UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

PROPOSIÇÃO DE UM SIMULADOR GINGA-NCL PARA DISPOSITIVOS PORTÁTEIS

FÁBIO GOMES DE SOUZA

MANAUS
2012

UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

FÁBIO GOMES DE SOUZA

PROPOSIÇÃO DE UM SIMULADOR GINGA-NCL PARA DISPOSITIVOS PORTÁTEIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr.-Ing. Vicente Ferreira de Lucena Júnior

MANAUS
2012

Ficha Catalográfica
(Catalogação realizada pela Biblioteca Central da UFAM)

Souza, Fábio Gomes de

S729p Proposição de um simulador Ginga-NCL para dispositivos portáteis/ Fábio Gomes de Souza. - Manaus: UFAM, 2012.
100 f.; il. color.

Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Amazonas, 2012.
Orientador: Prof. Dr. Vicente Ferreira de Lucena Júnior

1. Processamento de imagens – Técnicas digitais 2. Receptor portátil 3. Linguagem de programação I. Lucena, Júnior Vicente Ferreira de (Orient.) II. Universidade Federal do Amazonas III. Título

CDU 004.932(043.3)

FÁBIO GOMES DE SOUZA

PROPOSIÇÃO DE UM SIMULADOR GINGA-NCL PARA DISPOSITIVOS PORTÁTEIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica na área de concentração Controle e Automação de Sistemas.

Aprovado em 11 de janeiro de 2012.

BANCA EXAMINADORA

Prof. Dr.- Ing Vicente Ferreira de Lucena Junior

Universidade Federal do Amazonas- UFAM

Prof. Dr. Waldir Sabino da Silva Junior

Universidade Federal do Amazonas- UFAM

Prof. Dr. Raimundo da Silva Barreto

Universidade Federal do Amazonas- UFAM

Aos meus filhos,
à minha esposa,
e aos meus pais.

AGRADECIMENTOS

Ao Senhor Deus por ter me mantido vivo, e ter me dado forças para a realização deste trabalho.

Aos meus filhos Thiago Souza e Jennifer Souza pela compreensão de minha ausência parcial durante a execução deste trabalho.

A minha esposa Marjorie Souza por ter me dado forças para continuar quando pensei em desistir.

Aos meus pais e irmãos por sempre acreditarem em minha capacidade.

Ao meu orientador e amigo Prof. Dr.-Ing. Vicente Ferreira de Lucena Júnior por me motivar nos momentos de dificuldades.

Ao Instituto Nokia de Tecnologia por prover um ambiente de trabalho agradável e por acreditar em tecnologias nacionais.

Ao CETELI e CAPES.

RESUMO

O desenvolvimento de conteúdo interativo para sistemas de TV digital envolve um processo que requer o uso de diversas ferramentas; dentre as quais se destacam: ferramentas de edição, multiplexação, e apresentação de conteúdo interativo. Ao desenvolver aplicações interativas destinadas a dispositivos portáteis, tais como telefones celulares, é importante levar em consideração características particulares destes ambientes de apresentação, as quais podem variar entre dimensões de telas distintas, mecanismos de entrada de dados no sistema, suporte a canal de interatividade, dentre outros. A pesquisa realizada neste trabalho concentrou-se em ambientes de apresentação de aplicações interativas destinadas a receptores portáteis em conformidade com o Sistema Brasileiro de TV Digital. O simulador proposto, Simulador Ginga-NCL para Dispositivos Portáteis (SGDP), contempla funcionalidades essenciais, necessárias para simulação de um ambiente de apresentação de aplicações interativas. Sintonia do sinal de TV oneseg, extração de conteúdos interativos do fluxo de transporte, interpretação de scripts Lua, e controle de apresentação de aplicações NCL, são exemplos de funcionalidades essenciais contempladas pelo SGDP. Adicionalmente, o simulador proposto registra informações sobre o processamento, e apresentação, de conteúdos interativos para análise posterior por desenvolvedores. A plataforma Symbian foi escolhida para implantação do SGDP, e testes sistêmicos foram realizados para validação da ferramenta.

Palavras-chave: TV digital, Ginga, interatividade, receptor portátil, oneseg.

ABSTRACT

The development of interactive content for digital TV systems involves a process that requires the use of several tools, among which stand out tools for editing, multiplexing, and presenting interactive content. When developing interactive applications for portable devices, such as cell phones, it is important to consider unique characteristics of those presentation environments, which may vary between different screen sizes, different data input mechanisms, support for interactive channel, among others. The research conducted in this study focused on presentation environments for interactive applications designed to run on portable receivers in accordance with the Brazilian Digital TV System. The proposed simulator, *Simulador Ginga-NCL para Dispositivos Portáteis* (SGDP) -- Ginga-NCL Simulator for Handhelds Devices -- includes essential features necessary for simulating a interactive application presentation environment. Tuning oneseg TV signal, extraction of interactive content from transport stream, Lua scripts interpretation, and NCL applications presentation control, are examples of essential features contemplated by SGDP. Additionally, the proposed simulator logs information about the processing and presentation of interactive content for further analysis by developers. The Symbian platform was chosen for implementation of SGDP, and systemic tests were performed to validate the tool.

Keywords: Digital TV, Ginga, interactivity, portable receiver, oneseg.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A/V	*Audio and Video*
AI	Aplicação Interativa
AIN	Aplicação Interativa NCL
API	*Application Programming Interface*
bps	*bits per second*
DOM	*Document Object Model*
DSM-CC	Digital Storage Media – Command and Control
fps	*frames per second*
GDF	Ginga-NCL para Dispositivos Fixos
GDP	Ginga-NCL para Dispositivos Portáteis
GPLv2	*GNU General Public License version 2*
HTML	*HyperText Markup Language*
INdT	Instituto Nokia de Tecnologia
IRGDF	Implementação de Referência Ginga-NCL para Dispositivos Fixos
IRGDP	Implementação de Referência Ginga-NCL para Dispositivos Portáteis
ISDB	*International Standard for Digital Broadcasting*
ISDB-T	*ISDB Terrestrial*
ITU	*International Telecommunication Union*
Kbps	*Kilo bits per second*
MB	*Mega Bytes*
MIME	*Multipurpose Internet Mail Extensions*,
MMF	*MultiMedia Framework*
MPEG	*Moving Picture Experts Group*
NCL	*Nested Context Language*
NCM	Nested Context Model
OS	*Operating System*
PES	*Packetized Elementary Stream*
PFD	Processador de Fluxo de Transporte
POSIX	*Portable Operating System Interface*

PPGEE	Programa de Pós Graduação em Engenharia Elétrica
PUC-Rio	Pontifícia Universidade Católica do Rio de Janeiro
RAM	*Random Access Memory*
ROM	*Read OnlyMmemory*
SAX	*Simple API for XML*
SBTVD	Sistema Brasileiro de TV Digital
SGDP	Simulador Ginga para Dispositivos Portáteis
SI	*Service Information*
SIM	*Subscriber Identity Module*
SMS	*Short Message Service*
SSD	Simulador de Sinal Digital
STB	*Set-Top Box*
STL	*Standard Template Library*
TS	*Transport Stream*
TVD	TV Digital
XHTML	*eXtensible HyperText Markup Language*
XML	*eXtensible Markup Language*

SUMÁRIO

1. Introdução

Em dezembro de 2007 foi realizada no Brasil a primeira transmissão de sinal do Sistema Brasileiro de TV Digital (SBTVD). O novo sistema possibilita que telespectadores usuais interajam com programas televisivos transmitidos pelas emissoras de TV. Para possibilitar a interatividade, foi desenvolvida por pesquisados brasileiros uma especificação de *middleware* aberta denominada Ginga. O *middleware* Ginga é uma especificação de camada de software responsável por definir o modo de processamento e apresentação de programas de TV em terminais receptores (televisores, *set-top boxes*, receptores portáteis, etc.). Esta camada de software independe de plataforma de hardware, e garante que aplicações interativas tenham a mesma apresentação em diversos dispositivos distintos.

Baseado no padrão de TV digital japonês, porém com adição de suporte a compressão de vídeo H.264, o sistema SBTVD possui uma característica que permite a inclusão digital, aos telespectadores, através de aplicações interativas transmitidas gratuitamente pelas emissoras de TV digital. A adoção do SBTVD fez surgir um novo nicho tecnológico, onde a demanda por novos serviços relacionados à área TV digital tem aumentado consideravelmente. Adicionalmente, vários segmentos foram afetados de forma positiva com o advento da TV digital no Brasil, dentre eles destacam-se:

- **Laboratórios de pesquisa em universidades do país**: a academia contribuiu significativamente para o padrão brasileiro de TV digital, principalmente no que diz respeito à interatividade, pois as linguagens de programação (NCL e Lua), desenvolvidas por pesquisadores brasileiros, foram incorporadas ao padrão de TV nacional.
- **Emissoras de TV:** as emissoras de TV estão substituindo seus equipamentos analógicos por equipamentos digitais sofisticados e robustos, capazes de suprir as novas demandas dessa tecnologia digital. Devido à alta definição das imagens captadas pelas câmeras digitais, os equipamentos de transmissão e armazenagem de conteúdo, dentre outros, terão que ser substituídos. Com

isso, surge a demanda por profissionais capacitados e qualificados para operar estes equipamentos.

- **Indústria de terminais receptores: o** padrão brasileiro de TV digital proporcionou um crescimento incrível em investimentos financeiros nas áreas de pesquisa e desenvolvimento no país. Fabricantes de terminais receptores tiveram que investir na capacitação de mão-de-obra nacional.
- **Provedores de conteúdo interativo:** com o advento da interatividade, agências de publicidade têm a possibilidade de enriquecer suas propagandas televisionadas, fazendo com que telespectadores participem de suas propagandas por meio de conteúdo interativo transmitido pelas emissoras de TV. Para atender a esta nova forma de propagandas, provedores de conteúdo interativo necessitam de mão-de-obra qualificada, gerando demanda por profissionais qualificados nesta área.
- **Operadoras de telefonia:** atualmente o meio de interatividade mais comum utilizado por emissoras de TV é através de mensagens de texto SMS (*Short Message Service* – ou Serviço de Mensagem Curta), pois telespectadores concorrem a prêmios, e são tarifados a cada mensagem enviada. O Modelo atual gera tráfego de dados através de serviços SMS e consequentemente garante receitas às operadoras de telefonia móvel no país. Com o advento da interatividade em receptores portáteis, é possível encapsular as mensagens SMS em ações a serem executadas por aplicações interativas; pois usuários de TV digital não necessitarão mais executar vários passos para concorrer aos concursos oferecidos pelas emissoras de TV. Este novo modelo pode estimular aos usuários interagirem com a programação televisionada de forma mais frequente; neste cenário o trafego de mensagens SMS pode crescer, gerando uma receita ainda maior às operadoras de telefonia móvel.

As subseções a seguir apresentam a motivação para a realização deste trabalho de pesquisa, bem como a identificação do problema a ser solucionado. Adicionalmente, os objetivos específicos são elencados e a metodologia de trabalho é apresentada.

1.1 Motivação

Diferentemente de terminais receptores fixos, terminais portáteis têm se mostrado bastante atrativos quando se trata de interatividade com a programação das emissoras de TV. Segundo TELECO (2012), dados da Agência Nacional de Telecomunicações (ANATEL) indicam que o Brasil terminou o mês de fevereiro de 2012 com 247,6 milhões de celulares no mercado, mostrando que o uso de aparelhos celulares vem crescendo nos últimos anos. Dentre os principais fatores para o crescimento destacam-se a redução dos preços dos dispositivos, e os planos de dados oferecidos pelas operadoras de telefonia móvel no país (IAB 2011). Um dos fatores que favorecem o uso de receptores móveis como terminais de TV interativa está relacionado ao fato destes dispositivos possuírem uma infraestrutura pronta para o canal de interatividade, principalmente através de serviços SMS e acesso à Internet, gerando receita tanto para as emissoras de TV quanto para as operadoras de telefonia móvel.

Aplicações interativas, de acordo com o padrão brasileiro de TV digital (ABNT-N06, 2007) podem ser desenvolvidas seguindo dois paradigmas de programação:

- Paradigma declarativo, através da linguagem NCL.
- Paradigma imperativo (ou procedural), através da linguagem Java.

Para receptores fixos, o *middleware* Ginga deve obrigatoriamente possuir os dois ambientes de exibição de aplicações interativas. Já em terminais portáteis, somente o ambiente declarativo é obrigatório, sendo o imperativo opcional (ABNT-N06, 2007).

Embora existam ambientes para edição e apresentação de aplicações interativas NCL, frutos de trabalhos acadêmicos como a ferramenta Composer (GUIMARÃES, 2007) e Ginga-NCL (SOARES *et al*, 2007), estes ambientes são dedicados ao desenvolvimento de aplicações interativas para receptores fixos. CRUZ (2008) apresenta em sua dissertação de mestrado a primeira implementação de referência para dispositivos portáteis. Porém, em seu trabalho acadêmico, módulos obrigatórios que compõe o middleware Ginga, segundo o

padrão brasileiro de TV digital (ABNT-N06, 2007), não foram contemplados devido a limitações de API de acesso ao sistema da plataforma utilizada (CRUZ, 2008).

Diferentemente de conteúdos interativos destinados a receptores fixos, desenvolvedores de aplicações interativas para receptores portáteis devem levar em consideração uma série de características adicionais durante a concepção de suas obras (YAMAKAMI 2009), tais como: resoluções de telas distintas; limitação de recursos de hardware (processador, memória, etc.); usabilidade diferente de TV convencional; canal de interatividade; dentre outros. Portanto, há a necessidade de um ambiente de apresentação, onde o autor de conteúdo interativo tenha a possibilidade de verificar o comportamento de suas obras, levando em consideração essas características.

1.2 Problema

Apesar de contribuições significativas realizadas por CRUZ (2008) em seu trabalho de pesquisa, módulos essenciais do middleware de interatividade Ginga-NCL não foram contemplados em sua implementação de referência, e foram deixados como trabalhos futuros. Dentre os módulos destacam-se: (1) Sintonizador Onseg – responsável por sintonizar uma frequência de sinal de TV; (2) Processador de Dados – encarregado de extrair o conteúdo interativo a partir do fluxo de transporte entregue pelo Sintonizador Oneseg; (3) Máquina Lua – responsável por interpretar scripts da linguagem Lua.

A ausência dos módulos (1) e (2) no middleware torna impraticável a sintonia de um canal, bem como a extração da aplicação interativa para apresentação no terminal receptor. Por ser responsável pela interpretação de scripts Lua, a ausência de (3) restringe o desenvolvedor de conteúdo interativo ao uso apenas de recursos procedurais da linguagem NCL (*Nested Context Language*)

Ao desenvolver aplicações NCL, com foco em terminais de TV portáteis, é importante que autores de conteúdo interativo levem em consideração

particularidades características desta classe de dispositivos. Terminais portáteis possuem dimensões de telas distintas, variando tanto em altura quanto em largura, possibilitando que aplicações interativas sejam apresentadas em modo retrato (orientação de tela vertical) ou modo paisagem (orientação de tela horizontal). Cabe ao autor do documento interativo indicar a qual orientação de tela sua aplicação melhor se apresenta, ou criar leiautes que se adequem às dimensões de tela dos terminais receptores. Dessa forma, desenvolvedores de conteúdos interativos devem se preocupar com a apresentação de suas aplicações nas diversas resoluções de tela existentes. A Figura 1 exibe alguns modelos de terminais receptores portáteis existentes no mercado brasileiro.

Aplicações interativas desenvolvidas para serem executadas em terminais portáteis devem ter uma atenção especial devido aos recursos limitados de hardware, característicos destes terminais. Embora a especificação Ginga recomende os recursos de hardware mínimos necessários para apresentação de aplicações interativas, é importante evitar uso desnecessário de recursos avançados da linguagem, bem como analisar o desempenho da apresentação do documento interativo num ambiente real, ou através de simulações, antes de sua transmissão oficial pelo sinal digital.

Figura 1: Diversidade de terminais receptores portáteis

Outro fator que deve ser levado em consideração está relacionado à usabilidade de aplicações interativas em receptores portáteis. Para receptores fixos, a entrada de dados no sistema pelo usuário é realizada, principalmente,

através de um controle remoto, com o qual telespectadores já estão acostumados a utilizar. Porém, em terminais portáteis a entrada de dados pode ser feita de formas distintas, como por exemplo:

- Através de eventos gerados a partir de teclados físicos nos terminais receptores, semelhante ao mecanismo de entrada de dados dos telefones celulares Nokia N85 e LG GM630 apresentados na Figura 2.

Figura 2: Suporte a entrada de dados via teclado

- Por eventos de toque de tela no terminal receptor, semelhante aos mecanismos utilizados por celulares do tipo *touchscreen*; como os celulares STi CTV41 e LG Scarlet apresentados na Figura 3;

Figura 3: Celulares com entrada de dados via tela sensível a toque

- Ou ainda por ambas as formas apresentadas num mesmo terminal, como é o caso do celular Samsung V820 exibido na Figura 4.

Figura 4: Receptor com entrada de dados híbrido

Portanto, é importante que autores de aplicações interativas NCL desenvolvam suas aplicações levando em consideração estas formas de entrada de dados. Além dos problemas referentes ao mecanismo de entrada de dados, o autor de conteúdo interativo deve decidir o meio pelo qual sua aplicação irá se comunicar com o servidor de conteúdo interativo.

Muito se discute sobre o canal de interatividade para terminais receptores fixos (MANHÃES, 2005) (ABNT-N07, 2008) (TELECO, 2010); porém, ainda não se chegou a um consenso sobre qual é a melhor opção. Já para receptores portáteis, principalmente no caso de telefones celulares, toda infraestrutura está pronta para uso, pois o canal de interatividade pode ser implantado através serviços como SMS, ou acesso à Internet, bastando que os usuários possuam um cartão SIM no terminal receptor. As emissoras de TV do Brasil já utilizam o serviço SMS como forma de canal de interatividade; porém, no cenário da TV analógica, os telespectadores necessitam redigir um texto a ser enviado ao servidor, bem como digitar o número para o qual se deve enviar a mensagem. Para automatizar este processo, é importante que projetistas de aplicações interativas elaborem suas obras de forma a encapsular os procedimentos de interação.

Outra característica importante quando se desenvolve aplicações interativas para TV digital está relacionada ao tempo decorrido para baixar os conteúdos interativos no terminal receptor. Esse fato é ainda mais agravante quando se trata de TV digital portátil, onde a largura de banda destinada ao canal *oneseg* é bastante limitada, chegando a 450 Kbps para a programação como um todo, incluindo áudio, vídeo, e dados.

Durante a fase de desenvolvimento de aplicações computacionais, sejam elas destinadas a computadores pessoais, servidores, sistemas embarcados, dentre outros, um mecanismo bastante utilizados por desenvolvedores de programas está relacionado à utilização de registros de execução (ou *logs* de execução), onde é possível verificar se determinados trechos de códigos são alcançados durante a execução do programa. O desenvolvimento de aplicações interativas NCL não foge a esta regra; portanto terminais receptores de sinal

digital de TV devem prover meios de registros e extração de informações sobre execução de aplicações.

Conforme apresentado nesta seção, os módulos do middleware Ginga-NCL não contemplados por CRUZ (2008) são fundamentais para a extração do conteúdo interativo do fluxo de transporte, bem como para a apresentação adequada deste conteúdo no terminal receptor. Adicionalmente, esta seção destacou características importantes referentes ao desenvolvimento de aplicações interativas destinadas a dispositivos portáteis. Para que os mais diversos cenários sejam contemplados, se faz necessária uma ferramenta capaz de solucionar os problemas aqui apresentados. Essa ferramenta é alvo de pesquisa do trabalho proposto.

1.3 Objetivos

Este trabalho tem como objetivo proporcionar um ambiente de simulação de TV digital interativa para terminais portáteis compatível com padrão brasileiro. O ambiente proposto visa prover uma ferramenta para extração e apresentação de conteúdos interativos provenientes do sinal digital de TV, e dessa forma auxiliar autores de aplicações NCL a identificarem possíveis problemas, ou não conformidades, em suas aplicações NCL. O trabalho possui os seguintes objetivos específicos:

- Recomendar um simulador de recepção de sinal digital *oneseg* com o intuito de prover um mecanismo de identificação do tempo decorrido para baixar o conteúdo interativo no terminal receptor.

- Disponibilizar um ambiente de apresentação de aplicações interativas NCL para terminais portáteis, de forma que desenvolvedores de conteúdos interativos possam visualizar suas obras na tela do ambiente de simulação.

- Prover suporte a ambos os tipos de orientações de tela (modo retrato, e modo paisagem) de forma que possa ser verificada a disposição dos

componentes gráficos na tela do simulador conforme orientação definida pelo autor de conteúdo interativo.

- Prover funcionalidades de canal de interatividade através de serviços de rede de dados, como SMS ou TCP/IP.
- Propor um mecanismo de depuração de aplicações interativas NCL, onde desenvolvedores NCL possam identificar problemas em seu código de forma ágil.
- Prover mecanismos para o tratamento de comandos de edição Ginga.
- Realizar testes sistêmicos no simulador para validação de suas funcionalidades.

