.

O sistema JiTVPSI-CommService foi desenvolvido baseado em uma arquitetura modular de acordo com uma coesão funcional, em que cada uma dos módulos possui funcionalidades específicas. O módulo Monitor do Barramento de Comunicação possui a tarefa de coletar os arquivos XML de diretório determinado, tratar cada um desses arquivos como mensagens e adicioná-las em uma tabela do banco de dados. Essa tabela armazena todas as mensagens ainda não expiradas do sistema. Um outro módulo, o Monitor do Banco de Dados, é responsável por buscar mensagens do banco de dados e inseri-las num lista para serem enviadas para o seu destino. Enviar as mensagens para seu destino é a responsabilidade do Monitor da Lista de Envio. E ainda existe um último módulo que permite o acesso ao envio de mensagens e acesso às mensagens do banco de dados por

meio de uma API de Interatividade que se baseou na API de canal de interatividade do Ginga-J.

Testes foram realizados em ambiente de simulação com a plataforma JiTV. Houve uma significativa diminuição na complexidade do código das aplicações no que diz respeito à comunicação com o PSI através do canal de retorno. As tarefas de comunicação e controle de acesso ao canal de retorno foram retiradas das aplicações e completamente transferidas para a aplicação JiTVPSI-CommService.

O ambiente de testes é composto por um módulo de aplicação de testes desenvolvido especialmente para submeter mensagens para o JiTVPSIComm-Service. Do lado do PSI, serviços foram instalados para receber mensagens nas formas de serviços homologadas. Durante duas horas uma variedade de mensagens, síncronas e assíncronas e nos vários serviços aceitos, foram submetidas ao sistema JiTVPSIComm-Service. O sistema enviou todas as mensagens submetidas. Além disso, a expiração de mensagens no banco de dados e a quantidade de mensagens no barramento de comunicação foram controladas adequadamente. Testes com as aplicação JiTVDengue e JiTVEleicao foram realizados, adaptando a forma original de comunicação, pela comunicação utilizando o barramento de comunicação e a API de Interatividade, sendo que as mensagens continuaram chegando ao seu destino como originalmente.

6.1.1 Trabalhos Futuros

A aplicação JiTVPSI-CommService apresentou resultados satisfatórios quanto ao envio dos dados para o PSI com os serviços de comunicação propostos. Algumas outras estratégias de implementação ainda precisam ser adotadas de forma a criar novas facilidades para as aplicações se comunicarem com o meio externo. Os pontos para melhorias futuras do sistema JiTVPSIComm-Service são:

- desenvolvimento de novos serviços para envio de mensagens: Atualmente somente são suportados os serviços *e-mail*, *socket*, HTTP, FTP e *Web-Service*, seria necessário o desenvolvimento de novos serviços de comunicação e mesmo a melhoria dos serviços existentes.

- recebimento de mensagens: Uma funcionalidade desejada seria o recebimento de mensagens pelo sistema de comunicação. Atualmente é possível somente o envio de informações para o PSI, seria relevante que o sistema JiTVPSIComm-Service

pudesse receber mensagens de um servidor remoto e repassá-las às aplicações interativas em execução no terminal de acesso.

- testes em um terminal de acesso real: Os testes feitos por simulação utilizando a plataforma JiTV tiveram importante valor, mostrando resultados e auxiliando na descoberta de falhas no sistema. É de extrema importância que os testes sejam executados utilizando um terminal de acesso real para se comprovar os resultados obtidos em simulação.

- testes em aplicações com alto grau de interação: Outros testes devem ser realizados em aplicações com um maior grau de interação para verificar o desempenho do sistema JiTVPSIComm-Service em cenário de maior exigência.

- testes com interfaces de comunicação que demandem conexão: Da mesma forma, é de extrema importância que testes sejam realizados em terminais de acesso que demandem estabelecimento de conexão com uma interface de rede.

- testar o sistema em dispositivos móveis: Com o crescimento da demanda de sistemas e o surgimento de aparelhos portáteis que tenham acesso à TV Digital, vislumbra-se a necessidade de testes do sistema JiTVPSIComm-Service em dispositivos móveis de forma a garantir seu funcionamento nos mais diversos ambientes.

- extensibilidade por *Open Services Gateway initiative* (OSGI): OSGi é uma especificação de um serviço ou módulo na plataforma Java em tempo de execução. O modelo OSGI permite que se ative, desative, atualize ou desinstale componentes e serviços dinamicamente. O poder de ativação dinâmica de módulos de sistema proposto pelo OSGI poderia ser utilizado para estender o sistema JiTVPSIComm-Service de forma dinâmica para que qualquer desenvolvedor pudesse criar um novo módulo do sistema e publicá-lo no terminal de acesso com um simples carregamento por difusão.

REFERÊNCIAS

ANDREATA, J. A. *InteraTV: Um Portal para Aplicações Colaborativas em TV Digital Interativa Utilizando a Plataforma MHP*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Florianópolis, 2006. Engenharia Elétrica.

ARIB. Arib std-b23, application execution engine platform for digital broadcasting. *ARIB Standard*, 2004.

ATSC. DTV Application Software Environment Level 1 (DASE-1) PART 2: Declarative Applications and Environment. 2003.

BATISTA, C. E. C. F. et al. Tvgrid: A grid architecture to use the idle resources on a digital tv network. IEEE Computer Society, Washington, DC, USA, p. 823–828, 2007.

BECKER, V.; MONTEZ, C. TV Digital Interativa: conceitos e tecnologias. *SBC. (Org.). WebMidia e LA-Web 2004*, p. 39–77, 2004.

BECKER, V. et al. Datacasting e desenvolvimento de serviços e aplicações para tv digital interativa. In: *TEIXEIRA, Cesar Augusto Camilo; BARRÉRE, Eduardo; ABRÃO, Iran Calixto. (Org.). Web e Multimídia: Desafios e Soluções.* Poços de Caldas: [s.n.], 2005. v. 01, p. 01–30.

BEDICKS, G. et al. Results of the ISDB-T system tests, as part of digital TV study carried out in Brazil. *Broadcasting, IEEE Transactions on*, v. 52, p. 38–44, March 2006.

BOLAñO, C.; VIEIRA, V. R. TV digital no Brasil e no mundo: estado da arte. *Revista de Economia Política da las Tecnologías de la Información y Comunicación*, 2004.

CAMPISTA, M. E. M. et al. The ad hoc return channel: a low-cost solution for brazilian interactive digital tv. *IEEE Communications Magazine*, v. 45, p. 136–143, January 2007.

CARVALHO, E. R. de et al. The brazilian digital television system access device architecture. *Journal of Brazilian Computer Society*, v. 13, p. 95–113, August 2007.

CESAR, P.; VUORIMAA, P.; VIERINEN, J. A graphics architecture for high-end interactive television terminals. *ACM Trans. Multimedia Comput. Commun. Appl.*, ACM, v. 2, n. 4, p. 343–357, 2006.

DAVIC. *Disponível em http://www.davic.org. Acesso em junho de 2008.* 2008.

Digital Video Broadcasting. *Globally Executable MHP (GEM) Specification 1.0.3. Disponível em http://pda.etsi.org/pda/home.asp. Acesso em junho de 2008.* 2008.

DVB Return Channel Terrestrial. *Disponível em: http://www.broadcastpapers.com/whitepapers/DVBReturnChannel.pdf. Acesso em Abril de 2009.* 2009.

ECMAScript. 2009. Disponível em: http://www.ecmascript.org/docs/tc39-2009-043.pdf. Acesso em dezembro de 2009.

ETSI. *Multimedia Home Platform (MHP) Especification 1.1.1*. 2003.

ETSI. *TS 102 819: Globally Executable MHP (GEM)*. 2005.

FARIAS, M. C.; CARVALHO, M. M.; ALENCAR, M. S. Digital television broadcasting in brazil. *IEEE MultiMedia*, IEEE Computer Society, Los Alamitos, CA, USA, v. 15, n. 2, p. 64–70, 2008. ISSN 1070-986X.

FERNANDES, J.; LEMOS, G.; SILVEIRA, G. Introdução à televisão digital interativa: arquitetura, protocolos, padrões e práticas. *Jornada de Atualização em Informática do Congresso da Sociedade Brasileira de Computação (JAISBC2004)*, 2004.

FERRETTI, S.; ROCCETTI, M.; ANDRICH, J. Living the TV revolution: unite MHP to the web or face IDTV irrelevance! In: *WWW*. [S.l.: s.n.], 2006. p. 899–900.

Fórum SBTVD. 2009. ABNT/CEE-85. Disponível em: http://tvdilab.inf.pucpcaldas.br/repositorioopenginga/SBTVD-T-Parte4-NormaGingaJ.pdf. Acesso em dezembro de 2009.

Fórum SBTVD. *Fórum do Sistema Brasileiro de Televisão Digital define o padrão para a interatividade. Disponível em http://www.forumsbtvd.org.br/materias.asp?id=127. Acesso em Outubro de 2009.* 2009.

FUKS, H. et al. Middleware de Integração entre o Ambiente AulaNet e o Ginga. *Congresso da Sociedade Brasileira de Engenharia de Televisão*, São Paulo, 2007.

GIL, A. et al. *Surfing the WEB on TV: The MHP Approach*. Lausanne, Switzerland: [s.n.], 2002.

HAVi, Inc. *HAVi. Disponível em http://www.havi.org/. Acesso em junho de 2008.* 2008.