1.4 Organização dos Capítulos

O Capítulo 2 discorre sobre conceitos básicos relacionados ao sistema de TV digital brasileiro. Neste capítulo são apresentadas tecnologias de TV digital interativa, com um breve resumo sobre a evolução dos sistemas de TV, desde a primeira transmissão em preto e branco até a inclusão do suporte a interatividade, bem como o suporte e a alta definição de vídeos.

No Capítulo 3 apresenta-se os trabalhos relacionados ao ambiente de simulação proposto. Iniciando a partir da primeira proposta de middleware declarativo para o sistema SBTVD (MORENO, 2006) até sua adoção como parte integrante do middleware Ginga, apresentada por SOARES *et al* (2007). Este capítulo discorre também sobre a primeira implementação de referência do middleware declarativo Ginga-NCL para dispositivos portáteis (CRUZ, 2008). Ao término do capítulo uma breve comparação entre as soluções existentes é realizada, elencando as melhorias necessárias para apresentação de documentos NCL, principalmente os dedicados a terminais receptores portáteis.

A proposta do ambiente de simulação para dispositivos portáteis é apresentada no Capítulo 4. Este capítulo aborda o modelo de arquitetura do

simulador SGDP e apresenta soluções, através da modelagem de componentes de software, para processamento e renderização adequada de aplicações NCL em terminais portáteis. Detalhes sobre a implementação do simulador também são tratadas neste capítulo.

O Capítulo 5 discorre sobre os testes sistêmicos realizados para validação do simulador portátil. Nele, são apresentadas as aplicações NCL utilizadas para testar o simulador, bem como os dispositivos portáteis onde o simulador foi implantado.

No Capítulo 6 são apresentadas as considerações finais sobre o trabalho realizado, dando destaque para os problemas encontrados durante a elaboração deste trabalho, e elencando melhorias a serem realizadas no simulador SGDP.

2. Tecnologias Relacionadas

Neste capítulo são apresentados conceitos básicos para o entendimento do trabalho proposto. Dentre as principais tecnologias relacionadas destacam-se: TV digital interativa, sistema de TV digital ISDB-T, linguagens de programação NCL e Lua.

2.1 TV Digital Interativa

A televisão está em constante evolução tecnológica de forma a atender as necessidades sociais. O primeiro canal de TV a ser transmitido no mundo foi a BBC de Londres, fundada em 1936 (MONTEZ e BECKER 2005), desde então até o presente momento, a televisão passou por várias mudanças. A primeira mudança significativa foi a transmissão em cores em 1950, em seguida o número de canais disponíveis aos telespectadores, e, por conseguinte o surgimento do controle remoto para maior comodidade do telespectador ao mudar de canal. Este último veio a ser o primeiro componente digital integrado aos aparelhos de TV.

Um dos destaques consiste na digitalização de parte da produção do conteúdo televisivo, com introdução de câmeras e ilhas de edição. Uma vez digitalizado o conteúdo, o passo seguinte da evolução foi a transmissão dos fluxos de áudio e vídeo, e a recepção do sinal por aparelhos decodificadores (set-top-boxes). Já na década de 1980, as ilhas de edição digitais passaram a oferecer mais flexibilidade e maiores recursos aos editores. Os avanços mostravam ser possível também a transmissão digital, amplamente testada na década de 1990.

> *"*As pesquisas para a TV digital começaram no final da década de 1980 e se consolidaram na década de 1990, com o lançamento comercial dos dois primeiros padrões: o ATSC e o DVB, nos EUA e na Europa, respectivamente*"* (MONTEZ e BECKER 2005).

Surge então a era da TV digital, onde se destacam como vantagens a viabilidade de interação do usuário de TV com a programação televisionada. Com o advento da interatividade, o envio de informações deixou de ser unidirecional e o telespectador passou a trocar informações com a emissora de TV através de um único terminal. A Figura 5 ilustra a transição de um modelo de TV sem suporte ao canal de interatividade (analógico ou digital) para um modelo com suporte à interatividade integrada ao terminal receptor.

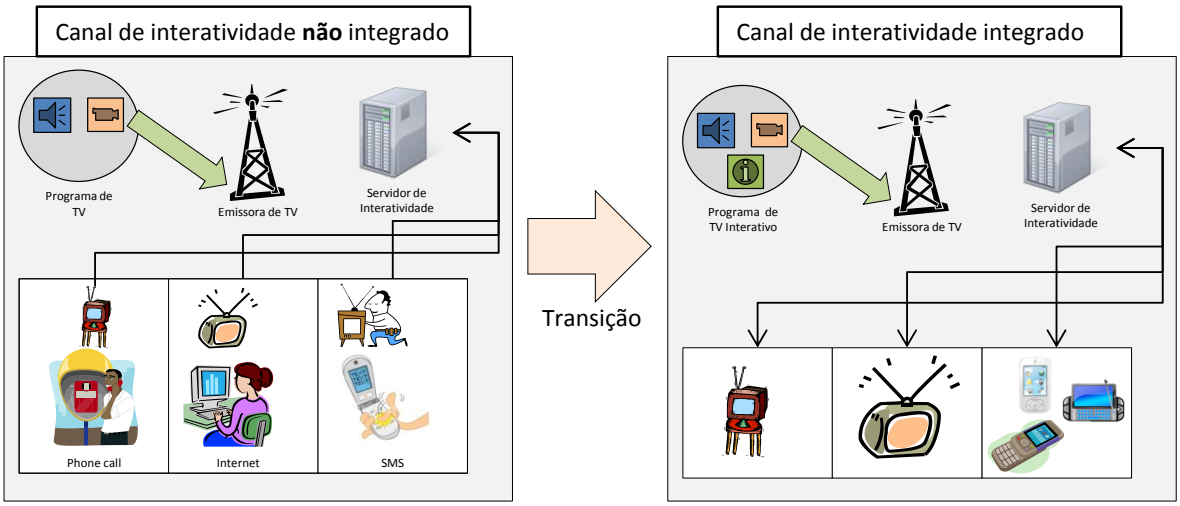

Figura 5: Transição do modelo interatividade

No cenário sem integração do canal de interatividade o usuário de TV necessita de um dispositivo adicional (telefone celular, computador, etc.) para interagir com a programação televisionada. Neste cenário, a programação de TV é assistida pelo telespectador, o qual segue as instruções de interação fornecidas pelo interlocutor. Os meios de comunicação para o canal de interatividade podem ser realizados de várias formas: através de chamadas telefônicas, pela Internet, ou envio de mensagens de texto SMS (*Short Message Service*). No cenário integrado, o próprio dispositivo receptor de sinal de TV é utilizado. O conteúdo interativo é apresentado na tela do terminal receptor, onde o usuário pode seguir as instruções contidas na aplicação interativa, e interagir de forma eficiente.

2.2 Sistema de TV Digital ISDB-T

Esforços em padronização das diversas camadas de tecnologias de TV digital foram gastos no decorrer de uma década (SCHWALB, 2003). Desde

modulação de sinal, encapsulamento de dados, tecnologias de codificação de vídeo, sincronização, acesso condicional, tecnologias de transmissão, até os padrões mais avançados que habilitam serviços interativos. Dentre os principais padrões de TV digital destacam-se: ATSC (*Advanced Television Systems Committee*); DVB (*Digital Video Broadcasting*); DBM (*Digital Multimedia Broadcasting*); e ISDB-T (*International Standard for Digital Broadcasting Terrestrial*). A Figura 6 exibe a distribuição da adoção de sistemas de TV digital no mundo.

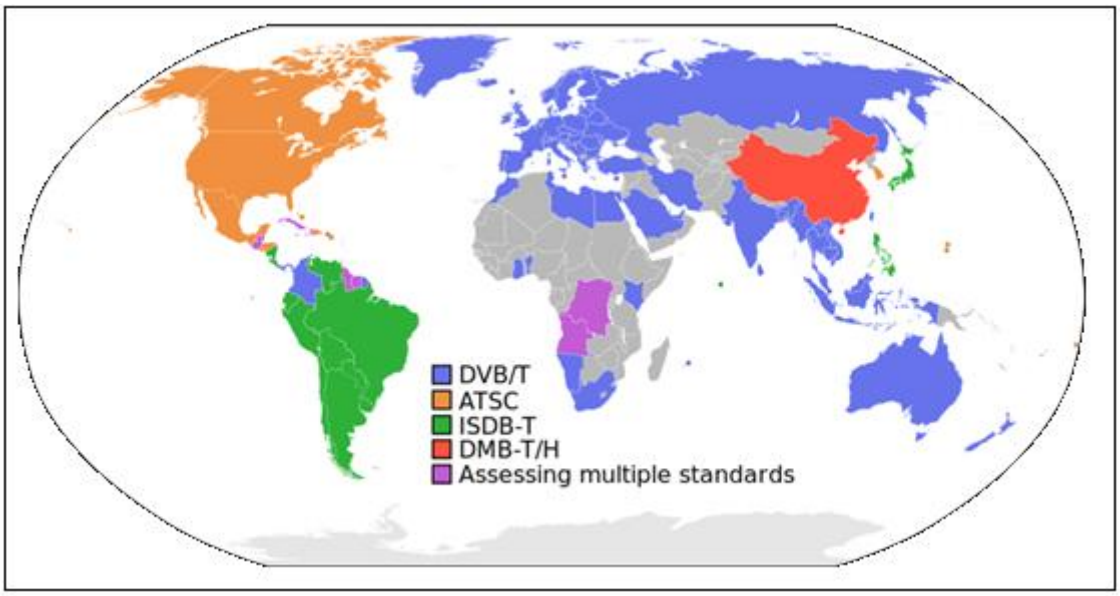

Figura 6: Mapa de adoção dos padrões de TV digital no mundo

FONTE: Extraído de http://www.wikipedia.org/ (Novembro de 2011)

O sistema ISDB-T é o mais recente dentre os sistemas de TV digital apresentados, e tem como principal diferença o suporte aos novos formatos de áudio e vídeo. Adicionalmente, no sistema ISDB-T, cada canal de TV possui um segmento (*oneseg*) da faixa de frequência dedicado à transmissão do sinal para receptores portáteis.

O sistema foi inicialmente padronizado e adotado pelo Japão, porém já está presente em quase toda América do Sul, em países da América Central, bem como nas Filipinas (DTV-BR, 2011). Após sua adoção pelo Brasil, o sistema ISDB-T sofreu modificações em suas especificações. Dentre as melhorias incorporadas ao padrão destaca-se o suporte a interatividade através do middleware Ginga. A Tabela 1 apresenta de forma resumida as principais

diferenças entre os padrões japonês e brasileiro em relação aos receptores portáteis.

	Japão	Brasil
Decodificação de Vídeo	H.264 AVC Baseline Level 1.2	H.264 AVC Baseline Level 1.3
Formato de Vídeo	QVGA 320x240 e 320 x180	SQVGA 160x120, 160 x90 QVGA 320x240 e 320 x180 CIF 352x288
Decodificação de Áudio	MPEG-2 AAC SBR	MPEG-4 HE AAC Nível 2 SBR + PS
Taxa de Quadros	15 quadros por segundo	5, 10, 12, 15, 24 e 30 quadros por segundo
Middleware de Interatividade	BML	Ginga-NCL
Transporte de Dados	DSM-CC Carrossel de Dados	DSM-CC Carrossel de Objetos

Tabela 1: Diferenças entre os perfis de receptores portáteis

2.3 Linguagens de Programação NCL e Lua

O middleware Ginga permite que conteúdos interativos para o padrão brasileiro de TV digital sejam desenvolvidos nas linguagens de programação NCL/Lua ou Java (SOARES e BARBOSA 2009). Porém, para terminais receptores portáteis, apenas a linguagem NCL é obrigatória (ABNT-N06, 2007). Aplicações NCL podem fazer uso de scripts Lua para prover maior poder de expressividade a autores de conteúdos interativos. As subseções a seguir apresentam as características das linguagens NCL e Lua.

2.3.1 Linguagem NCL

> "Criada no Laboratório TeleMídia da PUC-Rio, a linguagem NCL - Nested Context Language - é uma linguagem declarativa para autoria de

> documentos hipermídia baseados no modelo conceitual NCM - Nested Context Model.” (NCL, 2009)

NCL é uma linguagem declarativa baseada em XML (*eXtensible Markup Language*), e segue uma abordagem de modularização. O modelo da linguagem NCL tem uma separação rigorosa entre a estrutura que define o documento e seu conteúdo. NCL não visa apenas o suporte declarativo à interação do usuário, mas ao sincronismo espacial e temporal em sua forma mais geral, tratando a interação do usuário como um caso particular (SOARES e BARBOSA, 2009). Dentre os principais focos, ou áreas que atuação que a linguagem contempla, pode-se destacar:

- Suporte declarativo a adaptações de conteúdo e de formas de apresentação de conteúdo.
- Suporte declarativo a múltiplos dispositivos de exibição, interligados através de redes residenciais, ou de forma mais ampla.
- Edição e produção da aplicação em tempo de exibição, ou seja, ao vivo.

Com relação à área de TV digital, pode-se observar que as características da linguagem NCL se enquadram perfeitamente com a necessidade deste nicho tecnológico. Portanto a linguagem NCL se torna uma opção muito atraente para o desenvolvimento de conteúdos interativos. NCL, através de scripts Lua, provê ainda suporte a geração dinâmica de conteúdo, bem como capacidade computacional para processamento de algoritmos mais robustos.

Conforme mencionado anteriormente, o modelo da linguagem NCL tem uma separação bastante rigorosa entre o conteúdo e a estrutura que representam uma aplicação NCL. A linguagem não restringe a utilização de nenhum tipo de mídia, permitindo que objetos de imagem, vídeo, áudio, texto, e inclusive de execução (objeto Lua, por exemplo) sejam suportados; sendo de responsabilidade do formatador NCL a definição de objetos suportados em sua implementação.

O módulo de estrutura da linguagem NCL define a etiqueta `<ncl>` como elemento raiz, sendo `<head>` e `<body>` os elementos básicos. O elemento

<ncl> possui atributos para identificação da aplicação e indicação do perfil utilizado no documento NCL. No elemento <head> estão contidos os elementos referenciados pelo núcleo da aplicação, tais como as bases de: regiões, descritores, transições, conectores, e regras. Os elementos que formam o núcleo da aplicação estão contidos no elemento <body>. A Tabela 2 apresenta os elementos, atributos e conteúdos que definem a estrutura de documentos NCL (SOARES e BARBOSA, 2009).

Elemento	Atributo	Conteúdo
Ncl	id, title, xmlns	(head?, body?)
Head		(importedDocumentBase?, ruleBase?, transitionBase?, regionBase*, descriptorBase?, connectorBase?, meta*, metadata*)
Body	id	(port | property | media | context | switch | link | meta | metadata)?

Tabela 2: Elementos básicos da linguagem NCL 3.0

Conforme apresentado anteriormente, NCL é uma aplicação XML que segue uma abordagem de modularização, onde cada módulo da linguagem representa uma coleção de elementos e atributos XML relacionados semanticamente de forma a representar uma unidade funcional. Um perfil de linguagem representa a combinação de módulos coerentes. Ginga-NCL define dois perfis de linguagens para o padrão brasileiro de TV digital: EDTVProfile (*Enhanced Digital TV Profile* – Perfil Avançado de TV Digital) e BDTVProfile (*Basic Digital TV Profile* – Perfil Básico de TV Digital) (SOARES *et al*, 2007). O perfil NCL 3.0 completo, também chamado de perfil linguagem NCL 3.0 compreende todos os módulos NCL e fornece facilidades para a autoria declarativa de documentos NCL (ABNT-N06, 2007). A versão NCL 3.0 está particionada em quinze áreas funcionais, as quais estão subdivididas nos seguintes módulos:

- ***Structure*:** a área funcional *Structure* possui um único módulo denominado *Structure*, no qual a estrutura básica de um documento NCL é definida. Essa área define os elementos <ncl>, <head> e <body>.

- ***Layout*:** assim como a área funcional *Structure*, a área *Layout* possui apenas um módulo em sua definição, cujo nome é o mesmo da área: *Layout*. Esta área funcional especifica os elementos e atributos que definem a apresentação inicial dos objetos de mídias dentro de suas respectivas regiões.
- ***Component*:** a área funcional *Component* está particionada em dois módulos: *Media* e *Context*. O primeiro define os tipos básicos de objetos de mídia que pertencem ao documento NCL. O segundo é responsável pela definição de nós de contexto através de elementos `<context>`.
- ***Interfaces*:** a área funcional *Interfaces* permite a definição de interfaces de nós, os quais são utilizadas em relacionamentos com outras interfaces de nós. Esta área está particionada em quatro módulos: *MediaContentAnchor*, *CompositeNodeInterface*, *PropertyAnchor*, e *SwitchInterface*. O primeiro permite definições de âncoras de conteúdo para nós de mídia. *CompositeNodeInterface* habilita definições de portas para nós de composição. *PropertyAnchor* permite a definição de propriedades de nós como interfaces de nós. Por último, o módulo *SwitchInterface* habilita a definição de interfaces espaciais para elementos `<switch>`.
- ***Presentation Specification*:** tem um único módulo denominado *Descriptor*, cujo objetivo é especificar informações espaço-temporais necessárias para a apresentação de cada componente do documento NCL.
- ***Linking*:** define o módulo *Linking* responsável pela definição de elos que utilizam conectores.
- ***Connectors*:** está particionada em sete módulos básicos: *ConnectorCommonPart*, *ConnectorAssessmentExpression*, *ConnectorCausalExpression*, *CausalConnector*, *ConstraintConnector*, *ConnectorBase* e *CompositeConnector*. Os módulos são totalmente independentes dos demais módulos do perfil NCL 3.0, e formam o núcleo de uma nova linguagem de aplicação XML para definição de conectores, os quais podem ser utilizados para especificar relações de sincronização espaço-temporais, tratando relações de referência (de interação com usuário) como um caso particular de relações de sincronização temporal (ABNT-N06, 2007).

- ***Presentation Control*:** tem como objetivo especificar alternativas de conteúdo e apresentação para o documento. Está subdividida nos seguintes módulos: *TestRule*, *TestRuleUse*, *ContentControl* e *DescriptorControl*.
- ***Timing*:** tem um único módulo, *Timing*, que permite a definição de atributos temporais para componentes de um documento NCL.
- ***Reuse*:** a área funcional *Reuse* permite a reutilização de elementos NCL, e está particionada em três módulos: *Import*, *EntityReuse* e *ExtendedEntityReuse*. O módulo *Import* permite que uma base de entidades seja incorporada a outra base existente através do elemento `<importeBase>`, adicionalmente, um documento NCL pode ser importado através do elemento `<importNCL>`. Os módulos *EntityReuse* e *ExtenedEntityReuse* permitem que um elemento NCL seja reutilizado através dos atributos `refer` e `instance`, respectivamente.
- ***Navigational Key*:** possui apenas um módulo, o qual oferece as extensões necessárias para descrever as operações de movimentação de foco na tela.
- ***Animation*:** prover suporte ao desenho e movimento de objetos, e define apenas um módulo: *Animation*.
- ***Transition Effects*:** está particionada em dois módulos: *TransitionBase* e *Transition*. Ambos especificam um conjunto de efeitos de transição.
- ***Metainformation*:** composta pelo módulo *Metainformation*, contém informações sobre o conteúdo utilizado no documento NCL.

Todas as áreas funcionais apresentadas compõem ambos os perfis NCL para o padrão brasileiro de TV digital (EDTVProfile e BDTVProfile), com exceção das área funcionais *Animation*, *Transition Effects* e *Metainiformation*, as quais não pertencem ao perfil *BDTVProfile*.

A área funcional *Connectors* define as relações entre as entidades da linguagem NCL. As relações são baseadas em eventos, os quais seguem o modelo de máquina de estado apresentado na Figura 7. Os eventos NCL estão classificados da seguinte forma:

- **Eventos de apresentação**: definidos pela apresentação de um subconjunto das unidades de informação de um objeto de mídia, especificado em NCL pelo elemento `<area>`, ou pelo nó de mídia em si.
- **Eventos de seleção:** definidos pela seleção de um subconjunto das unidades de informação de um objeto de mídia em exibição, especificado na NCL pelo elemento `<area>`, ou pelo próprio nó de mídia.
- **Eventos de atribuição:** definidos pela atribuição de um valor a uma propriedade de um nó (representado por um elemento `<media>`, `<body>`, `<context>` ou `<switch>`), que deve obrigatoriamente ser declarado em um elemento `<property>`, filho do nó.
- **Eventos de composição:** definidos pela apresentação da estrutura de um nó de composição (`<body>`, `<context>` ou `<switch>`), e são utilizados para apresentar o mapa da composição (organização da composição).

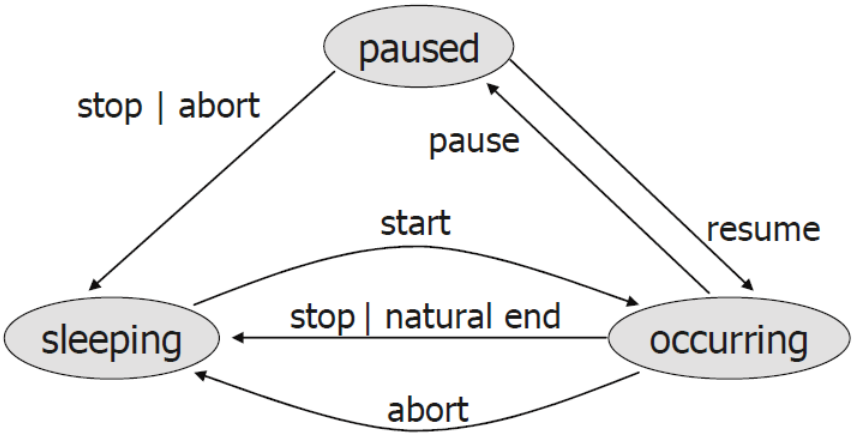

Figura 7: Máquina de estado de eventos da linguagem NCL 3.0

FONTE: Extraído de ABNT-N06 (2007)

2.3.2 Linguagem Lua

Linguagens declarativas não requerem habilidades especiais de programação, e NCL não é nenhuma exceção. Porém, linguagens com essas características deixam a desejar, principalmente quando há a necessidade em elaborar aplicações mais robustas, que fazem uso de desenhos gráficos,

processamento de entrada de dados, cálculos matemáticos, etc. Objetos NCLua fazem a integração entre a linguagem declarativa NCL e a linguagem imperativa Lua.

Lua provê um sistema de software funcional (Máquina Virtual Lua - MVL) capaz de interpretar programas escritos em sua linguagem. A máquina MVL é escrita em C ANSI, o que permite seu porte para diversos dispositivos, variando desde pequenos dispositivos até servidores *high-end* (EMMERICH, 2009).

> "Lua é uma linguagem de programação poderosa, rápida e leve, projetada para estender aplicações. Lua combina uma sintaxe simples para programação procedural com poderosas construções para descrição de dados baseadas em tabelas associativas e em semântica extensível" (SANT'ANNA et al, 2008).

Por ser uma linguagem de programação leve, e possível de ser estendida, Lua foi escolhida como linguagem de *script* para compor o *middleware* declarativo Ginga-NCL, e dessa forma, prover maior poder de expressividade aos desenvolvedores de conteúdos interativos para o padrão de TV digital brasileiro.

De forma análoga ao que ocorre em outras áreas tecnológicas que utilizam Lua em seu ambiente (na área de jogos eletrônicos, por exemplo), esta linguagem teve que ser estendida com novas funcionalidades para atender às necessidades do ambiente de TV digital brasileiro e se integrar à linguagem declarativa NCL. A integração entre as duas linguagens é possível através de *scripts* Lua encapsulados em objetos de mídia NCLua. Um *script* NCLua pode responder e processar entrada de dados provenientes de um controle remoto, por exemplo; ou ainda desenhar livremente dentro de uma região NCL a ele destinada.

Conforme apresentado na Subseção 2.3.1, a linguagem NCL tem como característica a separação distinta entre o conteúdo das mídias e a estrutura do documento que compõem a aplicação. Os objetos de mídias se relacionam através de suas âncoras, as quais por sua vez são classificadas em dois tipos: áreas e propriedades. As áreas definem porções de conteúdos que são exportadas como âncoras para o documento NCL, cabendo aos adaptadores de

mídia interpretar a semântica atribuída a elas. Porém, para objetos de mídias NCLua, a semântica não é definida por adaptadores de mídia, mas sim postergada para que possa ser interpretada, no momento adequado, pela máquina MVL conforme especificado pelo autor de cada *script*. As âncoras de propriedades definem pares do tipo chave/valor, os quais servem para parametrizar o comportamento dos adaptadores. Assim como nas âncoras de áreas, cabe aos adaptadores de mídia interpretar a semântica dada pelo perfil da linguagem, e no caso de NCLua, a semântica de uma âncora de propriedade é definida de pelo autor do conteúdo interativo.

NCLua estende as funcionalidades da linguagem Lua, e provê um conjunto de funcionalidades adicionais através de seus módulos (SANT'ANNA *et al*, 2008):

- ***Event:*** permite que objetos NCLua se comuniquem com o documento NCL e demais entidades externas, tais como canal de interatividade (eventos *sms* e *tcp*), dispositivos de entrada de dados (eventos *key* e *pointer*), dentre outros.
- ***Canvas:*** oferece interfaces de programação para desenhar primitivas gráficas e manipulação de imagens.
- ***Settings:*** exporta a tabela de configurações contendo variáveis definidas pelo documento NCL, e variáveis de ambiente.
- ***Persistent:*** permite que scripts NCLua armazenem, numa área restrita do middleware, dados para uso posterior.

2.4 Resumo sobre Tecnologias Relacionadas

Neste capítulo foram apresentadas, de forma sucinta, as principais tecnologias relacionadas ao simulador de TV digital para terminais receptores portáteis. A TV digital interativa abrange uma área bastante vasta. Porém, o trabalho proposto tem como foco apenas o sinal digital *oneseg* do sistema ISDB-T, dedicado a terminais portáteis.

O *middleware* Ginga (ABNT-N06, 2007) especifica dois subsistemas para processamento, e apresentação, de aplicações interativas: Ginga-NCL e Ginga-J.

O subsistema Ginga-NCL é item obrigatório para terminais receptores portáteis. Este subsistema é responsável por interpretar conteúdos interativos desenvolvidos na linguagem de programação NCL, com suporte a *scripts* Lua, apresentadas nas Seções 2.2 e 2.3, respectivamente. Portanto, um estudo aprofundado sobre essas tecnologias se faz necessário.

O capítulo a seguir discorre sobre os trabalhos relacionados, onde são apresentadas as implementações de referências dos *middlewares* Ginga-NCL para receptores fixos e portáteis.

3 Trabalhos Relacionados

Este capítulo discorre sobre os trabalhos relacionados ao Simulador Ginga-NCL para Dispositivos Portáteis (SGDP). A Seção 3.1 apresenta as características do *middleware* declarativo Ginga-NCL para Dispositivos Fixos (GDF), o qual teve como seu precursor o *middleware* Maestro apresentado por MORENO (2006) em sua dissertação de mestrado. A Seção 3.2 descreve o trabalho realizado por CRUZ *et al* (2008) em sua proposta de porte do middleware GDF para dispositivos portáteis. A última seção deste capítulo faz uma análise comparativa entre os trabalhos relacionados.

3.1 Ginga-NCL para Dispositivos Fixos (GDF)

Ginga-NCL surgiu a partir de uma proposta de *middleware* declarativo para o modelo de referência do sistema SBTVD. Após o trabalho realizado por MORENO (2006), o *middleware* Maestro foi escolhido para compor, juntamente com o FlexTV (SOUZA FILHO *et al*, 2007) o *middleware* brasileiro de TV digital denominado Ginga. Depois de aceitos como subsistemas do *middleware* Ginga, Maestro e FlexTV tiveram seus nomes modificados para Ginga-NCL e Ginga-J, respectivamente. Sendo o primeiro responsável pela apresentação de documentos NCL e o segundo por processar aplicações interativas Java.

Por serem precursoras de Ginga-NCL, as tecnologias apresentadas por MORENO (2006) são válidas para este *middleware*, e, portanto seu trabalho é apresentado de forma resumida nas subseções a seguir. As Subseções 0 e 3.1.2 discorrem, respectivamente, sobre a arquitetura e a implementação do *middleware* Maestro. 3.1.3 destaca as melhorias implementadas no *middleware* GDF.

3.1.1 Arquitetura Inicial do Middleware Ginga-NCL

A arquitetura do *middleware* Maestro foi elaborada de forma modular (MORENO, 2006), conforme ilustrado na Figura 8; e está dividida nos seguintes módulos principais: Sintonizador, Filtro de Seções, DSM-CC, Núcleo e Exibidores; os quais são apresentados nessa subseção.

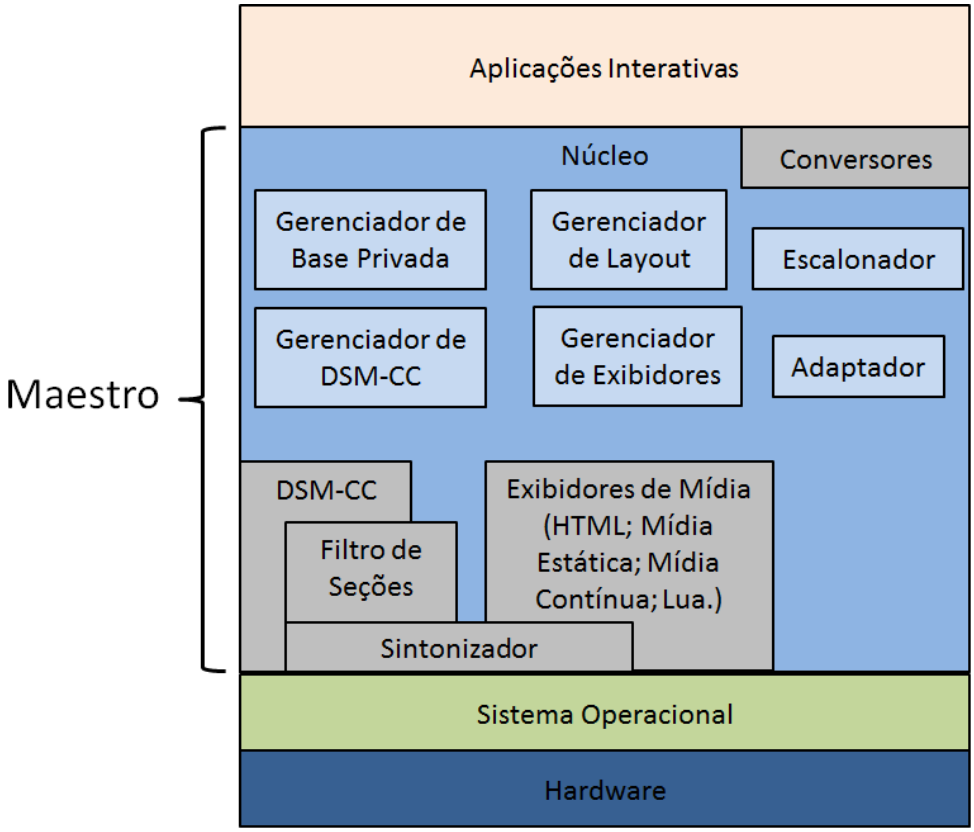

Figura 8: Arquitetura inicial do middleware declarativo Ginga-NCL

FONTE: Adaptado de MORENO (2006)

O *middleware* Maestro é responsável, de modo geral, pela recepção do sinal digital, extração do conteúdo interativo, e apresentação do documento NCL. O módulo Sintonizador é o ponto de entrada do fluxo de dados no sistema. Ele é responsável por sintonizar uma frequência de canal, demodular o sinal de TV, e extrair o fluxo de transporte do sinal digital.

O módulo Filtro de Seções atua nas seções privadas do sistema de transporte MPEG-2, as quais são utilizadas para transportar dados de tabelas de informações de serviço, tais como mapa de programação, e informações de

eventos da grade de programação. Além das tabelas, o módulo Filtro de Seções é utilizado para entregar o fluxo de dados ao módulo DSM-CC, responsável por tratar as funcionalidades desse protocolo, o qual normalmente solicita as seções contendo carrossel de objetos ou eventos DSM-CC.

Uma das principais funções do módulo DSM-CC consiste em decodificar o carrossel de objetos, criando um sistema de arquivos para esse carrossel de acordo com o sistema operacional da plataforma (MORENO, 2006). O módulo DSM-CC foi definido na arquitetura do *middleware* Maestro com o objetivo de oferecer uma API para manipulação do carrossel de objetos, bem como de eventos de fluxo, obtidos através do módulo Filtro de Seções. Após a extração da aplicação interativa do fluxo de transporte, pelo módulo DSM-CC, o módulo Conversor traduz os documentos NCL em estruturas de execução apropriadas para o controle de apresentação dos documentos NCL.

O submódulo Gerenciador de Base Privada do formatador NCL é responsável por agrupar e gerenciar um conjunto de documentos NCL denominado de base privada. Sempre que uma frequência de canal de TV é sintonizada, o *middleware* abre uma nova base privada para agrupar os documentos transmitidos no fluxo de transporte. Ao sintonizar uma nova frequência, a base privada anterior é excluída.

O Gerenciador DSM-CC permite que provedores de conteúdo acessem as bases privadas de suas aplicações NCL nos terminais de acesso através de eventos DSM-CC. Portanto, este componente faz solicitações ao Gerenciador de Base Privada para acessar a base privada do fluxo de transporte corrente.

O Gerenciador de *Layout* do formatador tem como função, mapear as regiões de apresentação de mídias, definidas no documento NCL, em componentes gráficos disponíveis no terminal de acesso. Cabe também ao Gerenciador de *Layout* definir as características espaciais dos exibidores, bem como dos objetos de execução. Os exibidores de objetos mídias são de responsabilidade do submódulo Gerenciador de Exibidores. O Adaptador é encarregado de lidar com estratégias de adaptação ao contexto, relacionadas ao perfil do usuário telespectador. Por fim, para escalonar os objetos a serem apresentados, MORENO (2006) define o submódulo Escalonador.

A seção seguinte apresenta a prova de conceito do *middleware* Maestro através da implementação dos módulos básicos que compõem sua arquitetura.

3.1.2 Implementação Inicial do Middleware Ginga-NCL

> "A implementação do middleware declarativo Maestro teve início com a adaptação do Formatador HyperProp (Rodrigues, 2003), originalmente implementado na linguagem Java (Rodrigues, 2003), ao contexto da TV Digital" (MORENO, 2006).

Conforme afirmado no texto de MORENO (2006), a implementação do *middleware* Maestro teve início a partir de uma implementação Java do formatador HyperProp, e foi adaptado para a linguagem C++ visando plataformas que utilizam Linux como sistemas operacional.

Embora os módulos Filtro de Seções, Sintonizador e DSM-CC tenham sido modelados na arquitetura do *middleware* Maestro, suas implementações não foram realizadas por MORENO (2006). As implementações dos módulos foram desenvolvidas pelo laboratório Lavid da Universidade Federal da Paraíba como parte integrante do *middleware* procedural FlexTV, e posteriormente integrado ao *middleware* Maestro durante o processo de integração ao sistema Ginga. Os submódulos Gerenciador de Bases Privadas e Gerenciador DSM-CC, pertencentes ao componente Núcleo, apesar de modelados foram deixados como trabalhos futuros.

O *middleware* Maestro teve sua prova de conceito integrada ao perfil simples de terminal de acesso definido pelo sistema brasileiro de TV digital, com processador de 700 MHz e 128 MB de memória RAM.

3.1.3 Evolução do Middleware Ginga-NCL

Posteriormente, após aprovação do *middleware* Maestro como padrão para o sistema ISDB-T, o *middleware* passou a se chamar Ginga-NCL, e com os avanços dos trabalhos realizados no *middleware* procedural FlexTV, (ou Ginga-J) formam hoje o *middleware* Ginga do sistema brasileiro de TV digital.

A arquitetura do *middleware* Ginga está dividida em três módulos principais. O primeiro foi denominado Núcleo Comum por prover funcionalidades comuns aos subsistemas Ginga-NCL e Ginga-J, os quais compõem os serviços específicos do middleware Ginga. A Figura 9 ilustra a arquitetura básica do sistema Ginga.

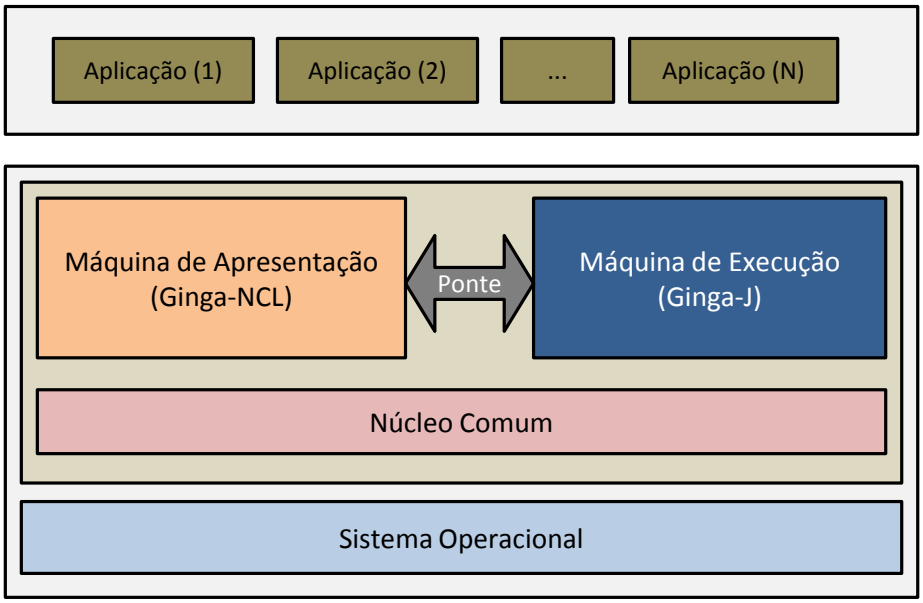

Figura 9: Arquitetura básica do middleware Ginga

FONTE: Adaptado de ABNT-N06 (2007)

O subsistema Ginga-NCL é encarregado de processar aplicações declarativas NCL, e controlar sua apresentação no terminal receptor através do formatador NCL. Este subsistema tem a linguagem de programação NCL como seu pilar principal. Além de NCL, o subsistema Ginga-NCL suporta *scripts* Lua, habilitando autores de conteúdos interativos a terem maior poder de expressão

em suas obras. Ginga-J é o subsistema de Ginga responsável por processar as aplicações desenvolvidas em código procedural Java, conhecidas como Xlet.

O Núcleo Comum atende a ambos os subsistemas Ginga-NCL e Ginga-J provendo a estes procedimentos para obtenção e decodificação de conteúdos interativos, transmitidos por fornecedores em fluxos de transporte MPEG-2, ou através de canal de interatividade.

Em comparação ao núcleo do *middleware* Maestro, o Núcleo Comum do middleware Ginga especifica os seguintes módulos adicionais:

- **Gerenciador Gráfico:** encarregado de abstrair as interfaces do sistema operacional para renderização de elementos gráficos.
- **Gerenciador de Atualizações:** módulo responsável por gerenciar as atualizações de componentes de software do sistema, transmitidos via sinal digital.
- **Gerenciador de Contexto:** encarregado de realizar o gerenciamento das informações do sistema sobre qual o middleware executa, bem como informações sobre o usuário de TV.
- **Persistência:** módulo para tratamento de persistência de dados no terminal receptor.

O *middleware* Ginga possui uma implementação de referência, a qual pode ser obtida gratuitamente a partir do portal comunitário, disponibilizado pelo governo brasileiro em SOFTWAREPUBLICO (2012). A implementação de referência do *middleware* contempla os módulos apresentados nesta seção, porém com algumas limitações em suas funcionalidades, as quais não apresentam grandes impactos para a apresentação de aplicações interativas. Como exemplos, se podem citar os módulos: Gerenciador de Atualizações, e Gerenciador de Contexto; além de exibidores de monomídias para o formato GIF animado.

3.1.4 Considerações sobre Ginga-NCL para Dispositivos Fixos (GDF)

O sistema ISDB-T permite a recepção do sinal digital por duas categorias de terminais:

- Fixos, receptores de sinal digital *fullseg*.
- Portáteis, receptores de sinal digital *oneseg*.

A implementação de referência apresentada nesta seção tem como foco os terminais receptores fixos, e foi desenvolvida para rodar sobre a plataforma aberta Linux. Entretanto, é importante prover uma implementação de referência para dispositivos portáteis, de forma a verificar a viabilidade do *middleware* em terminais com baixo recurso de *hardware*. A subseção 3.2 apresenta o trabalho realizado por CRUZ *et al* (2008) durante o porte da implementação GDF para dispositivos móveis.

3.2 Ginga-NCL para Dispositivos Portáteis (GDP)

Ginga-NCL para Dispositivos Portáteis (GDP) é a primeira implementação de referência para terminais receptores de TV digital portáteis, e foi apresentada por CRUZ *et al* (2008). As subseções a seguir discorrem sobre seus trabalhos.

3.2.1 Arquitetura do Middleware GDP

O GDP é um porte da arquitetura da Implementação de Referência para Dispositivos Fixos (IRDF) para plataformas portáteis, porém com restrições de alguns módulos em sua arquitetura. A Figura 10 exibe os módulos que compõem a arquitetura do GDP. Como pode ser observado, o middleware GDP está dividido em dois subsistemas lógicos: Máquina de Apresentação Ginga-NCL (MAG) e Núcleo Ginga (NG). O primeiro tem como responsabilidade principal orquestrar a apresentação da aplicação interativa no terminal receptor. O segundo é

responsável pela recepção do sinal de TV, processamento dos dados de entrada no sistema, bem como por prover funcionalidades de baixo nível, próximas ao sistema operacional.