HSQLDB. 2009. Disponível em: http://hsqldb.org/. Acesso em março de 2009.

Instituto Brasileiro de Geografia e Estatística. *IBGE. Pesquisa Nacional Por Amostra de Domicílios. Disponível em: http://www.ibge.gov.br/home/estatistica/populacao/trabalhoerendimento/pnad2008/. Acesso em outubro de 2009.* 2008.

ISDTV-T Forum. *ISDTV-T Standard*. 2006.

JiTV. *Plataforma JiTV (Java Interactive Television). Disponível em http://tvdilab.inf.pucpcaldas.br. Acesso em junho de 2008.* 2008.

JONES, J. DVB-MHP/Java TV data transport mechanisms. In: *Proceedings of the 40th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Pacific '02)*. Sydney, Australia: [s.n.], 2002. v. 10, p. 115–121.

JUNIOR, M. M.; HIRAKAWA, A. R.; JATOBA, P. L. de O. Samba project: A test bed for plc application as a digital inclusion tool. In: . [S.l.: s.n.], 2008.

KSOAP. *Ksoap 2. Disponível em http://ksoap2.sourceforge.net/. Acesso em junho de 2009.* 2008.

LEITE, L. E. C. et al. Flextv - uma proposta de arquitetura de middleware para o sistema brasileiro de tv digital. *Revista de Engenharia de Computação e Sistemas Digitais*, v. 2, p. 29–50, 2005.

LIN, F.; SUN, J. An interactive media platform scheme for dtv receiver compliant with mhp. *Consumer Electronics, IEEE Transactions on*, v. 53, n. 4, p. 1370–1377, November 2007.

MARGALHO, M.; FRANCêS, C. R.; COSTA, J. C. W. Canal de Retorno para TV Digital com Interatividade Condicionada por Mecanismo de Sinalização Contínua e Provisionamento de Banda Orientado a QoS. *IEEE LATIN AMERICA TRANSACTIONS*, v. 5, n. 5, 2007.

MARGALHO, M.; FRANCêS, R.; WEYL, J. Return Path in Brazilian Digital Television with Interactivity Based on Continuous Signalization Mechanism and QoS Bandwidth Control. *Latin America Transactions, IEEE (Revista IEEE America Latina)*, v. 5, p. 367–372, September 2007.

MELONI, L. G. P. Return channel for the brazilian digital television system-terrestrial. *Journal of Brazilian Computer Society*, v. 13, p. 83–94, March 2007.

MESQUITA, M. G.; SOUZA, O. de. Interactive Applications for Digital TV Using Internet Legacy - A Successful Case Study. *VI International Telecommunications Symposium (ITS2006)*, Fortaleza-CE, Brasil, p. 472–476, 2006.

MHP. *Disponível em http://www.mhp.org. Acesso em junho de 2009.* 2009.

MIRABELLA, O.; BRISCHETTO, M.; RAUCEA, A. Home control system over an mhp based architecture. *Conference on Human System Interactions*, p. 281–286, May 2008.

PENG, C.; VUORIMAA, P. Digital Television Application Manager. *Proceedings of the IEEE International Conference on Multimedia and Expo 2001*, Tokyo, Japan, p. 685–688, August 2001.

PIESING, J. The DVB Multimedia Home Platform (MHP) and Related Specifications. *Proceedings of the IEEE*, v. 94, p. 237–247, January 2006.

POLO, E. I.; HIRAKAWA, A. R.; JUNIOR, M. M. Plc as a return channel for interactive digital tv. In: . [S.l.: s.n.], 2007. p. 707–711.

RICHER, M. S. et al. The ATSC Digital Television System. *Proceedings of the IEEE*, v. 13, p. 37–43, 2006.

SANTOS, D. T. dos; VALE, D. T. do; MELONI, L. G. P. Digital TV and Distance Learning: Potentials and Limitations. *Frontiers in Education Conference*, IEEE Computer Society, Los Alamitos, CA, USA, 2006.

SANTOS JUNIOR, J. B. dos et al. A platform for difusion interactive multimedia content: an approach focused on iptv system and broadcasting digital television system. In: *EATIS '08: Proceedings of the 2008 Euro American Conference on Telematics and Information Systems*. New York, NY, USA: ACM, 2008. p. 1–5. ISBN 978-1-59593-988-3.

SANTOS JUNIOR, J. B. dos et al. Prototyping Interactive Digital Television Programs Using Java and XML: A Platform for Developers and Viewers. *EuroITV2007. Amsterdam, The Netherlands. Proceedings of EuroITV*, 2007.

SANTOS JUNIOR, J. B. dos et al. Canal de Retorno em Televisão Digital Interativa: Estratégias para Prototipação de Aplicações Usando um Provedor de Serviços Interativos. *WebMedia*, 2009.