O subsistema NG é o ponto de entrada dos fluxos de dados do conteúdo interativo para o sistema Ginga-NCL. Os módulos que compõem o subsistema NG provêm a MAG interfaces para sintonizar um canal de TV digital; extrair informações de serviços disponibilizados pela emissora de TV; bem como processar o conteúdo interativo, multiplexado juntamente com a programação audiovisual, transmitida pela emissora de TV. Para uma melhor compreensão do subsistema NG, seus submódulos são descritos a seguir:

Figura 10: Arquitetura da implementação de referência GDP

FONTE: Extraído de CRUZ *et al* (2008)

- **Sintonizador 1-seg:** sintoniza uma frequência de canal no segmento destinado à recepção móvel (segmento central dos treze utilizados para transmissão do sinal de TV digital no sistema ISDB-T – oneseg).
- **Processador de Dados:** semelhante ao módulo apresentado na seção 3.1, porém, para recepção *oneseg* este componente não necessita processar grandes quantidades de fluxos de dados, uma vez que a programação

destinada a dispositivos portáteis é limitada se comparada com a recepção *fullseg*.

- **Persistência:** este módulo gerencia o armazenamento das aplicações interativas (documentos NCL e scripts Lua), bem como os conteúdos aos quais essas aplicações referenciam (arquivos de monomídias, por exemplo).
- **Exibidores:** o módulo Exibidores prover ao Núcleo Ginga interfaces para decodificação das monomídias utilizadas pela aplicação NCL.
- **Transporte:** faz o tratamento dos protocolos e interfaces de rede, além de atender as demandas da MAG por conteúdos interativos provenientes de outras fontes de fluxos de dados diferente do Sintonizador 1-seg.
- **Gerenciador de Contexto:** gerencia as informações sobre o sistema embarcado no dispositivo e sobre o perfil do usuário telespectador.
- **Gerenciador Gráfico:** responsável por realizar o controle espacial da renderização dos componentes gráficos utilizados pela aplicação interativa.
- **Gerenciador de Atualizações:** módulo encarregado de gerenciar atualizações de componentes do sistema, os quais podem ser atualizados de forma independente. Este módulo recebe os dados do componente Transporte.
- **Máquina Lua:** tendo em vista que a norma brasileira de TV digital especifica Lua como linguagem de *script* para o *middleware* Ginga-NCL, este módulo foi adicionado ao Núcleo Ginga, e é responsável por interpretar os *scripts* desta linguagem.
- **Adaptadores:** os adaptadores foram definidos para padronizar a comunicação entre a MAG e a API de decodificação de conteúdo dos exibidores.

A máquina de apresentação MAG é responsável pelo ciclo de vida e pela apresentação do conteúdo interativo. Uma vez que o conteúdo interativo é disponibilizado pelo NG, a MAG é notificada e seu formatador NCL passa a controlar a aplicação interativa. Os módulos que compõem a MAG são apresentados a seguir:

- **Conversor:** assim como especificado por MORENO (2006), o módulo conversor traduz um documento NCL para uma estrutura de execução apropriada para o controle da apresentação da aplicação interativa.
- **Escalonador:** orquestra a apresentação dos objetos que compõem a aplicação interativa, tomando como base as relações definidas no documento de autoria. O Escalonador controla a execução síncrona do documento NCL.
- **Formatador:** ponto de entrada da aplicação interativa para a Máquina de Apresentação Ginga-NCL. Recebe o conteúdo interativo do Núcleo Ginga e controla o ciclo de vida de sua apresentação.
- **Gerenciador de Bases Privadas:** cada provedor de conteúdo interativo pode enviar informações a serem persistidas no terminal receptor. Essas informações devem ser acessadas somente por aplicações interativas “autorizadas”; portanto, o Gerenciador de Bases Privadas garante o acesso às bases privadas somente por conteúdos interativos autorizados.
- **Gerenciador de Leiaute:** faz a associação entre os objetos de mídias e suas regiões de apresentação, e gerencia o controle espacial dos exibidores.
- **Gerenciador de Exibidores:** responsável por gerenciar, e prover tratadores de monomídias conforme descrito no documento de autoria da aplicação NCL.
- **Gerenciador de Contexto NCL:** após a conversão do documento NCL em estruturas de execução, é necessário gerenciar os contextos onde essas estruturas serão tratadas e apresentadas; portanto, o módulo Gerenciador de Contexto NCL foi adicionado à arquitetura segundo CRUZ (2008).

Analogamente ao trabalho apresentado por MORENO (2006), a arquitetura do GDP foi elaborada de forma modular, com funções bem definidas para cada módulo.

3.2.2 Implementação de Referência para Dispositivos Portáteis

O processo de porte da implementação IRDF para Symbian são apresentados nesta seção, destacando as dificuldades encontradas, bem como as soluções aplicadas em cada caso.

CRUZ (2008), baseado em sua arquitetura, analisou as plataformas disponíveis no momento de seu trabalho, e chegou à conclusão que Symbian seria a plataforma ideal para sua proposta. Portanto, sua arquitetura foi implementada para este sistema operacional.

> "a implementação realizada foi desenvolvida utilizando a plataforma nativa do sistema operacional Symbian, utilizando Symbian C++" (CRUZ 2008).

A plataforma Symbian suporta os tipos básicos da linguagem C++, e tem Symbian C++ como linguagem de programação. Os tipos primitivos de Symbian C++ são redefinições de tipos básicos da linguagem C++ com intuito de manter uma padronização na codificação de classes destinadas ao sistema embarcado em questão. Como exemplo, pode-se citar o tipo `TInt`, que trata-se de uma redefinição do tipo `int` em C++. Porém, Symbian C++ não tem suporte as estrutura padrões das linguagens C e C++, muito menos suporte a biblioteca STL (*Standard Template Library*), a qual é amplamente utilizada por programadores C++. CRUZ (2008), relata a falta dessas bibliotecas como dificuldade para o porte da IRDF para Symbian, conforme texto a seguir:

> "A implementação de referência Ginga-NCL para terminais fixos faz uso de estruturas definidas nas bibliotecas padrões C, C++ e STL (Standard Template Library). As bibliotecas encontradas no Symbian C++, porém, diferem dessas bibliotecas, o que poderia afetar a arquitetura da implementação realizada como um todo" (CRUZ, 2008).

Porém, numa tentativa de padronização entre Symbian C++ e as bibliotecas padrões C e C++, o consórcio Symbian, juntamente com a Nokia, desenvolveram as bibliotecas PIPS e Open C. Juntas essas bibliotecas cobrem

boa parte das funcionalidades utilizadas na IRDF, faltando apenas à biblioteca STL, a qual foi posteriormente portada por um membro da comunidade Symbian.

Um dos requisitos importantes para aplicações que exijam sincronismo de mídias, como é o caso de aplicações interativas NCL, é a capacidade de processamento paralelo de tarefas (ou simulação de processamento paralelo). A implementação IRDF faz uso de processos leves, ou *threads*, da biblioteca POSIX; porém, por restrições da plataforma Symbian, a manipulação de objetos específicos de mídias só pode ser realizada no processo principal da aplicação. Uma alternativa para solucionar o problema foi o uso de Objetos Ativos, os quais foram utilizados para simular o comportamento de *threads* (CRUZ, 2008).

Além do uso de Objetos Ativos como forma alternativa ao uso de *threads*, CRUZ (2008) teve que adaptar o módulo Conversor, de forma a atender restrições de recursos de *hardware*, características de terminais portáteis. Após realizar testes de desempenho sobre a interpretação e apresentação de documentos NCL, CRUZ (2008) concluiu não haver necessidade em usar API DOM para realização de *parser* de documentos XML. Portanto, o módulo Conversor da implementação IRDF sofreu alterações em seu porte, de formar a utilizar API SAX.

O modelo de apresentação de conteúdos em Symbian é realizado através de uma estrutura denominada Janela (`Window`). O módulo Gerenciador de Leiaute, a partir de uma Janela, define as regiões para apresentação das mídias do documento NCL. Cada região foi mapeada a uma estrutura de Symbian denominada Controle (`Control`), as quais foram utilizadas para a renderização dos conteúdos decodificados pelos exibidores.

Outro componente bastante importante da arquitetura GDP é o módulo Exibidores. A implementação de referência oferece suporte a decodificação de objetos de áudio, vídeo, imagem e documentos HTML. Os exibidores de áudio e vídeo foram implementados através do arcabouço de multimídia MMF (*Multi Media Framework*) disponível na plataforma. O exibidor de imagem, por questões de limitações de Symbian, foi implementado apenas para imagens específicas da plataforma (formato mbm). A estrutura Symbian `CBrCtlInterface` foi utilizada para exibição de documentos HTML.

A implementação GDP teve sua prova de conceito implantada em dois ambientes de execução:

- Emulador de dispositivos Symbian para computador pessoal.
- Dispositivo portátil (Nokia 5700 Express Music com processador de 369 MHz e 64 MB de memória RAM).

3.2.3 Considerações sobre Ginga-NCL para Dispositivos Portáteis

Os resultados alcançados por CRUZ (2008) foram bastante expressivos. Principalmente levando-se em consideração os recursos de *hardware* e *software* disponíveis durante a elaboração de seu trabalho. Porém, alguns módulos importantes de sua arquitetura não foram contemplados pela implementação de referência, ficando como proposta de trabalhos futuros.

> "Como os receptores disponíveis no mercado não dispunham de recepção na modulação SBTVD-T (modulação ISDB), os módulos Sintonizador, Filtro de Seções e Processador de Fluxo de Dados — usados no tratamento do fluxo de transporte — não foram implementados, ficando como trabalhos futuros a serem realizados" (CRUZ, 2008).

Entretanto, com base em estudos preliminares, foi observada a viabilidade em criar um componente de *software* capaz de simular o módulo Sintonizador, de forma a abstrair a parte de demodulação de sinal. Neste cenário, o fluxo de dados, ao invés de vir pelo ar, seria obtido a partir de arquivos de fluxos de transporte, armazenados localmente no terminal receptor. A partir desta simulação, é possível portar e validar os módulos: Filtro de Seção e Processador de Fluxos de Dados.

Adicionalmente aos módulos apresentados, a implementação de referência GDP não contempla a exibição de mídias NCLua, pois segundo CRUZ (2008), a plataforma escolhida para porte não havia suporte ao interpretador Lua, essencial para execução de scripts dessa linguagem.

3.3 Resumo Comparativo entre GDF e GDP

Com base nos estudos realizados, apresentados neste capítulo, pode-se concluir que a implementação de referência Ginga-NCL para Dispositivos Fixos (GDF), que tem como base o *middleware* Maestro, atende as necessidades referentes à apresentação de aplicações NCL para terminais receptores fixos. Porém, a mesma sentença não pode ser afirmada para a implementação Ginga-NCL para Dispositivos Portáteis (GDP), onde componentes importantes da arquitetura do *middleware* proposto por CRUZ (2008) foram deixados para trabalhos futuros, como são os casos dos módulos: Sintonizador *1-seg*, Filtro de Seções, Processador de Fluxo de Dados, e interpretador Lua; sendo este último fundamental para prover maior poder de expressividade a autores de conteúdos interativos, através de uso de objetos procedurais NCLua. Outra limitação da implementação de CRUZ (2008) está relacionada ao fato de sua solução não realizar a extração de aplicações interativas multiplexadas no fluxo de transporte, o qual é o ponto de entrada para o Filtro de Seções, após processamento do sinal digital pelo módulo Sintonizador *1-seg*. Portanto, uma solução para dispositivos portáteis com suporte aos módulos não contemplados por CRUZ (2008) se faz necessário. A Tabela 3 resume os módulos necessários para processamento e apresentação adequada de aplicações interativas NCL para dispositivos portáteis.

Máquina de Apresentação	**Núcleo Ginga**
Conversor	Simulador de Sinal Digital
Formatador	Processador de Fluxo de Dados
Escalonador	Persistência
Gerenciador de Bases Privadas	Exibidores
Orientador de Tela	Transporte
Gerenciador de Leiaute	Gerenciador de Contexto
Gerenciador de Exibidores	Gerenciador Gráfico
Gerenciador de Contexto NCL	Máquina Lua e suas extensões para TV

Tabela 3: Módulos desejados para um simulador de TV digital portátil

Por não haver na literatura nenhuma solução que atenda as necessidades listada na Tabela 3, está sendo proposto um simulador de TV digital para dispositivos portáteis, o qual é apresentado no Capítulo 4. O simulador proposto

visa contemplar todos os módulos dos subsistemas Máquina de Apresentação e Núcleo Ginga, listados na Tabela 3.

4 Simulador Ginga-NCL para Dispositivos (SGDP)

O Simulador Ginga-NCL para Dispositivos Portáteis (SGDP) é uma proposta de ferramenta para simulação de comportamento de receptores de TV digital portáteis para o sistema ISDB-T. A arquitetura da ferramenta proposta tem como base os trabalhos realizados por MORENO (2006) e CRUZ *et al* (2008) apresentados no Capítulo 3. Adicionalmente, o simulador de TV portátil faz uso de módulos de software elaborados pelo programa de TV digital do Instituto Nokia de Tecnologia (INdT).

A seção 4.1 discorre sobre a arquitetura do sistema, onde são apresentados os componentes de software responsáveis pelo processamento e apresentação de documentos NCL. Em seguida, na seção 4.2, são apresentadas as características da plataforma alvo para implantação do simulador. A seção 4.3 discorre sobre detalhes de implementação da ferramenta proposta, e apresenta um resumo comparativo entre os trabalhos de MORENO (2006), CRUZ *et al* (2008) e a solução aqui proposta.

4.1 Arquitetura SGDP

Assim como os trabalhos realizados por MORENO (2006) e CRUZ *et al* (2008) o SGDP tem uma arquitetura modular. A Figura 11 exibe os componentes de *software* que formam sua arquitetura. Pelo desenho ilustrativo pode ser observado que a arquitetura do SGDP é semelhante à apresentada por CRUZ *et al* (2008), e também está dividida em dois subsistemas principais: Núcleo Ginga e Máquina de Apresentação. Adicionalmente a estes subsistemas, o módulo *Logger* está sendo proposto na arquitetura do SGDP para prover coleta de informações sobre a apresentação do documento interativo pelo *middleware*.

Tanto o Núcleo Ginga quanto a Máquina de Apresentação, modelados por CRUZ *et al* (2008), sofreram modificações de forma a atender às especificações atuais das normas que regem o sistema brasileiro de TV digital ISDB-T (ABNT-N06, 2007). Pelo fato deste trabalho ter como objetivo prover um ambiente de

simulação para TV digital portátil, o módulo Sintonizador *1-seg*, modelado por CRUZ *et al* (2008), foi substituído pelo módulo Simulador de Sinal Digital (SSD) neste trabalho.

A Figura 11 exibe todos os componentes de software que formam a arquitetura do simulador SGDP, porém apenas os componentes com marcações ● e ▲ representam as contribuições exclusivas desenvolvidas neste trabalho, sendo os demais componentes elaborados juntamente com a equipe de TV digital do Instituto Nokia de Tecnologia (INdT). A marcação ● indica que o componente de software foi totalmente elaborado neste trabalho de pesquisa, e a marcação ▲ indica que o componente original sofreu modificações de forma a atender as necessidades do simulador de TV.

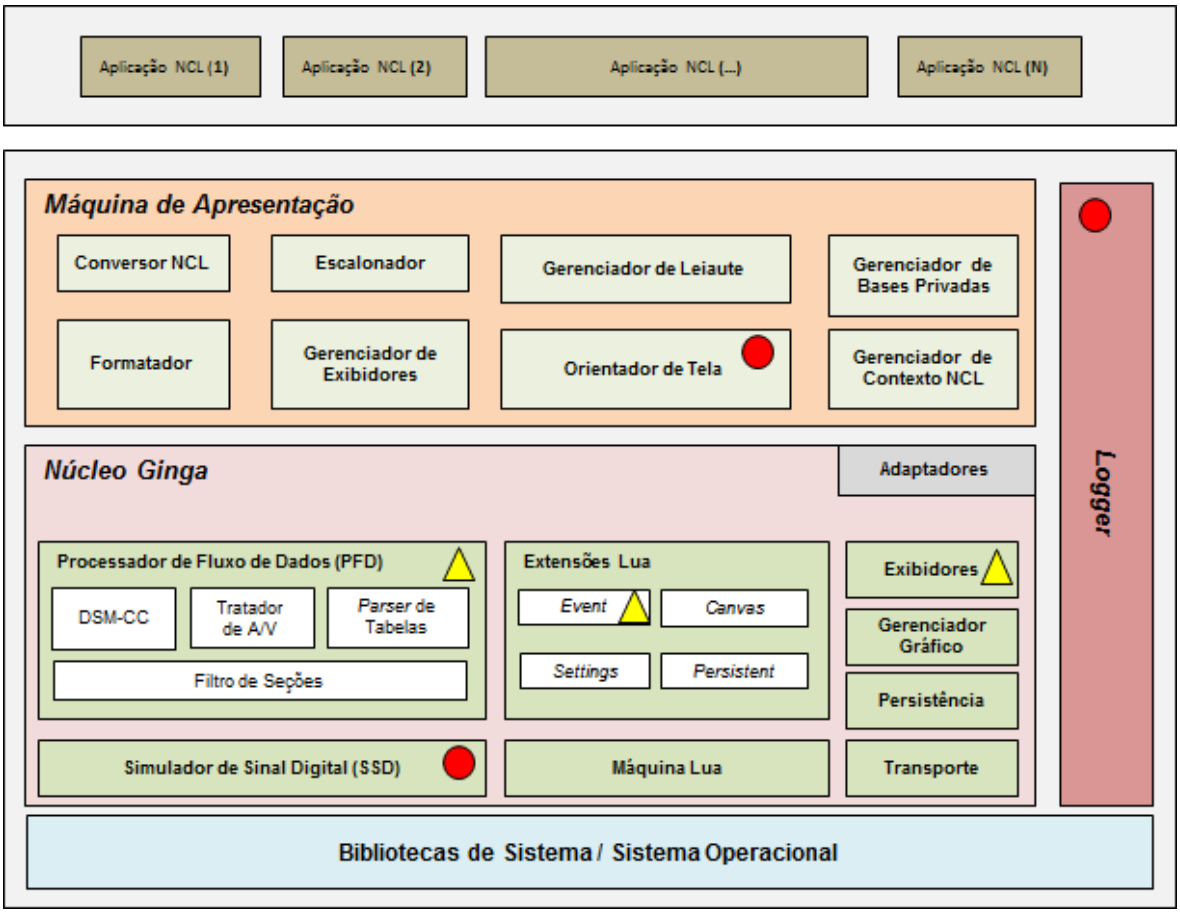

Figura 11: Arquitetura do simulador SGDP

As Subseções 4.1.1 e 4.1.2 discorrem sobre os subsistemas Núcleo Ginga e Máquina de Apresentação respectivamente. O módulo *Logger* é apresentado na Subseção 4.1.3.

4.1.1 Núcleo Ginga

Os módulos Máquina Lua, Transporte, Persistência, Exibidores, Gerenciador Gráfico, e Adaptadores, do subsistema Núcleo Ginga, possuem as mesmas funcionalidades de CRUZ (2008) apresentadas na Seção 3.2.1. Filtro de Seções de DSM-CC seguem as definições de MORENO (2006) da seção 0.

Conforme apresentado na Subseção 1.2, uma das características importantes quando se desenvolve aplicações interativas para TV digital está relacionada ao tempo decorrido para baixar os conteúdos interativos pelo terminal receptor. Com o objetivo de permitir que o tempo real de *download* seja observado na prática, está sendo proposto o módulo simulador Simulador de Sinal Digital (SSD).

A função do módulo SSD é garantir a simulação de recepção do sinal *oneseg*. Portanto, é imprescindível que a taxa de bits/segundos, de entrada no sistema, mantenha a mesma taxa de bits/segundos entregues pelo sinal *oneseg*.

O sinal *oneseg* tem restrições em sua largura de banda, pois a programação principal de áudio e vídeo chega a ocupar cerca de 96% da banda, sobrando apenas 4% para aplicações interativas, e demais informações de serviço. Portanto, ao desenvolver aplicações interativas destinadas a dispositivos portáteis, é importante levar essas informações em consideração. Aplicações interativas com grande quantidade de mídias podem levar bastante tempo para serem baixadas pelo terminal receptor.

O módulo SSD é responsável pela simulação do sinal de TV digital *1-seg*, e garante a entrega dos fluxos de transporte de acordo com a taxa de bits/segundos especificada. Dessa forma é possível obter o tempo real em que o conteúdo interativo levou para ser baixado.

O SSD recebe como entrada um arquivo no formato de fluxo de transporte MPEG-2 (*MPEG-2 transport stream*) contendo a programação interativa. O Processador de Fluxo de Dados (PFD), através do submódulo Filtro de Seções, se inscreve no simulador SSD para receber os fluxos de transporte. À medida que os fluxos são recebidos, o submódulo Filtro de Seções os processa, e faz a

devida demultiplexação dos elementos, os quais são encaminhados para os respectivos submódulos:

- **DSM-CC**, caso o fluxo de dados seja referente aos módulos DSM-CC que compõem as aplicações interativas.
- ***Parser* de Tabelas**, caso o fluxo de dados seja do tipo seção de tabela de informação de serviço (*Service Information* - SI).
- **Tratador de A/V**, caso o fluxo de dados seja referente a pacotes de fluxos elementares (*Packetize Elementary Stream* - PES) de áudio ou vídeo.

Dentre os fluxos de dados entregues pelo simulador SSD estão os fluxos elementares de áudio e vídeo, que juntos formam a programação principal transmitida pela emissora de TV. Para decodificar os fluxos elementares no simulador SGDP, está sendo proposto o módulo Tratador de A/V, cuja função é prover um conjunto de interfaces para exibição do vídeo principal da programação de TV. Os fluxos elementares de áudio e vídeo chegam ao Tratador A/V empacotados no formato de pacotes PES, portanto é de responsabilidade do tratador processar os pacotes antes da decodificação dos fluxos elementares.

Conforme apresentado por MORENO (2006), as informações de serviços de um programa de TV estão multiplexadas no fluxo de transporte através de tabelas SI. Para recuperar as tabelas de serviço, se faz necessário um módulo capaz de montá-las a partir de seções de tabelas SI. Portanto, o módulo *Parser* de Tabelas foi modelado com a finalidade de suprir a necessidade de recuperar os metadados referentes às informações de serviço.

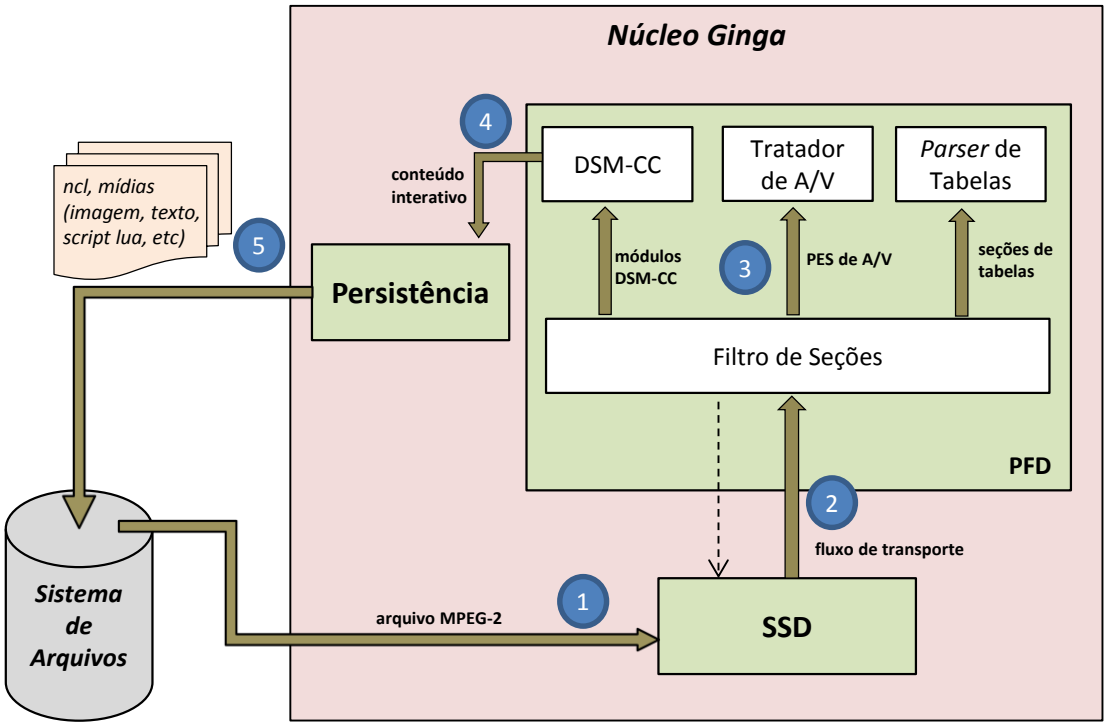

Figura 12: Extração de conteúdos interativos do fluxo de transporte

A Figura 12 representa uma versão simplificada da arquitetura, com foco no processamento do fluxo de transporte MPEG-2. Através da figura ilustrativa é possível observar o fluxo de dados trafegados durante a extração do conteúdo interativo a partir de um sinal digital simulado. As marcações numéricas indicam a sequência de processamento dos dados. Iniciando pela leitura do arquivo MPEG (1); passando pelo simulador SSD, o qual envia os fluxos de transporte para o submódulo Filtro de Seções (2); em seguida os elementos demultiplexados são enviados aos seus respectivos submódulos DSM-CC, Tratador de A/V ou *Parser* de Tabelas (3); após processados pelo DSM-CC, os documentos que compõem a aplicação NCL são encaminhados ao módulo de persistência (4), o qual os armazena no sistema de arquivos (5).

Além de funcionalidades para extração do conteúdo interativo do sinal simulado, o Núcleo Ginga provê à Máquina de Apresentação interfaces para decodificação de monomídias, acesso à rede de dados, interpretação de *scripts* Lua, dentre outras. A Figura 13 demonstra o relacionamento entre os módulos do Núcleo Ginga responsáveis por prover as funcionalidades específicas à Máquina de Apresentação.

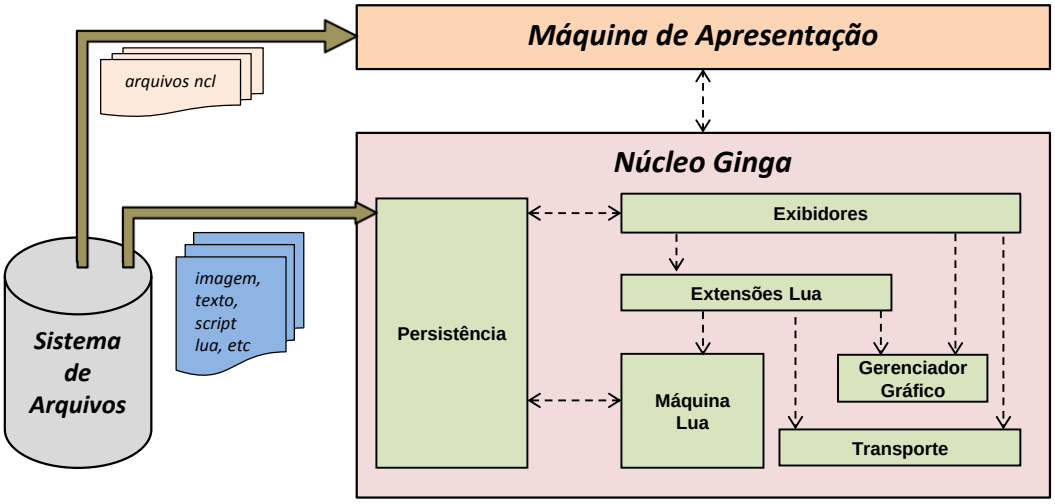

Figura 13: Núcleo Ginga – processamento de aplicações interativas

O módulo Exibidores recebe requisições para decodificação e exibição de objetos de mídia. Os arquivos correspondentes às mídias são requisitados ao módulo Persistência. Arquivos de texto, imagem, áudio e vídeo são decodificados por elementos específicos de plataforma. *Scripts* Lua são encaminhados ao módulo Extensões Lua.

O módulo Extensões Lua contempla os módulos de extensão da linguagem Lua (*Event*, *Canvas*, *Persistent* e *Settings*) definidos para o padrão ISDB-T (seção 2.3.2). Ele interage com a Máquina Lua, responsável pela interpretação dos *scripts* da linguagem. A Máquina Lua requisita ao módulo Persistência os arquivos de monomídias referenciados nos *scripts* Lua.

O Gerenciador Gráfico provê aos módulos Exibidores e Extensões Lua rotinas para manipulação de elementos gráficos da plataforma. O módulo Transporte oferece interfaces de acesso ao canal de retorno.

O módulo Gerenciador de Atualizações não é contemplado na arquitetura proposta, por não haver necessidade em atualizar os componentes de software via o sinal digital de TV.

4.1.2 Máquina de Apresentação

O subsistema Máquina de Apresentação, modelado neste trabalho, possui finalidades semelhantes às apresentadas por CRUZ (2008), descritas na Seção 3.2. Uma vez que os conteúdos interativos são extraídos do fluxo de transporte, e armazenados no sistema de arquivos, cabe à Máquina de Apresentação processar os documentos NCL, e gerenciar o ciclo de vida da aplicação interativa. Os módulos que compõem a Máquina de Apresentação do SGDP, representados na Figura 11, seguem a mesma modelagem de CRUZ (2008). Entretanto, para possibilitar o gerenciamento de mudança de orientação de tela, foi adicionado ao módulo Gerenciador de Leiaute, o submódulo Orientador de Tela. Segundo atualizações da norma brasileira (ABNT-N06, 2007), uma aplicação interativa pode informar o tipo de orientação de tela para sua exibição; portanto, cabe ao Orientador de Tela a responsabilidade de apresentar o conteúdo interativo na respectiva orientação de tela definida pelo autor obra interativa.

A Figura 14 ilustra o relacionamento entre os módulos do simulador SGDP, dando destaque para o processamento e apresentação de documentos NCL (a extração do conteúdo interativo pode ser observada na Figura 12).

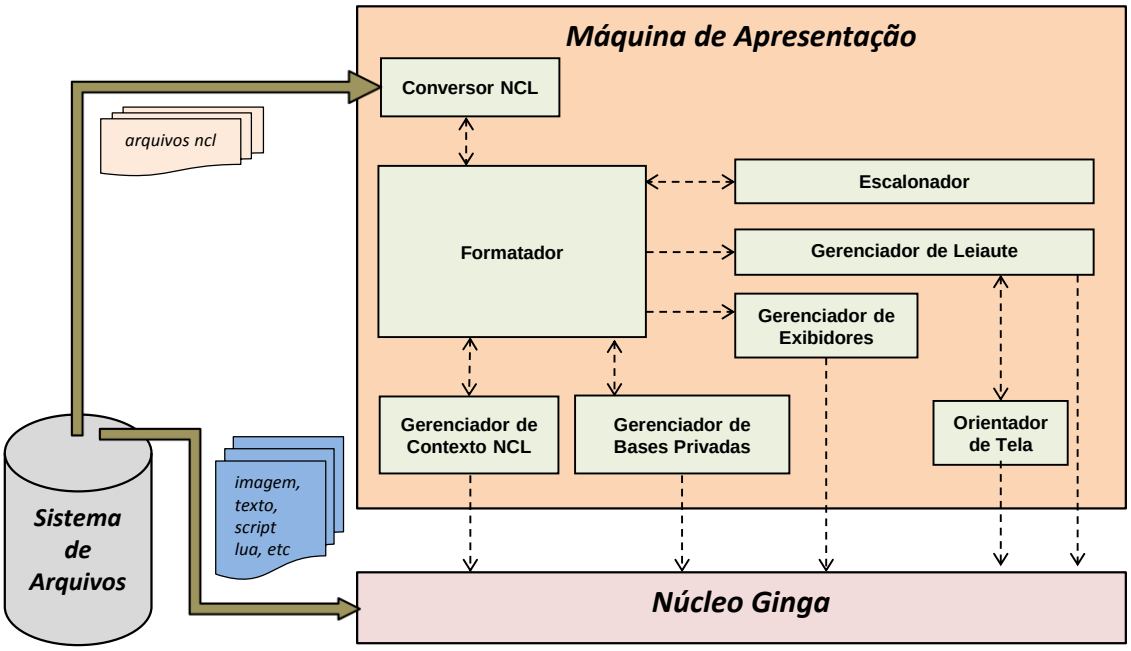

Figura 14: Relacionamento entre os módulos da máquina de apresentação

O módulo Conversor NCL, faz a leitura dos arquivos NCL e cria uma representação do modelo NCM em forma de estruturas capazes de serem processadas pelo Formatador. A representação do modelo NCM é acessada pelo Formatador, o qual coordena os principais módulos necessários para a apresentação adequada da aplicação NCL. O Formatador NCL interage com o módulo Escalonador, e recebe notificações temporais para exibição de mídias nos instantes especificados pelo autor da obra interativa através de elementos NCL. Após realizar o mapeamento das regiões, dos objetos de mídia, o módulo Formatador requisita ao Gerenciador de Leiaute regiões gráficas da tela do dispositivo para exibição apropriada das mídias. O módulo Orientador de Tela é responsável por realizar o monitoramento de orientação de tela do dispositivo móvel, e por notificar ao Gerenciador de Leiaute sobre mudanças dinâmicas de orientação.

O módulo Gerenciador de Exibidores oferece ao Formatador interfaces para exibição de monomídias (texto, imagem, NCLua, etc.) representadas por elementos `<media>`. Cada elemento de mídia tem um formato de arquivo específico, e deve ser decodificado de acordo com seu tipo MIME (*Multipurpose Internet Mail Extensions* – Extensões Multi-função para Mensagens de Internet). Portanto, é de responsabilidade do Gerenciador de Exibidores garantir que o exibidor correspondente ao tipo MIME do objeto de mídia seja criado.

O Gerenciador de Bases Privadas faz o monitoramento de eventos DSM-CC e notifica o Formatador sobre mudanças nas estruturas do documento NCL.

4.1.3 Logger

Uma característica importante durante a fase de desenvolvimento de programas é a capacidade de o sistema gerar informações sobre as operações realizadas pela execução de um programa. Essas informações auxiliam programadores a verificar se os fluxos de operações estão sendo executadas conforme definido no código programado. Aplicações interativas com foco em receptores de TV digital não fogem a essa regra, portanto é importante que sistemas de TV digitais (*middlewares*) forneçam um mecanismo para registrar e

disponibilizar os eventos relevantes durante a apresentação do conteúdo interativo. Para atender a esta necessidade, está se propondo o módulo *Logger*.

A expressão em inglês *log* (registro) é amplamente utilizada no meio computacional para representar o registro de informações sobre a execução de um programa, portanto o módulo *Logger* é o agente responsável por prover *logs* de apresentação de aplicações interativas.

A Figura 15 exibe o fluxo de dados no simulador SGDP. Do lado esquerdo pode ser observado o elemento de entrada para o sistema (fluxos de transporte MPEG-2). Os fluxos de transporte são processados pelo simulador, e o resultado do processamento é apresentado na tela do dispositivo portátil. As informações sobre o processamento do fluxo de transporte, bem como as informações sobre a apresentação do conteúdo interativo são armazenadas em arquivos de *log*.

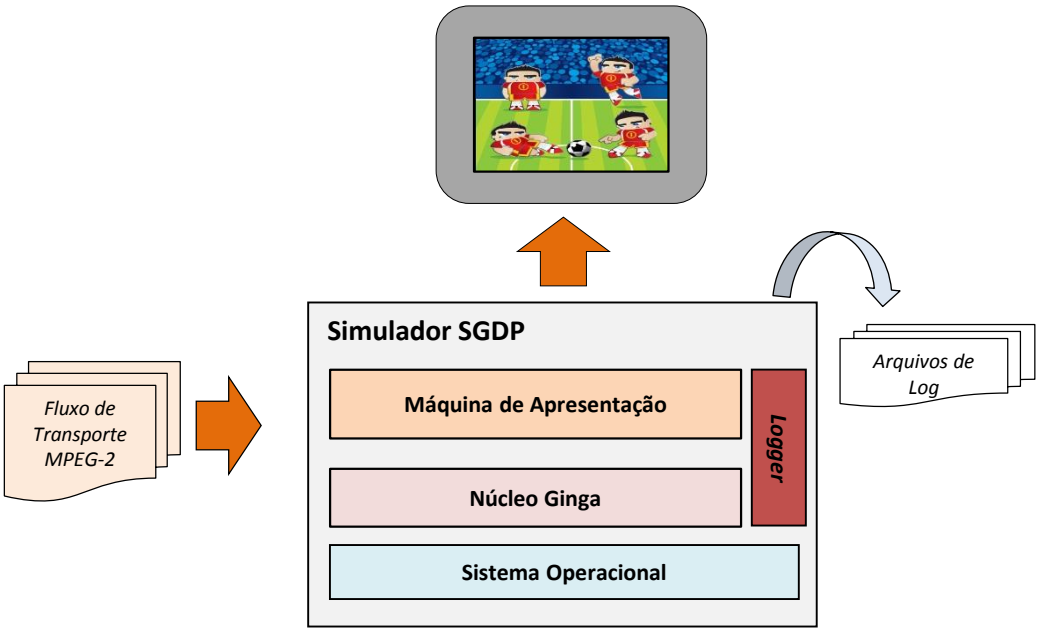

Figura 15: Fluxo de dados do simulador SGDP

4.2 Plataforma Alvo para Implementação do SGDP

Embora dispositivos móveis, de forma geral, sejam associados a telefones móveis, há uma grande variedade destes dispositivos com os mais diversos propósitos. Alguns deles de usabilidade abrangente, como é o caso de telefones

inteligentes (*smartphones*), e dispositivos *tablets*; outros, como dispositivos de monitoramento de sinais vitais na área de saúde, por exemplo, têm uso mais específico. Porém, o trabalho em questão tem como foco os dispositivos portáteis voltados para área de telefonia celular, em particular os telefones móveis.

Os telefones inteligentes necessitam de um sistema operacional móvel para gerenciar os recursos de *hardware* disponíveis na plataforma. A capacidade de um sistema operacional impacta diretamente nas funcionalidades do dispositivo.

Dentre os principais sistemas operacionais para dispositivos móveis na atualidade destacam-se as plataformas Symbian, Android, BlackBerry, iOS, e Windows Mobile. A Figura 16 demonstra a distribuição mundial (em porcentagem) de uso dos principais sistemas operacionais móveis nos anos de 2009 e 2010 (GARTNER, 2011).

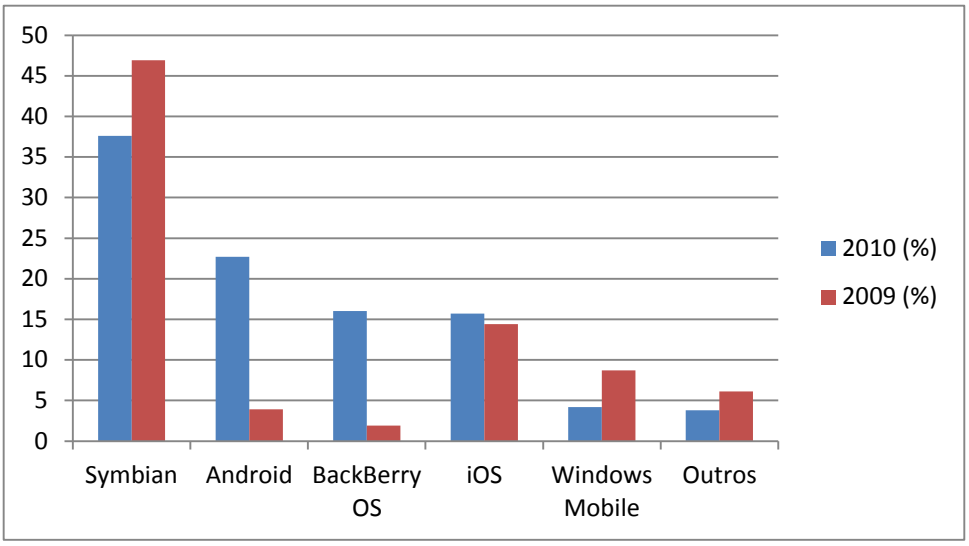

Figura 16: Distribuição mundial de uso dos sistemas operacionais móveis

FONTE: Dados extraídos de GARTNER (2011)

Pelas informações do gráfico, é possível observar que o sistema operacional Symbian está presente em grande parte dos telefones inteligentes, chegando a aproximadamente 50% da fatia do mercado mundial em 2009. Embora esse número tenha sido reduzido para 37% no ano de 2010, a plataforma Symbian ainda acumula uma quantidade expressiva de usuários.

Em sua dissertação de mestrado, CRUZ (2008) realizou uma análise comparativa entre os sistemas operacionais móveis disponíveis, e observou que a

plataforma Symbian possui características que permitem o desenvolvimento e implantação do *middleware* Ginga sem a necessidade de adquirir ferramentas proprietárias.

O sistema operacional Symbian é um sistema multitarefa com funcionalidades que incluem: sistema de arquivo, *framework* de interfaces gráficas de usuário, suporte a multimídias, além de bibliotecas para funcionalidades de comunicação. A plataforma é composta pelos seguintes componentes de software (BABIN, 2007):

- **Kernel:** núcleo central da plataforma, responsável por gerenciar os recursos de memória do sistema, escalonar a execução de programas, alocar memória compartilhada, e tratar qualquer funcionalidade relacionada ao acesso privilegiado a CPU.
- **Bibliotecas Base:** bibliotecas de usuário, sistemas de arquivos, e gerenciador de banco de dados. A biblioteca de usuário contem APIs que provêm funcionalidades como manipulação de *strings*, listas encadeadas, funções matemáticas, dentre outras.
- **Protocolos e Serviços de Aplicação:** os protocolos e serviços de aplicações provêm acesso às funcionalidades e serviços de aplicação aos programas.
- ***Framework* de Aplicação:** implementa a funcionalidade base da interface gráfica de usuário.
- **Arquitetura de Comunicações:** provê suporte a serviços e protocolos de comunicação, como TCP/IP, Bluetooth, USB, SMS e mensagens de e-mail.
- **Bibliotecas de Funcionalidades de *Middleware*:** APIs e *frameworks* não cobertos nos itens anteriores, tais como serviços de multimídia, segurança da plataforma, etc.