SANTOS JUNIOR, J. B. dos et al. Trends on building interactive applications in the brazilian digital televison system. *7th Annual IEEE Consumer Communications and Networking Conference*, Las Vegas, CA, USA, January 2010.

SOARES, L. F. G.; BARBOSA, S. D. J. *TV digital interativa no Brasil se faz com Ginga: Fundamentos, Padrões, Autoria Declarativa e Usabilidade*. Rio de Janeiro, RJ: Editora PUC-Rio, 2008. 105-174 p. Capítulo 3 (Não Publicado).

SOARES, L. F. G.; FILHO, G. L. de S. Interactive Television in Brazil: System Software and the Digital Divide. *Proceedings of the EuroITV 2007 Conference*, 2007.

SOARES, L. F. G.; RODRIGUES, R. F.; MORENO, M. F. Ginga-NCL: The Declarative Environment of the Brazilian Digital TV System. *Journal of the Brazilian Computer Society*, v. 13, p. 37–46, 2007.

SOUZA, G. L. F.; LEITE, L. E. C.; BATISTA, C. E. C. F. Ginga-J: The Procedural Middleware for the Brazilian Digital TV System. *Journal of the Brazilian Computer Society*, v. 13, p. 47–56, 2007.

Sun Microsystems. 2009. Sun JavaDTV. Disponível em: http://java.sun.com/javame/technology/javatv/. Acesso em dezembro de 2009.

Sun Microsystems. *Sun Java ME Technology. Disponível em http://java.sun.com/javame/technology/. Acesso em janeiro de 2009.* 2009.

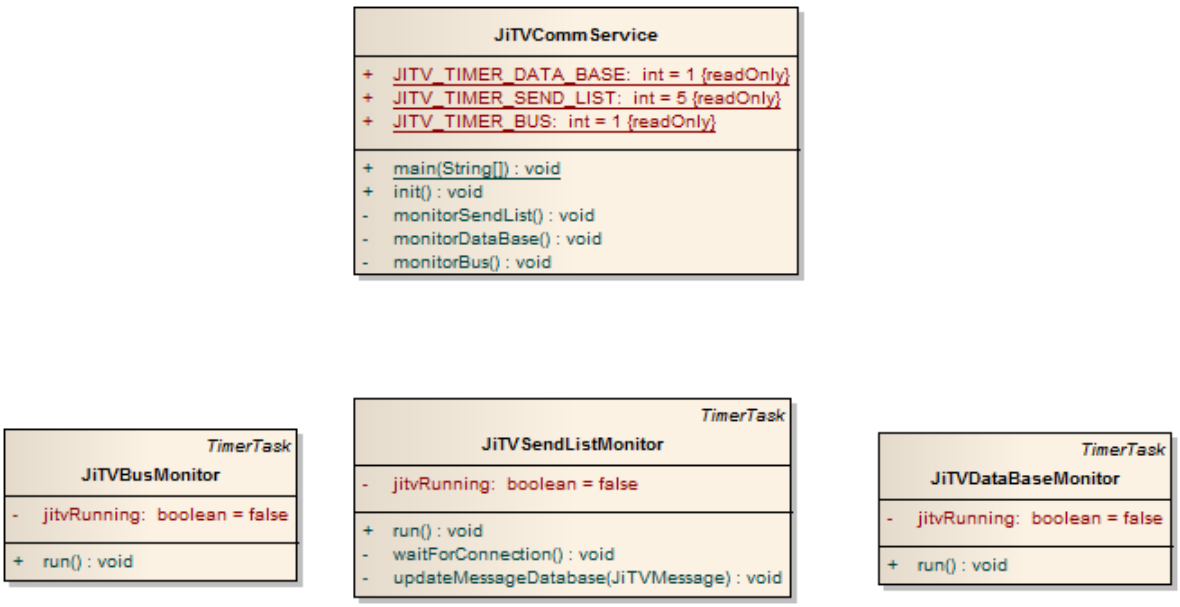
Sun Microsystems. *Sun JavaTV: Java Technology in Digital TV. Disponível em http://java.sun.com/products/javatv/. Acesso em junho de 2009.* 2009.

VRBA, V.; CVRK, L.; SYKORA, M. Framework for digital TV applications. In: *ICNICONSMCL '06: Proceedings of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies*. Washington, DC, USA: IEEE Computer Society, 2006. p. 184. ISBN 0-7695-2552-0.

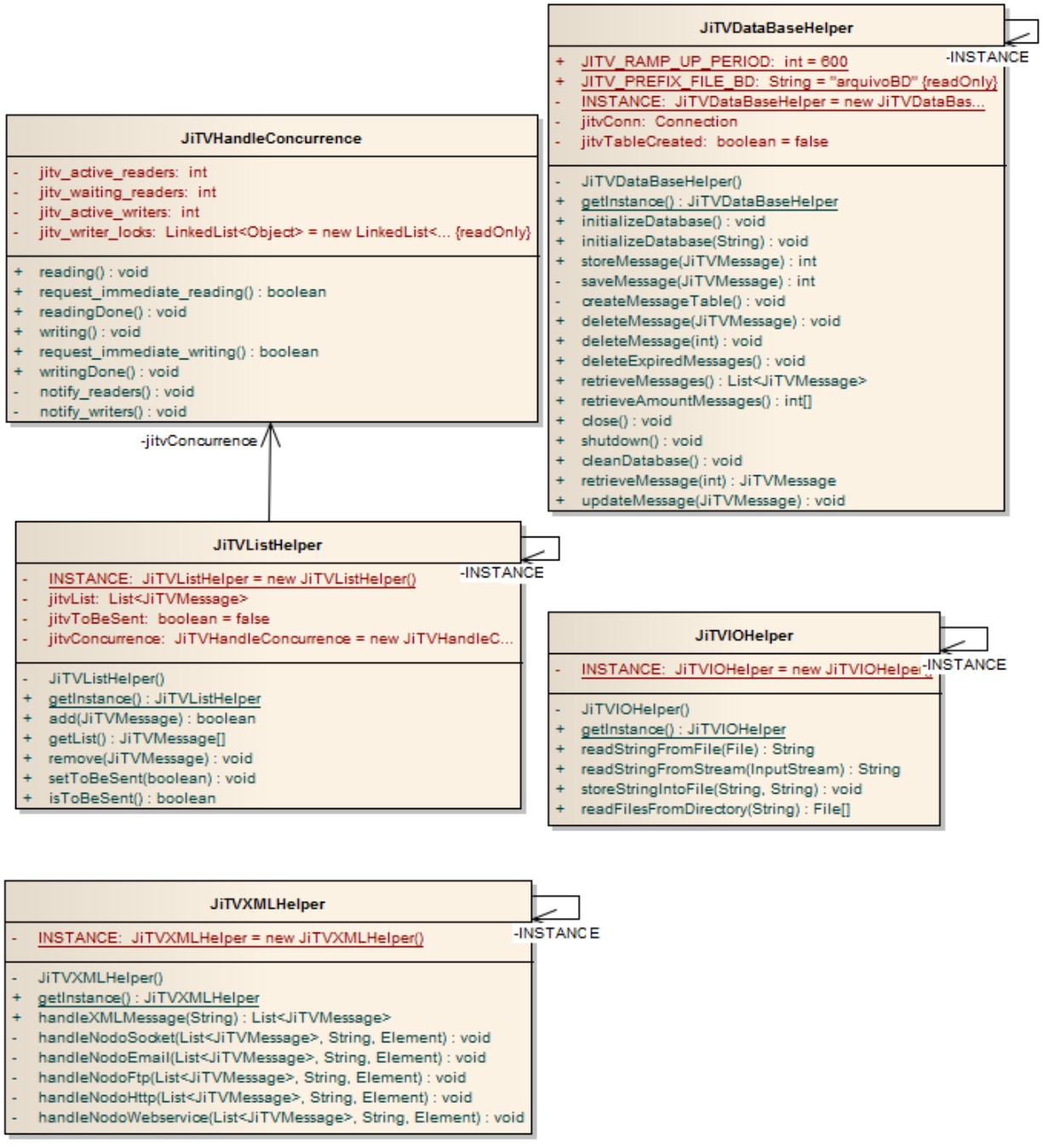
YANG, C.-S. et al. Personalized idtv program in multimedia home platform. In: *AINAW '08: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications - Workshops*. Washington, DC, USA: IEEE Computer Society, 2008. p. 473–476. ISBN 978-0-7695-3096-3.