Por questões de segurança, os serviços oferecidos pelo Sistema Operacional (SO) Symbian são baseados numa arquitetura Cliente/Servidor; onde o servidor atua no gerenciamento dos recursos disponibilizados aos programas através de APIs clientes. A manipulação de arquivos em Symbian é possível através serviços providos pelo servidor de arquivos. Analogamente, o acesso a

recursos gráficos na plataforma é realizado através de serviços providos servidor de janelas.

Embora Symbian possua suporte a execução de múltiplas *threads*, esta abordagem não é amplamente utilizada por desenvolvedores de aplicações Symbian, por questões de desempenho de execução (BABIN, 2007). Ao invés de *threads*, Objetos Ativos são utilizados. Objetos Ativos simulam o comportamento de múltiplas *threads* de um único processo, porém são executados em uma única *thread*, a qual consiste de um escalonador ativo. O escalonador possui dois elementos fundamentais que garantem a simulação de *threads*: despachante de eventos e lista de objetos ativos.

Com base em estudos preliminares, foi observado que a conclusão de CRUZ (2008), sobre a plataforma Symbian, permanece válida nos dias atuais. Adicionalmente a este fator, o sistema operacional Symbian é amplamente utilizado por usuários de telefones inteligentes (GARTNER, 2011). Portanto, a plataforma Symbian foi escolhida como plataforma alvo para a implementação do simulador SGDP, embora também seja possível implementar o simulador em outras plataformas fechadas, desde que as devidas APIs seja disponibilizadas.

4.3 Implementação do Simulador SGDP

Esta seção discorre sobre o trabalho realizado durante a implementação do simulador SGDP para a plataforma Symbian. As seções a seguir descrevem o processo de programação do simulador proposto. Por se tratar de um sistema complexo, programar todos os módulos de software apresentados na seção 4.1 torna-se uma tarefa árdua; sendo sua implementação impraticável de ser realizada por um único desenvolvedor no prazo estipulado para a conclusão deste trabalho. Portanto, os módulos de software, apresentados na Figura 11 da Seção 4.1, desenvolvidos pelo Instituto Nokia de Tecnologia (INdT) foram reutilizados; e serão apresentados nesta seção juntamente com os módulos específicos deste trabalho. Todos os módulos de software que compõem a solução do simulador SGDP foram programados nas linguagens C, C++, e SymbianC++. As bibliotecas

STL (Standar Tamplate Library), PIPS (PIPS Is Posix on Symbain) e Open C foram amplamente utilizadas.

4.3.1 Núcleo Ginga

Conforme apresentado na subseção 4.2, os recursos para manipulação de arquivos em Symbian são oferecidos através de APIs do servidor de arquivos. A classe `RFs` provê métodos para abertura de sessões de comunicação com o servidor, e oferece serviços para leitura e escrita de dados no sistema de arquivos. Portanto, o servidor de arquivos Symbian foi amplamente utilizado, garantindo as funcionalidades do módulo Persistência. O módulo Transporte é coberto através dos componentes da arquitetura de comunicação da plataforma, como os componentes de software: servidor de soquetes, gerenciador de interfaces de rede, e protocolos de comunicação.

O Simulador de Sinal Digital foi programado conforme sua modelagem representada pelo diagrama de classe da Figura 17. `TsFileReader` é responsável por realizar a leitura dos bytes contidos no arquivo de fluxo de transporte. A simulação de taxa de bits para o sinal *oneseg* é garantida pelo `BitrateSimulator` – especialização da classe Symbian `CPeriodic` – o qual prover notificações periódicas ao simulador `OnesegSimulator`. A cada período de tempo o simulador `OnesegSimulator` envia um bloco de fluxo de transporte – `TsDataChunk` – ao observador registrado – `TsDataChunkListener`.

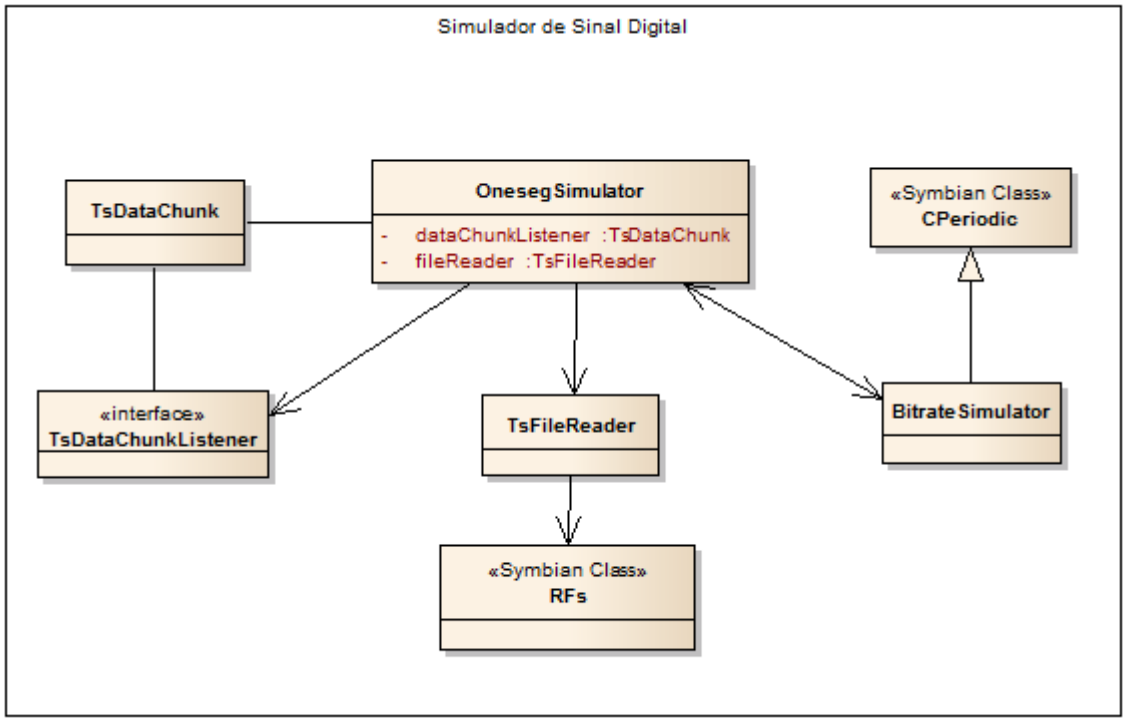

Figura 17: Diagrama de Classe – Simulador de Sinal Digital

Para a programação do submódulo Filtro de Seções, foi definida a classe `Demux`. Esta classe realiza a interface `TsDataChunkListener` e provê métodos para leitura dos pacotes de fluxo de transporte. A cada pacote extraído, o `Demux` notifica os respectivos consumidores: `PesConsumer` e `SectionConsumer` (consumidores de pacotes de fluxo de transporte dos tipos PES, e seção de tabelas, respectivamente). Para decodificação de pacotes PES de áudio e vídeo foi definida a classe `AvManager`, responsável por interagir com os decodificadores disponíveis na plataforma. Para a interpretação das tabelas de serviços de informação, representadas pelas seções de tabelas, foi implementada a classe `TableParser`. As requisições de tabelas de serviços de informação são encaminhadas à classe `TableManager`. Os PES de dados são processados pelo componente de software DSM-CC, o qual possui `DsmccManager` como elemento principal. O diagrama de classe do módulo PFD é apresentado na Figura 18.

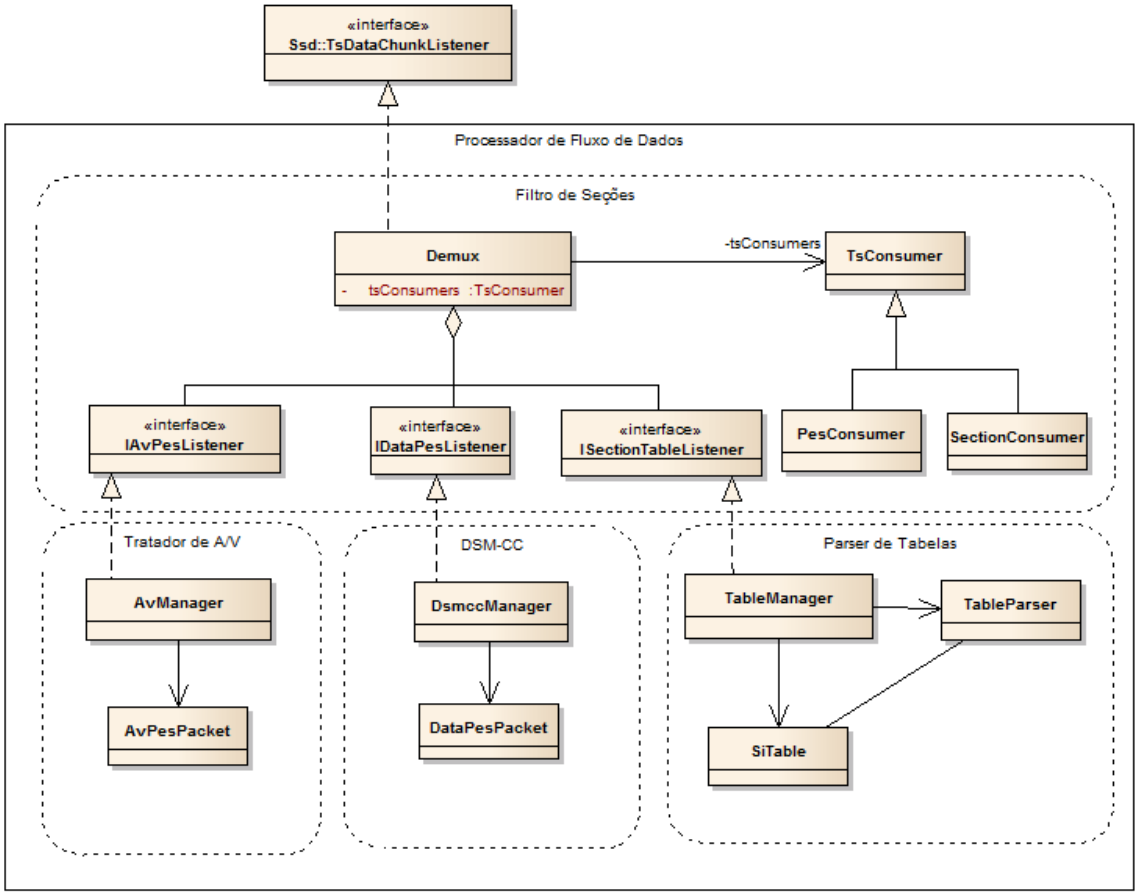

Figura 18: Diagrama de Classe – Processador de Fluxo de Dados

Aplicações Symbian dependem de um conjunto de componentes do sistema operacional para operar. Diversos requisitos comuns para desenvolvimento de aplicações, como manipulação de desenhos na tela, não necessitam serem programadas por desenvolvedores de aplicações, pois são providas pelo sistema operacional através de processos servidores. O servidor de janelas (*Window Server*) de Symbian provê suporte extensivo para o desenvolvimento de componentes de interfaces gráficas.

Conforme apresentado por CRUZ (2008), em Symbian há dois elementos básicos para o desenvolvimento de interfaces gráficas: *Control* e *Window*. Cada *Control* deriva da classe abstrata `CCoeControl`, e representa uma área retangular da tela, capaz de receber entrada de dados do usuário, ou exibir dados de saída do sistema; objetos da classe `CCoeControl` podem ser aninhados em outros *Controls*. Um elemento *Window* é um recurso de sistema, pertencente ao servidor de janelas, que representa uma área retangular da tela. Cada *Window*

contém um ou mais *Controls*, e provê mecanismos para exibição de seus *Controls* na tela.

O servidor de janelas Symbian atende aos requisitos definidos pelo módulo Gerenciador Gráfico; portanto, foi utilizado para prover as funcionalidades deste módulo.

Os exibidores de mídias foram modelados conforme ilustrados pelo diagrama de classe da Figura 19. `ExhibitorBase` é uma generalização de exibidores, e provê um conjunto de métodos comuns a todos os exibidores de mídias. As especializações estendem de `ExhibitorBase`, e sobrescrevem os métodos específicos para cada exibidor. Como podem ser observadas na Figura 19, as especializações mantêm relacionamentos com as classes Symbian que provêm funcionalidades específicas da plataforma. `AudioExhibitor` e `VideoExhibitor` fazem uso das APIs de `CAudioPlayerUtility` e `CVideoPlayerUtility` respectivamente. `XhtmlExhibitor` utiliza APIs do sistema para manipulação de arquivos XHTML (`CBrCtlInterface` e `CBrCtlLinkResolver`). A exibição de imagens é tratada por objetos da classe `ImageExhibitor`, a qual se relacionada com as classes para manipulação de imagem em Symbian (`CImageDecoder`, `CFbsBitmap`, e `CGulIcon`).

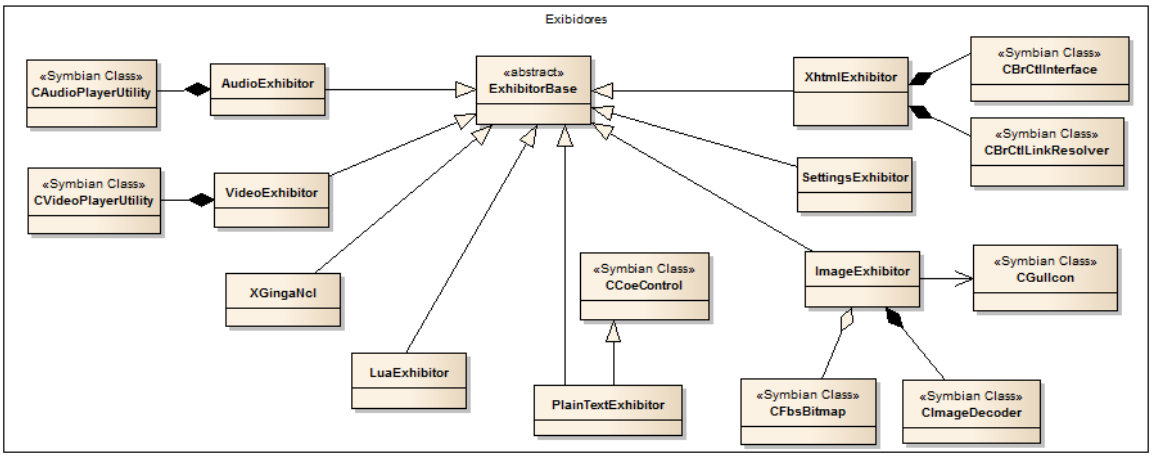

Figura 19: Diagrama de Classe – Exibidores

Pelo fato de objetos NCLua serem tratados com mídias em NCL, a classe `LuaExhibitor` foi modelada e implementada. A classe `LuaExhibitor` se relacionada com o módulo Máquina Lua, e provê interfaces de programação para interpretação de *scripts* NCLua. `PlainTextExhibitor` estende de

`CCoeControl` de Symbian e provê interfaces para apresentação de textos simples, sem formatação de fonte.

A Máquina Lua foi portada para Symbian pelo Instituto Nokia de Tecnologia (INdT) a partir do código disponibilizado em LUA (2010). As extensões Lua para TV digital foram divididas em quatro módulos (apresentados na Subseção 2.3.2). O módulo *Lua Canvas*, o qual provê funcionalidades para manipulação de desenhos em regiões da tela, foi modelado na classe `LuaCanvas`. `LuaCanvas` herda de `CCoeControl`, e faz o mapeamento das diretivas de desenhos *Canvas* em chamadas de métodos do sistema. As funcionalidades do módulo *Lua Persistent* são garantidas pela classe `LuaPersistent`, a qual faz uso de serviços do servidor de arquivos Symbian através da classe `RFs`. O módulo *Lua Settings*, que exporta uma tabela com variáveis contidas no nó *application/x-ginga-settings* do documento NCL, foi modelado na classe `LuaSettings`.

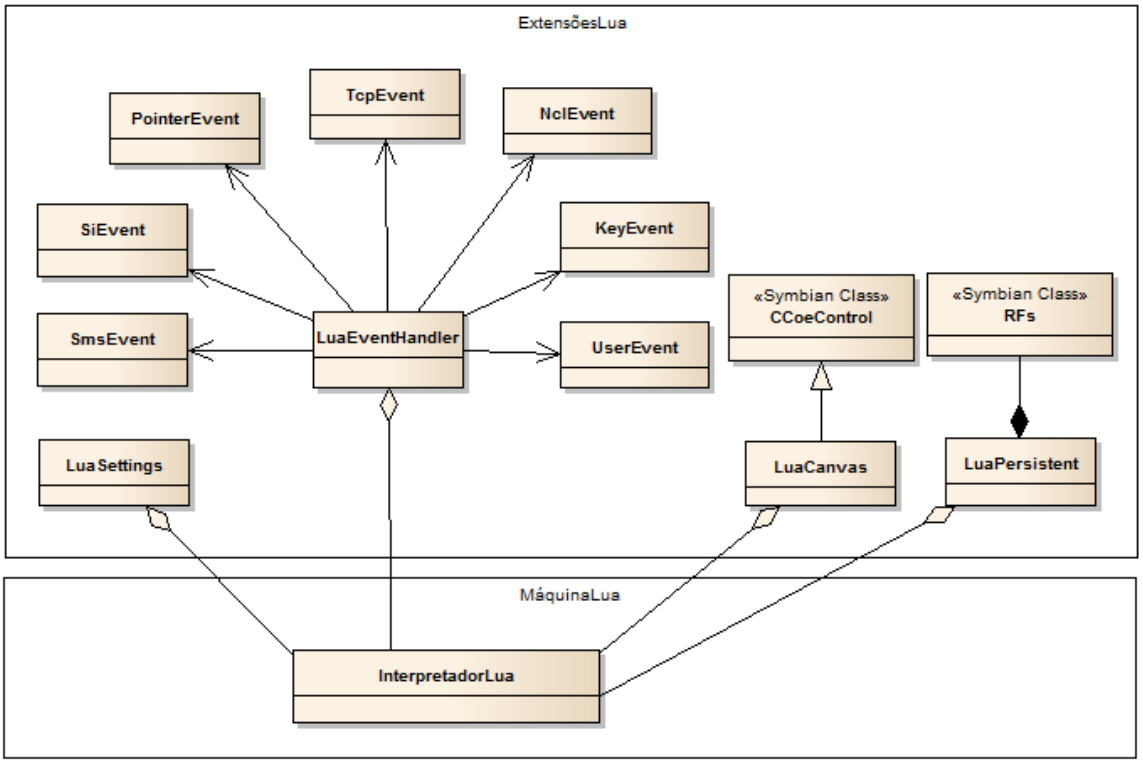

Figura 20: Diagrama de Classe – Extensões Lua e Máquina Lua

As responsabilidades do módulo *Lua Event* são garantidas pela classe `LuaEventHandler`, a qual faz o tratamento dos eventos Lua definidos em *scripts* NCLua. Para cada classe de evento Lua foi definida uma classe Symbian correspondente, como pode ser observado pelo diagrama de classe da Figura 20.

4.3.2 Máquina de Apresentação

O formatado NCL foi definido como componente núcleo para subsistema Máquina de Apresentação do simulador SGDP. Ele é composto por elementos fundamentais que garantem a apresentação adequada de aplicações interativas NCL. O diagrama de classe da Figura 21 exibe o relacionamento entre os elementos que compõem formatador NCL.

A classe `NclFormatter` provê um conjunto de métodos para o gerenciamento das classes principais dos componentes de software que formam a Máquina de Apresentação NCL (as classes `ExibitorManager`, `LayoutManager`, `NclConverter`, `EventScheduler`, e `NclContextManager` são apresentadas posteriormente).

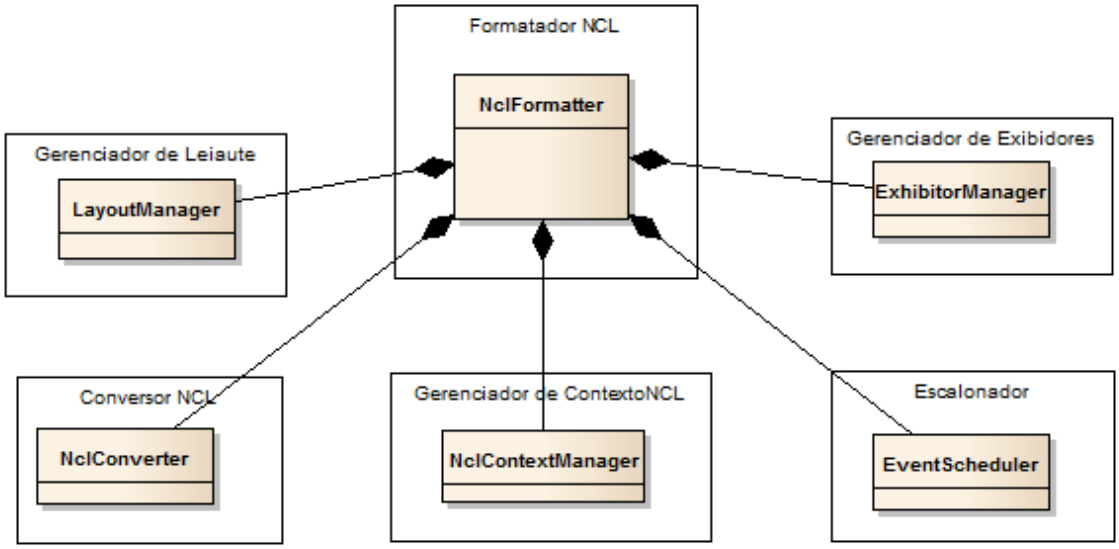

Figura 21: Diagrama de Classe – Formatador NCL

A conversão de documentos NCL em estruturas Symbian C++ foi realizada através de APIs SAX; onde para cada elemento XML, da linguagem NCL, foi criada uma classe Symbian C++ correspondente. Para o gerenciamento de contextos NCL, as estruturas representativas dos elementos NCL foram modeladas em conjunto de classes conforme suas áreas funcionais. Os nós NCL foram mapeados em classes Symbian C++ conforme ilustrado no diagrama de classe da Figura 22.