APÊNDICE A – DIAGRAMAS DE CLASSES

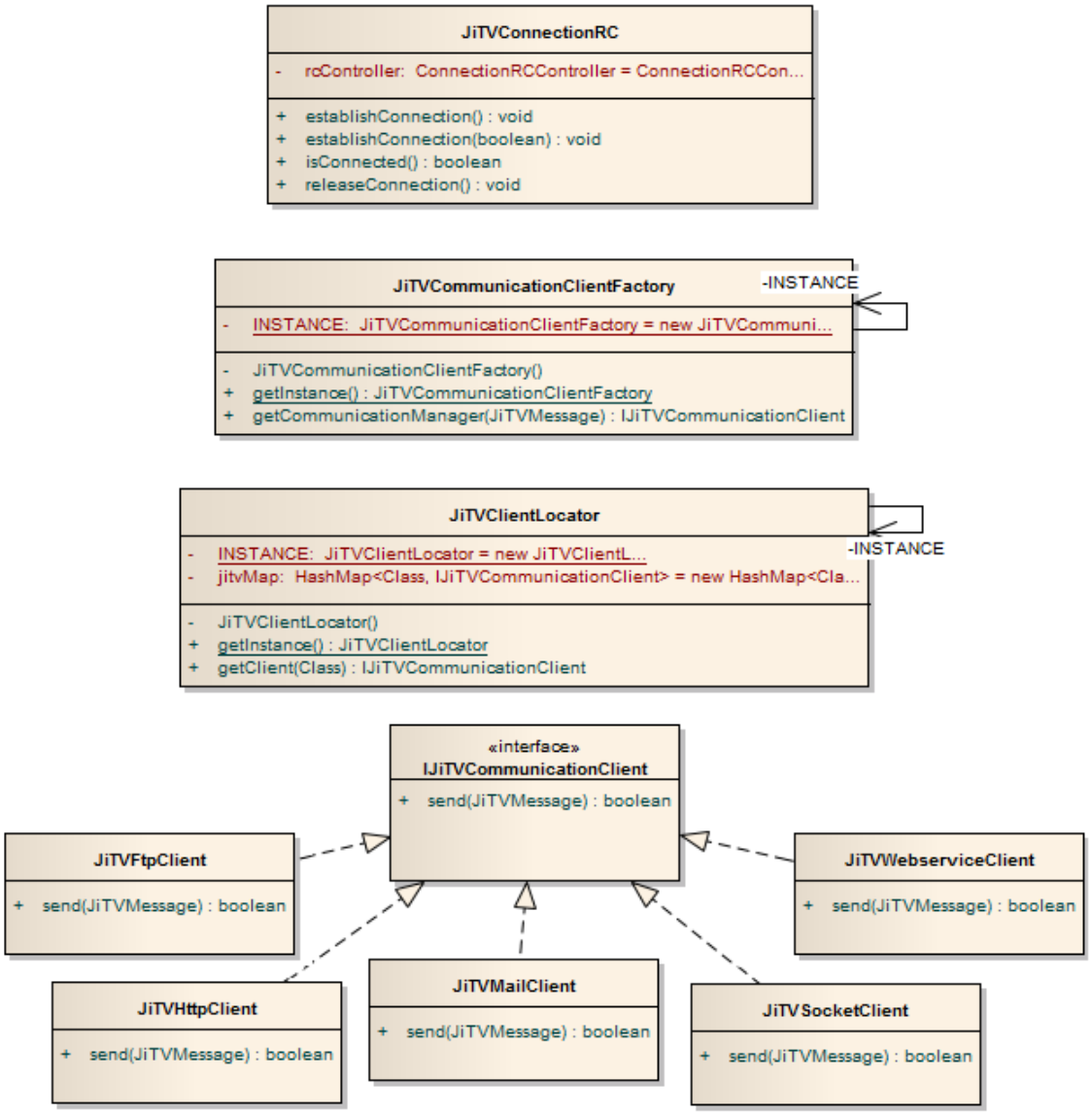
Pacote: **br.pucminas.jitv.commservice**