A interface `IEventListener` define métodos abstratos para tratamento de eventos de transição NCL. `IActionTarget` provê interfaces para tratar ações provenientes de elementos do tipo *action*. `INode` é a interface base para implementação dos nós de mídia, contexto, e escolha. A classe `BaseNode` implementa as funcionalidades básicas comuns aos nós NCL. `BaseCompositeNode` trata-se de uma especialização de `BaseNode` para nós de composição; sua classe irmã, `ExibitorBase`, define as funcionalidades básicas para os exibidores de mídias, de onde cada classe exibidora deriva. Os nós de contexto e de escolha são representados respectivamente pelas classes `ContextNode` e `SwitchNode`. `ParamStore` é uma classe auxiliar que estende de `IParamProvider`, e provê métodos para armazenamento e recuperação de parâmetros NCL.

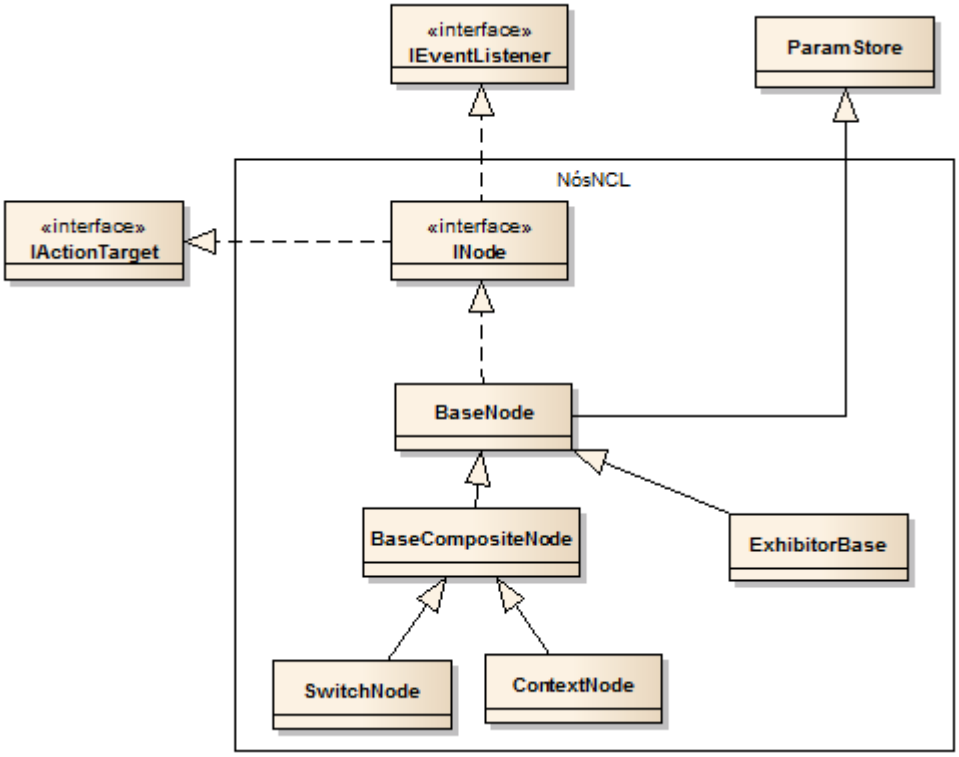

Figura 22: Diagrama de Classe – Nós NCL

Os nós NCL podem conter interfaces NCL para recebimento de ações, ou monitoramento de propriedades e conteúdos. As interfaces da linguagem NCL âncora de conteúdo, âncora de propriedade, e porta foram mapeadas, respectivamente, nas seguintes estruturas Symbian C++: `ContentAnchor`, `PropertyAnchor`, e `Port`. Para notificações de alterações nas propriedades e

nos conteúdos NCL, foram especificadas, respectivamente, as interfaces Symbian `IPropertyAnchorListener` e `IContentAnchorListener`. Pelo fato de interfaces NCL serem capazes de receber ações, suas classes realizam `IActionTarget`. A classe `BaseEvent` disponibiliza métodos para registro e notificação de eventos NCL, seguindo o modelo de máquina de estados apresentada na subseção 2.3.1. A Figura 23 apresenta o diagrama de classe do mapeamento das interfaces NCL.

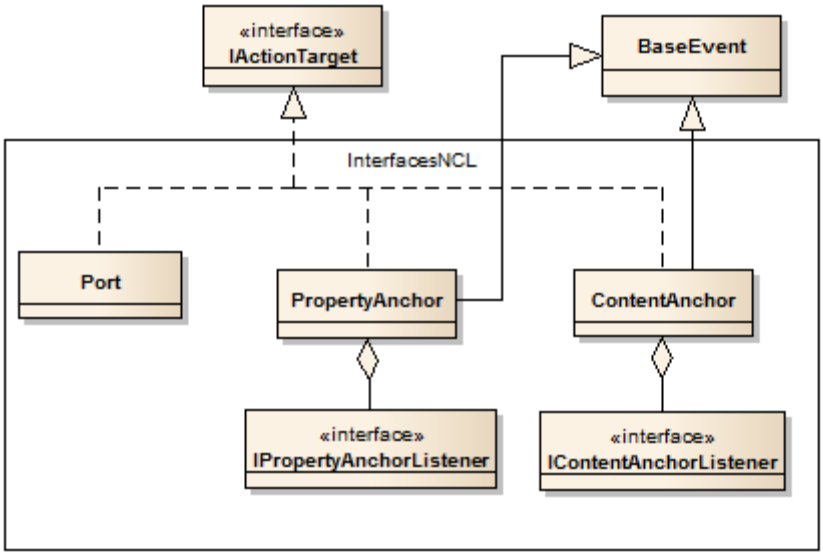

Figura 23: Diagrama de Classe – Interfaces NCL

Conforme apresentado na subseção 2.3.1, as relações entre os elementos NCL são baseadas em eventos, e seguem um modelo de máquina de estados finito. Eventos ocorrem após certas condições serem satisfeitas; portanto, deve-se fazer uma associação (*bind*) entre o evento e a condição de sua ocorrência. Uma vez que a condição é satisfeita, deve-se associar a ação de ocorrência ao nó de destino. Portanto, para garantir esse comportamento, foi implementado um conjunto de classes, que juntas são responsáveis por realizarem o monitoramento das condições de eventos, bem como associar as ações de ocorrência aos nós de destino. A Figura 24 mostra o diagrama de classe dos elementos envolvidos neste processo.

`Condition` representa uma classe base para monitoramento de condições de ocorrência de eventos. `SimpleConditionMonitor` e `CompoundConditionMonitor` são especializações de `Condition` para monitoramento de condições simples e compostas, respectivamente. As

especializações recebem notificações de eventos, através da interface `IEventListener`, e avaliam suas respectivas condições. Para recebimento de notificação de condições satisfeitas, foi definida a interface `IConditionListener`.

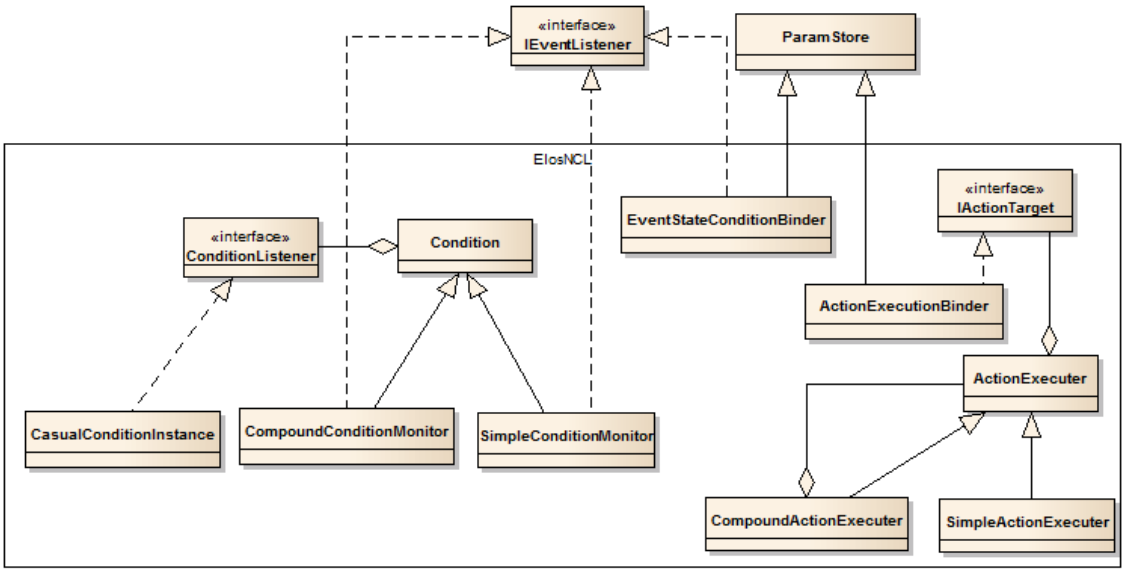

Figura 24: Diagrama de Classe – Elos NCL

As classes `SimpleActionExecuter` e `CompoundActionExecuter` são responsáveis por executar as ações NCL simples e compostas, respectivamente. Ambas as classes estendem da classe abstrata `ActionExecutor`, a qual contém uma lista de `IActionTarget`. A conexão entre as condições NCL com suas respectivas ações são garantidas pela classe `CasualConnectorInstance`. `EventStateConditionBinder` faz a associação entre as condições NCL com os elementos NCL (nós e interfaces). `ActionExecutorBinder` é responsável por associar as ações NCL com os elementos alvo da ação.

Em NCL, nós de escolha (*switch nodes*) fazem uso de regras NCL para identificar a ocorrência de condições NCL, e tomar decisões sobre qual nó de escolha deve ser iniciado. As regras NCL foram modeladas em classes Symbian C++ conforme ilustrado no diagrama de classe da Figura 25.

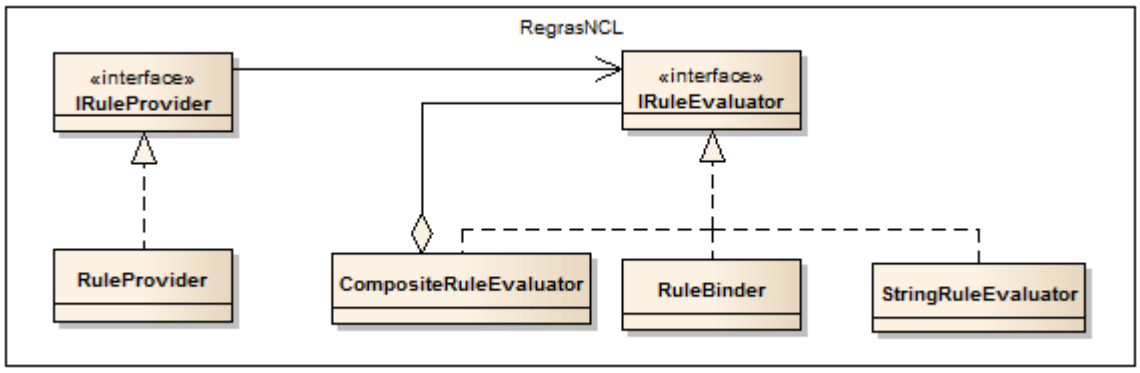

Figura 25: Diagrama de Classe – Regras NCL

A interface `IRuleEvaluator` especifica o comportamento esperado de um avaliador de regras NCL. Sua realização `StringRuleEvaluator` é responsável por avaliar regras comuns do tipo *string*. Regras complexas são avaliadas por `CompositeRuleEvaluator`. `RuleProvider` realiza a interface `IRuleProvider`, e faz o mapeamento de todas as regras contidas no documento NCL. `RuleBinder` realiza as associações (*binds*) das regras NCL.

As funcionalidades do módulo Escalonador são providas através da classe `EventScheduler`, uma especialização de `CActive` de SymbianC++. Assim como todo objeto ativo Symbian, `EventScheduler` se registra no escalonador do sistema através do método estático `CActiveScheduler::Add(CActive* aActiveObject)`.

`EventScheduler` contém uma lista de ouvintes de eventos (`IEventSchedulerListener`), os quais são responsáveis por propagar notificações de eventos NCL aos elementos interessados (realizações de `IEventListener`). `EventState` indica o estado atual do evento (*occuring, sleeping, paused*). `Transition` representa o tipo de transição do evento (*start, stop, pause, resume, abort,* e *repeat*). O diagrama de classe da Figura 26 ilustra as classes que formam o módulo Escalonador.

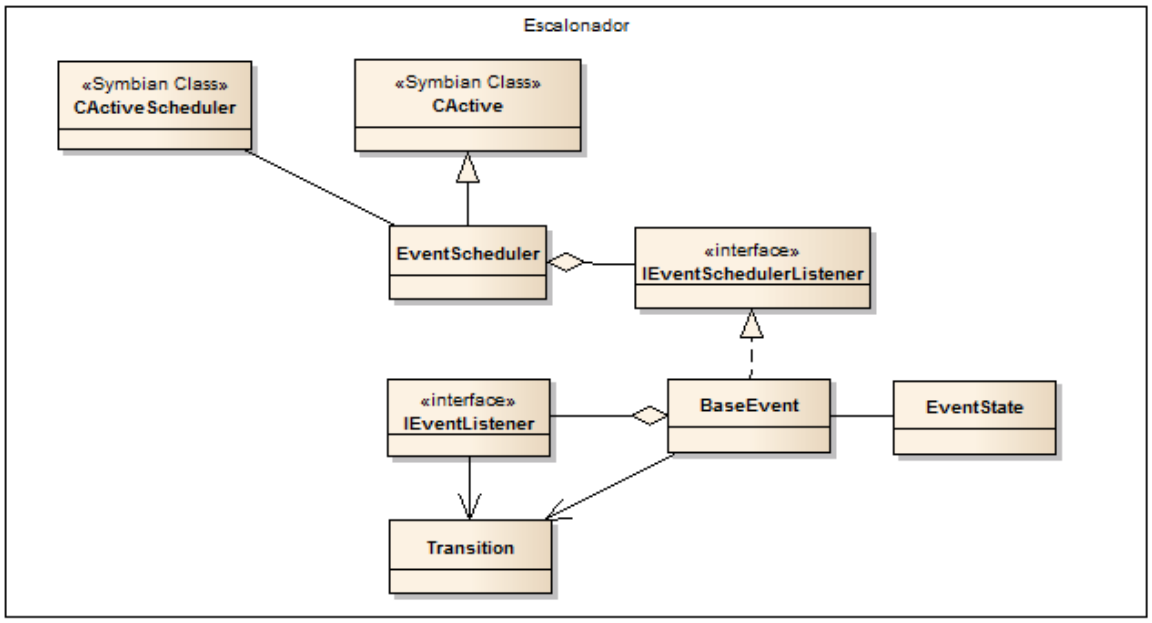

Figura 26: Diagrama de Classe – Escalonador

A classe `ExhibitorManager` foi definida para gerenciar o ciclo de vida dos exibidores de mídias. Os respectivos exibidores são instanciados com base no tipo MIME definido no nó de mídia do documento NCL, ou em sua ausência, com base na extensão do arquivo de mídia. A Figura 27 mostra o relacionamento entre o Formatador NCL, o Gerenciador de Exibidores, e o módulo Exibidores.

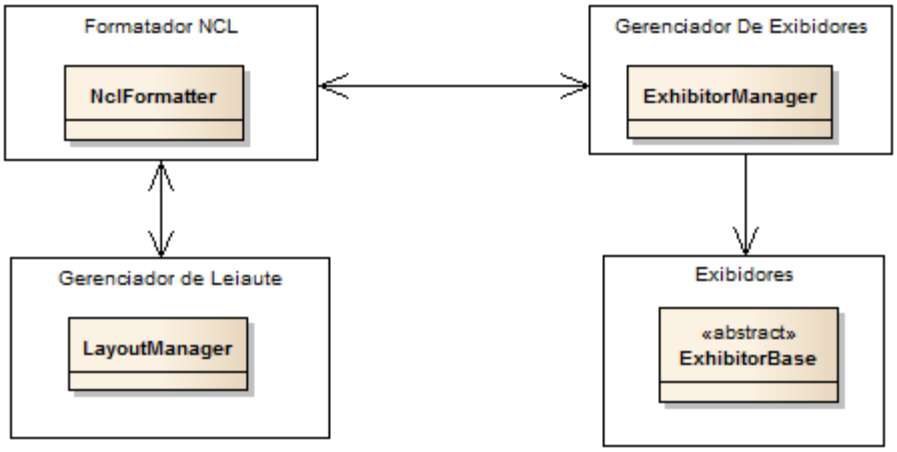

Figura 27: Relacionamento entre os módulos da Máquina de Apresentação

O formatador requisita ao Gerenciador de Exibidores a criação de um exibidor específico para um objeto de mídia; o gerenciador identifica o tipo de mídia, e solicita ao módulo Exibidores uma instância do respectivo exibidor. Após o recebimento, o gerenciador retorna ao formatador o exibidor específico requisitado.

Para a implementação do Gerenciador de Leiaute, foi definida a classe `LayoutManager`, a qual prover métodos para criação, e gerenciado de regiões de tela para apresentação dos objetos de mídia. `LayoutManager` também é responsável por converter os índices Z (*Z-index*) da linguagem NCL em índices Z dos componentes gráficos Symbian (*Window* e *Controls*) utilizados pelos exibidores de mídia.

As funcionalidades especificadas pelo módulo Orientador de Tela são garantidas através do construtor base `BaseConstructL`() da classe Symbian `CAknAppUi`. `BaseConstructL()` recebe como argumento uma combinação de informações (*flags*) para construção da interface gráfica da aplicação Symbian. Os valores referentes à orientação de tela podem ser do tipo:

- `EAppOrientationPortrait`: indicando ao sistema que a aplicação deve ser renderizada somente na orientação horizontal.
- `EAppOrientationLandscape`: para renderização vertical dos componentes gráficos.
- `EAppOrientationAutomatic`: indicando ao sistema que o controle de orientação de tela é automático, onde os componentes gráficos são reposicionados, e redimensionados a cada mudança de orientação.

4.3.3 Logger

A implementação do subsistema *Logger* foi realizada através de interfaces do componente de software Persistência. Uma das maiores dificuldades encontrada durante a implementação deste subsistema está relacionada à definição de informações a serem registradas no arquivo de *log*. Porém, com base no validador de documentos NCL, da ferramenta de edição NCL Eclipse 1.6 (NCLECLIPSE, 2011), e estudos sobre as linguagens NCL (SOARES e BARBOSA, 2007) e Lua (SANT'ANNA *et al*, 2008) foram definidos os registros das seguintes informações:

- Erros de gramática em documentos NCL: através da indicação de erros de gramática, autores de documentos NCL podem identifica, de forma rápida e precisa, os motivos pelos quais sua aplicação interativa não está sendo apresentada adequadamente na tela do simulador.
- Erros de gramática em *scripts* NCLua: de forma análoga aos erros de gramática em documentos NCL, a indicação de erros de má-formatação em scripts NCLua devem ser registrados e disponibilizados aos autores de obras interativas, para que possam ser prontamente corrigidos
- Erros de execução em *scripts* NCLua: estes tipos de erros são identificados em tempo de execução dos *scripts*, e podem afetar negativamente o comportamento da apresentação do conteúdo interativo. Portanto é essencial que o simulador de interatividade registre erros desta categoria.
- Ausência de mídias referenciadas pelos documentos NCL e scripts NCLua: as linguagens NCL e Lua permitem acesso a mídias (arquivos de texto, imagens, áudio, vídeo, etc.), através de referências codificadas em seus documentos declarativos (NCL), e scripts (Lua). Portanto, ao apresentar o conteúdo interativo no simulador, erros de referências podem ocorrer, como por exemplo: ausência da mídia no sistema de arquivos, indicação errada de caminhos onde as mídias encontram-se, etc. Ao registrar essas informações, autores de aplicações interativas tem a possibilidade de identificar possíveis erros de referência de mídias em suas obras.
- Eventos NCL (*start, stop, pause, resume, abort*): os eventos NCL seguem o modelo de máquina de estado apresentado na Figura 7 (Seção 2.3.1). Ao registrar os eventos e as transições de estados, autores de conteúdos interativos tem a possibilidade de verificar se suas obras estão se comportando conforme especificado pelo documento NCL.
- Tempo decorrido para baixar uma aplicação interativa do fluxo de transporte: O sinal digital dedicado à recepção móvel (*oneseg*) apresenta limitações em sua largura de banda. Portanto, ao elaborar uma aplicação interativa, o tamanho (em bytes) da solução deve ser cuidadosamente planejado, pois impacta diretamente no tempo de *download* pelo terminal receptor.