Pacote: **br.pucminas.jitv.commservice.helper**

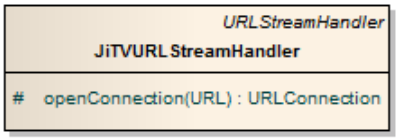


Pacote: **br.pucminas.jitv.commservice.services**

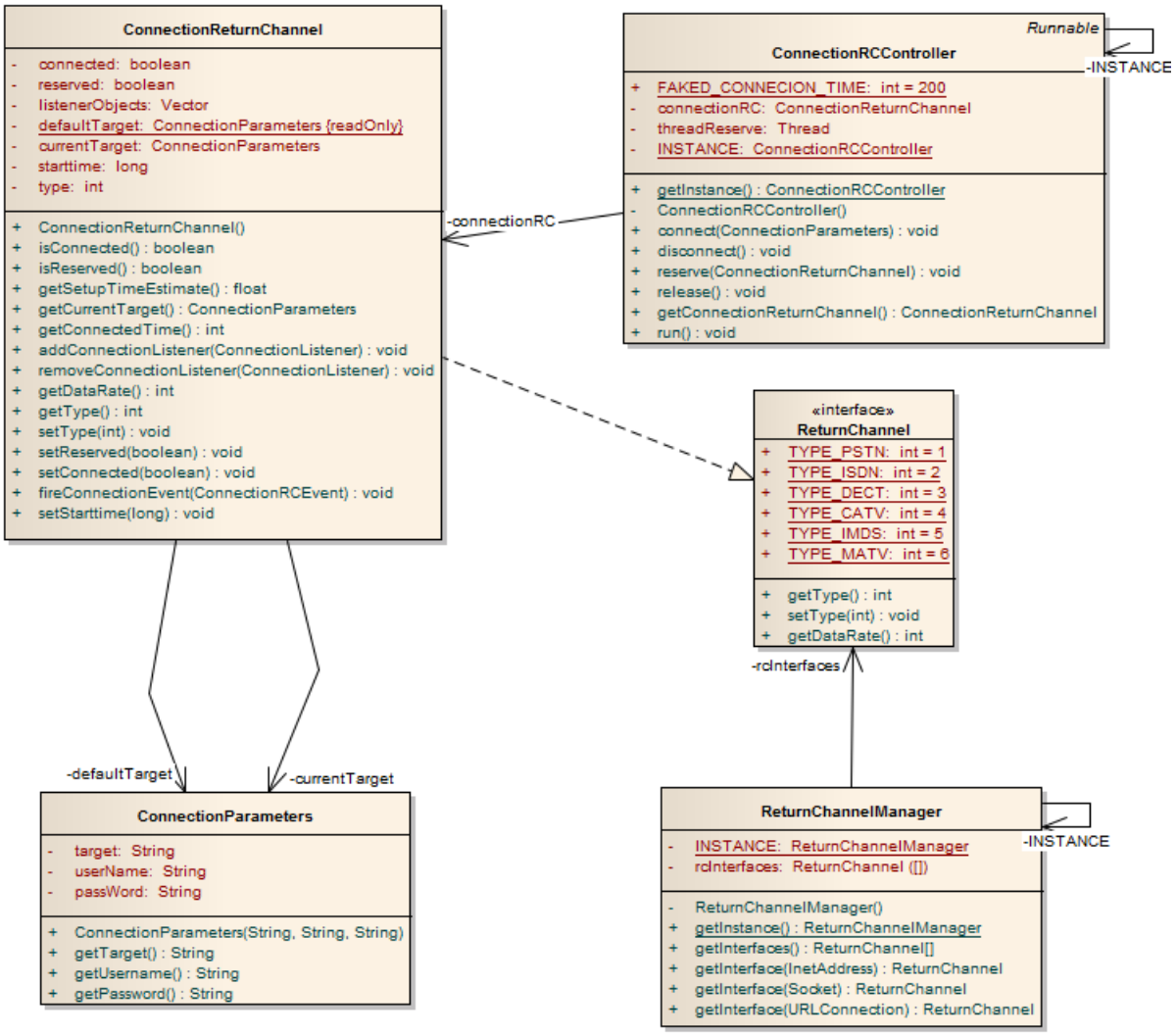


Pacote: **br.pucminas.jitv.commservice.vo**

Pacote: **br.pucminas.jitv.net**

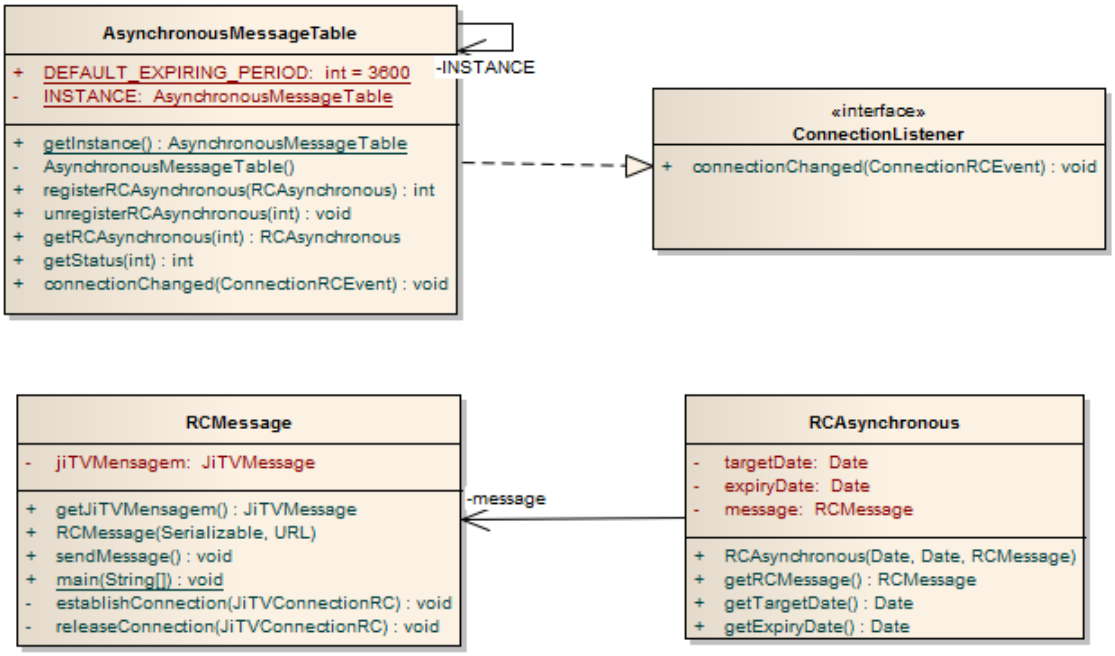


Pacote: **br.pucminas.jitv.net.rc** (conexão)