Com base nas informações de registro de processamento e apresentação do conteúdo interativo, desenvolvedores de aplicações interativas tem a possibilidade de identificar possíveis problemas em suas obras, e corrigi-las adequadamente.

4.3.4 Resumo Comparativo entre SGDP, IRGDP e IRGDF

Embora CRUZ (2008) tenha realizado um ótimo trabalho ao propor o Ginga-NCL para Dispositivos Portáteis (GDP), alguns módulos definidos em sua arquitetura de *middleware* não foram contemplados pela Implementação de Referência para Dispositivos Portáteis (IRGDP). Por conseguinte, não foi possível validar, em sua solução, aplicações interativas em conformidade com o padrão brasileiro de TV digital, principalmente as que fazem uso de objetos de mídias NCLua. Além da ausência de suporte a objetos NCLua, a IRGDP não provê funcionalidades para extração de conteúdos interativos a partir dos fluxos de transporte MPEG-2. A implementação do simulador SGDP contempla as funcionalidades principais de um *middleware* de TV digital, relacionadas ao fluxo de processamento do conteúdo interativo até sua apresentação na tela do terminal portátil. As Tabelas 5, 6, e 7 exibem comparações entre os módulos do *middleware* Ginga-NCL implementados pelas soluções IRGDF, IRGDP e SGDP. A interpretação dos símbolos apresentados nas Tabelas 5, 6, e 7 seguem as definições de legenda da Tabela 4.

Símbolos	Descrição
-	Módulo **não** implementado
+	Módulo **parcialmente** implementado
++	Módulo **totalmente** implementado
	Soluções para dispositivos portáteis

Tabela 4: Legenda para interpretação das tabelas 4, 5 e 6.

Módulos que compõem a Máquina de Apresentação	IRGDF	IRGDP	SGDP
Conversor	++	++	++
Formatador	++	++	++
Escalonador	++	++	++
Gerenciador de Bases Privadas	+	-	+
Orientador de Tela	-	-	++
Gerenciador de Leiaute	++	++	++
Gerenciador de Exibidores	++	++	++
Gerenciador de Contexto NCL	++	++	++

Tabela 5: Comparação entre as Máquinas de Apresentação

Pela Tabela 5 observa-se que todos os módulos que compõem a arquitetura básica do subsistema Máquina de Apresentação foram contemplados em sua totalidade pela solução SGDP, com exceção do módulo Gerenciador de Bases Privadas, o qual foi contemplado parcialmente por não haver necessidade em gerenciar bases privadas de outros canais digitais, uma vez que o simulador suporta apenas um canal.

O módulo Orientador de Tela, não contemplados pelas soluções IRGDF e IRGDP, foi implementado em sua totalidade na solução SGDP. Este módulo é essencial para dispositivos móveis, principalmente pelas características de manuseio destes tipos de dispositivos, onde as telas podem se encontrar nas posições verticais, ou horizontais, durante o início da aplicação interativa; além de estarem sujeitos a mudanças dinâmicas de orientação de tela durante a apresentação do conteúdo interativo.

Módulos que Compõem o Núcleo Ginga		IRGDF	IRGDP	SGDP
Sintonizador	*Fullseg*	+	-	-
	Oneseg	+	-	+
Processador de Fluxo de Dados	Filtro de Seções	++	-	++
	DSM-CC	++	-	+
	Fluxo Principal de A/V	++	-	++
	Parser de Tabelas	++	-	++
Persistência		++	+	++
Exibidores		++	+	++
Transporte		+	+	+
Gerenciador de Contexto		+	-	+
Gerenciador Gráfico		++	++	++
Gerenciador de Atualizações		-	-	-
Máquina Lua		++	-	++
Extensão Lua para TV Digital		++	-	++
Máquina Virtual Java		++	-	-

Tabela 6: Comparação entre os Núcleos de Processamento

O grande diferencial entre a solução de CRUZ (2008) e as melhorias apresentadas pela implementação SGDP está no subsistema Núcleo Ginga, como pode ser observado na Tabela 6.

Por serem soluções voltadas para receptores portáteis, IRGDP e SGDP, não contemplam o submódulo Sintonizador *Fullseg*, o qual é coberto parcialmente pela solução IRGDF. Entretanto, para prover as funcionalidades de recepção do sinal digital *oneseg*, SGDP implementa o módulo Simulador de Sinal Digital apresentado na subseção 4.1. Os submódulos que compõem o Processador de Fluxo de Dados foram implementados em sua totalidade, com exceção do submódulo DSM-CC, onde o suporte a comandos de edição não foram cobertos; sendo deixados como trabalhos futuros.

Os módulos Persistência, Exibidores, e Transporte, contemplados parcialmente pela implementação IRGDP, foram implementados no simulador SGDP. Sendo o exibidor de mídias do simulador capaz de processar e apresentar

fluxos de vídeo ao vivo (simulado), bem como objetos NCLua. O módulo Transporte provê serviços de envios de mensagens de texto SMS definidos no módulo de eventos da linguagem de extensão Lua. Diferentemente da implementação IRGDP, a solução SGDP provê as funcionalidades da linguagem Lua para TV digital, através do porte da Máquina Lua para Symbian, bem como da implementação dos módulos de extensão da linguagem.

Por não haver necessidade em atualizar os componentes de softwares através de dados enviados pelo sinal digital, o Gerenciador de Atualizações não foi contemplado. A Máquina Virtual Java, presente na solução para dispositivos fixos, não foi implementada, pois o simulador proposto tem como foco apenas aplicações NCL/Lua.

Diferentemente da solução IRGDP, SGDP possui suporte a registros de informações (arquivos de *log*) sobre a apresentação de aplicações NCL. A Tabela 7 exibe o suporte de registros de *log* das soluções apresentadas.

Logger	**IRGDF**	**IRGDP**	**SGDP**
Logger	+	-	+

Tabela 7: Comparação entre os mecanismos de *log*

4.4 Conclusão sobre o Simulador SGDP

Neste capítulo foi apresentada a solução SGDP, a qual propôs um simulador de TV digital para dispositivos portáteis, segundo padrão brasileiro de TV digital. A arquitetura proposta foi implementada para a plataforma Symbian, onde os detalhes de implementação foram apresentados.

Após a implementação, foi realizada uma análise comparativa entre as soluções existentes (Subseção 4.3.4) e observou-se que o simulador SGDP atende às necessidades de simulação para este nicho tecnológico.

A adição do módulo Orientador de Tela ao subsistema Máquina de Apresentação possibilitou a apresentação de aplicações interativas nas em orientações de tela no modo retrato ou paisagem. O Simulador de Sinal Digital

(SSD) adicionado ao Núcleo Ginga permitiu a simulação de processamento de recepção de sinal, onde foi possível extrair o conteúdo interativo do fluxo de transporte através do módulo Processador de Fluxo de Dados (PFD).

O porte da Máquina Lua, pelo Instituto Nokia de Tecnologia (INdT), para a plataforma Symbain, possibilitou a integração do interpretador Lua ao simulador SGDP, provendo assim suporte a interpretação de scripts Lua embutidos nas aplicações NCL.

O capítulo a seguir discorre sobre os testes realizados durante a validação da solução proposta.

5 Testes Sistêmicos

Este capítulo tem como objetivo apresentar os testes realizados para validação do simulador proposto. Para isso, dentre as técnicas de testes de software, foi utilizada a técnica de testes sistêmicos. Nesta técnica, ao invés de validar cada funcionalidade do sistema individualmente, um conjunto de funcionalidades é testado como um todo.

Para realização dos testes sistêmicos foram utilizadas três aplicações NCL/Lua desenvolvidas pelo Instituto Nokia de Tecnologia (INdT). A primeira aplicação simula um ambiente de eleição, onde é possível selecionar o candidato a ser votado através de opções de seleção da aplicação. A segunda aplicação exibe informações sobre uma partida de futebol, onde é possível saber, por exemplo, a escalação dos times que disputam a partida. A terceira e última aplicação de teste faz uso de mensagens de textos SMS para eliminação de participantes de um *reality show*. Os testes foram realizados em três dispositivos portáteis com características distintas, os quais são apresentados nas subseções a seguir.

5.1 Aplicação NCL: Eleições 2010

A aplicação interativa Eleições 2010, apresentada na Figura 28, faz uso extensivo dos objetos de mídias (imagem, fluxo de vídeo ao vivo, textos, e *scripts* NCLua). Eleições 2010 possui uma tela principal com opções de escolha para votos em governadores e presidente da república. Ao selecionar o candidato, o voto do usuário de TV é enviado, via canal de interatividade, ao servidor do provedor de conteúdo interativo.

A aplicação Eleições 2010 foi testada num celular Nokia N85, o qual possui um processador de capacidade de 369 MHz e 128MB de memória RAM. Os resultados foram satisfatórios, embora as transições entre as telas que compõem a aplicação tenham sofrido atrasos não definidos pelo autor do conteúdo interativo. Neste dispositivo foi possível observar a aplicação nos dois formatos de orientação de tela (paisagem e retrato).

Figura 28: Aplicação Eleições 2010 apresentada num Nokia N85

5.2 Aplicação NCL: Copa 2010

A aplicação interativa Copa 2010 disponibiliza ao usuário telespectador informações referentes à partida de futebol transmitida pela emissora de TV. Nesta aplicação é possível obter, dentre outras informações, a escalação dos times, as tabelas de jogos, bem como adquirir ingressos para as próximas partidas. Com o intuito de validar as funcionalidades da aplicação Copa 2010, o simulador SGDP foi instalado num celular Nokia 5800 XpressMusic, cujo processador tem a capacidade de 434 MHz, e possui 128MB de memória RAM. A Figura 29 exibe a tela principal da aplicação sendo apresentada pelo simulador SGDP num celular Nokia 5800.

Diferentemente do N85, o celular Nokia 5800 não possui teclado físico, onde o principal mecanismo de entrada de dados do usuário no sistema é realizado através de eventos de toque na tela do dispositivo. Portanto foi necessário mapear as funcionalidades de teclas, definidas pela norma brasileira (ABNT-N06, 2007) num teclado virtual, representado pelo mapeamento de regiões de tela no dispositivo.

Figura 29: Aplicação Copa 2010 apresentada num Nokia 5800

5.3 Aplicação NCL: *Reality Show*

A aplicação *Reality Show* provê a usuários de TV a capacidade de votar em participantes de programas desta categoria, onde, por exemplo, um candidato pode ser eliminado do programa através da soma dos votos dos telespectadores. O voto é realizado de maneira simples, bastando que o usuário telespectador pressione uma única tecla, e confirme seu voto. O voto é enviado ao servidor de conteúdo interativo através de mensagens de texto SMS.

Figura 30: Aplicação *Reality Show* apresentada num Nokia N97

Para validar as funcionalidades da aplicação, o simulador de interatividade móvel foi instalado num celular Nokia N95, com capacidade de processamento de 434 MHz, e memória RAM de 128MB. A Figura 30 exibe a tela principal da

aplicação sendo apresentada pelo SGDP. Dentre os dispositivos analisados, o celular Nokia N95 mostrou melhor desempenho com relação ao tempo de mudança de transições de tela, onde não foi observador atrasos durante as transições.

5.4 Eleições 2010 nos Celulares: N85, N95, e 5800

Figura 31: Aplicação Copa 2010 em celulares Nokia N97, 5800, e N85

Com o intuito de validar o comportamento da ferramenta durante a apresentação de uma mesma aplicação NCL em dispositivos distintos, o simulador SGDP foi instalado nos celulares Nokia: N97, 5800, e N85. A aplicação interativa escolhida foi Copa 2010. A Figura 31 exibe a apresentação de Copa 2010 nos três dispositivos.

Dentre os terminais de simulação utilizados, N97 demonstrou melhor desempenho ao apresentar a aplicação interativa. Neste dispositivo não foi observado atrasos durante as transições de telas da aplicação. Adicionalmente, o celular N97 possui suporte a entrada de dados através de toque de tela e teclado

físico QWERT; ao contrário dos dispositivos 5800, e N85, onde apenas um tipo de entrada de dados é permitido. A Tabela 8 mostra as especificações básicas dos três dispositivos utilizados para teste.

Dispositivo	Memória		Processador	Entrada de Dados
	Total	Disponível para aplicação		
N85	128 MB	70 MB	369 MHz	Teclado físico
5800 XpressMusic	128 MB	50 MB	434 MHz	Tela sensível ao toque
N97	128 MB	50 MB	434 MHz	Tela sensível ao toque + teclado físico

Tabela 8: Especificações dos dispositivos utilizados

5.5 Validação do *Logger*

Com o intuito de validar as funcionalidades do módulo *Logger*, a aplicação interativa Eleições 2010 (Seção 5.1) sofreu modificações em seu documento original. Nesta esta aplicação, erros de gramática foram inseridos em seu documento de entrada no sistema, onde a etiqueta de encerramento do elemento `<body>` foi removida. Ao testar o documento modificado no simulador, a aplicação interativa não foi renderizada na tela do dispositivo, porém a informação de não conformidade foi registrada no arquivo de registros (arquivo de *log*).

Outra alteração realizada na aplicação Eleições 2010 está relacionada à remoção de mídia referenciada pelo documento NCL, onde o arquivo de imagem do logotipo da aplicação foi excluído do sistema de arquivos. Ao validar o documento NCL, nenhum erro foi detectado pelo simulador; porém, ao tentar exibir a mídia referenciada, o simulador não a localizou no sistema de arquivos; e consequentemente a mídia não foi exibida conforme especificada no documento NCL, porém o registro deste erro foi armazenado no arquivo de registros pelo módulo *Logger*.

Os tempos decorridos para baixar aplicações apresentadas neste capítulo (Eleições 2010, Copa 2010, e *Reality Show*) foram registrados e são apresentados na Tabela 9.

Aplicação	**Tamanho**	**Tempo de *Download*** m = minuto; s = segundo
Eleições 2010	106,0 KB	02m53s
Copa 2010	167,0 KB	03m46s
Reality Show	88,6 KB	02m07s

Tabela 9: Tamanho e tempo de *download* das aplicações de teste

Observa-se pela Tabela 9 que embora as aplicações sejam relativamente pequenas, levando-se em consideração todas as mídias que as compõem, o tempo de *download* é bastante expressivo na transmissão pelo sinal *oneseg*. Portanto é importante que aplicações interativas dedicadas a receptores portáteis sejam planejadas levando-se em consideração essa particularidade de terminais portáteis.

5.6 Conclusão do Capítulo

Neste capítulo foram apresentados os testes realizados durante o processo de validação do simulador. Através do SGDP foi possível validar os comportamentos esperados das aplicações interativas testadas. Embora o simulador instalado no dispositivo N85 tenha conseguido apresentar adequadamente as aplicações interativas, a execução do simulador neste dispositivo mostrou que há a necessidade em melhorar o desempenho do *middleware* Ginga implementado, pois em alguns casos houve atrasos durante transições de tela da aplicação interativa.

Durante as realizações dos testes foi possível observar que os desenhos gráficos, e a usabilidade de aplicações interativas, devem ser modelados levando-se em consideração as características de entrada de dados dos terminais receptores. Outro ponto que deve ser levado em consideração está relacionado ao tamanho (em bytes) da aplicação. Aplicações interativas pesadas podem levar

vários minutos para serem baixadas pelo terminal receptor; neste contexto, o tempo destinado para a apresentação do conteúdo interativo pode ser prejudicado.

Embora testes sistêmicos tenham sido utilizados como forma de validação da ferramenta proposta, outras formas de testes devem ser aplicados em trabalhos futuros, de forma a garantir as funcionalidades individuais dos módulos que compõem o sistema.

O objetivo geral deste trabalho, de propor um ambiente de simulação de TV digital para terminais portáteis, foi alcançado com sucesso, onde a arquitetura do simulador, juntamente com sua implementação, foram validados conforme resultados apresentados neste capítulo..

6 Considerações Finais

Desenvolver conteúdos interativos para terminais de TV digital é uma tarefa desafiadora por causa dos recursos disponíveis, sejam eles relacionados a ferramentas de edição, ferramentas de multiplexação, e até mesmo ao terminal de recepção. Porém, o desafio torna-se ainda maior quando se trata de dispositivos portáteis. Para receptores com essas características, os recursos de *hardware* e largura de banda de transmissão são bastante limitados se comparados com o perfil *fullseg*. Portanto, o simulador SGDP vem contribuir para que desenvolvedores de conteúdos interativos NCL possam contar um ambiente de simulação de TV digital portátil, e se utilizar deste ambiente para identificar possíveis problemas em suas aplicações interativas, de forma a aperfeiçoar os recursos utilizados.

Para manter maior aderência a norma brasileira de TV digital (ABNT-N06, 2007), as arquiteturas dos middlewares de referências para diapositivos fixos e portáteis foram analisados, e utilizados como base para a elaboração do simulador portátil.

Diferentemente de receptores fixos, o simulador SGDP suporta mudanças de orientação de tela, específicas de terminais receptores portáteis. Adicionalmente, o SGDP disponibiliza ao desenvolvedor de conteúdo interativo registros sobre a apresentação de sua aplicação.

6.1 Dificuldades Encontradas

Dentre as dificuldades encontradas durante a elaboração deste trabalho destacam-se:

- Constantes mudanças na norma brasileira de TV digital (ABNT-N06, 2007).
- Falta de documentação sobre as funcionalidades implementadas na solução Ginga-NCL para dispositivos fixos.

- Indisponibilidade do código-fonte da implementação de referência Ginga-NCL para dispositivos portáteis.
- Limitações da plataforma Symbian
    - Necessidade de permissões para acesso a APIs restritas do sistema.
    - Gerenciamento gráfico pesado.
    - Restrição de uso da API XHTML na plataforma Symbian.
- Diferentes mecanismos de entrada de dados para o sistema (teclados de celulares, teclado QWERT, tela sensível ao toque, etc.).
- Ausência de suíte de testes para validação das funcionalidades de implementação do middleware Ginga-NCL.

6.2 Trabalhos Futuros

A proposta inicial do simulador visava contemplar todas as especificações definidas pela norma brasileira de TV digital (ABNT-N06, 2007), porém alguns módulos da arquitetura do simulador SGDP não foram implementados. Como exemplo pode-se citar o módulo que faz o tratamento dos eventos de comando de edição, o qual foi deixado para trabalhos futuros devido sua complexidade de implementação. Outra funcionalidade não contemplada pelo simulador SGDP foi o suporte a múltiplos dispositivos, onde é possível interagir, através de receptores portáteis, com dispositivos fixos, como s*et-top-boxes*, por exemplo.

A implementação do módulo *Logger* pode ser expandida de forma a adicionar mais informações sobre o processamento e apresentação dos documentos interativos. Adicionalmente, pode-se parametrizar este componente para que desenvolvedores de conteúdos interativos tenham a possibilidade de configurar as informações desejadas.

6.3 Resultados Alcançados

Dentre os resultados alcançados, frutos deste trabalho acadêmico, destacam-se:

- Artigos científicos publicados na conferência internacional Euro iTV 2011 (Apêndice A - Publicações)
- Contribuições para Fórum SBTVD em relação o perfil de aplicações interativas destinadas à recepção *oneseg*.
- Ferramenta de simulação de TV digital portátil, disponibilizada no site oficial de aplicativos Nokia (http://store.ovi.com/). Porém com suporte apenas a apresentação de aplicações NCL, onde a simulação no sinal digital *oneseg* não fui publicada.

Os resultados alcançados por este trabalho de pesquisa foram bastante abrangentes e contribuíram de forma significativa para o desenvolvimento da tecnologia de TV digital do sistema brasileiro, principalmente para o perfil portátil (ou perfil *oneseg*).

Referências

ABNT-N06. Associação Brasileira de Normas Técnicas. Norma Brasileira NBR 15606. Televisão Digital Terrestre – Codificação de Dados e Especificações de Transmissão para Radiodifusão Digital. São Paulo, SP, Brasil, Novembro 2007.

ABNT-N07. Associação Brasileira de Normas Técnicas. Norma Brasileira NBR 15607. Televisão Digital Terrestre – Canal de Interatividade. São Paulo, SP, Brasil, Março 2008.

BABIN, Steve. Developing Software for Symbian OS: A Beginner's Guide to Creating Symbian OS v9 Smartphone Applications in C++. Publisher: John Wiley & Sons. Pub. Date: December 14, 2007. Print ISBN: 978-0-470-72570-2. Web ISBN: 0-470725-70-2

BRACKMANN, Christian – Usabilidade em TV Digital / Christian Puhlmann Brackmann; orientador Prof. Dr. Paulo Roberto Gomes Luzzardi. Dissertação de Mestrado em Ciência da Computação - Universidade Católica de Pelotas (UCPel), 2010

CORREA, Rivaldo; GONDIM, Paulo - DIGITAL TELEVISION AND BANKING INCLUSION IN BRAZIL: ALTERNATIVES TO ACCESS TECHNOLOGIES. IEEE – Third International Conference on Digital Society, 2009. ICDS '09. DOI = 10.1