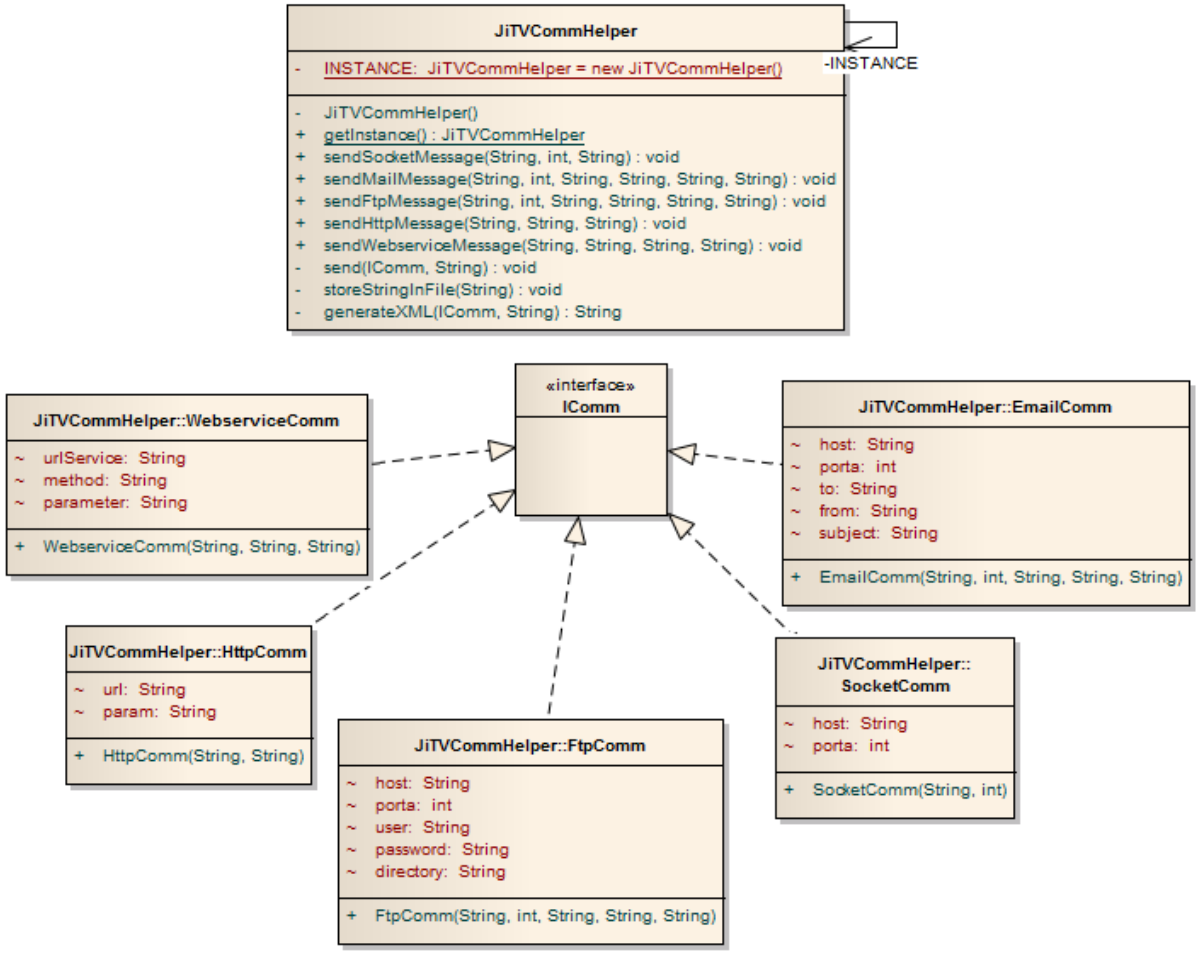


Pacote: **br.pucminas.jitv.net.rc** (exceções e eventos)

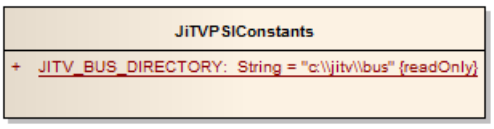
Pacote: **br.pucminas.jitv.net.rc** (mensagens)



Pacote: **br.pucminas.jitv.utils.comm**



Pacote: **br.pucminas.jitv.utils.io**



Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo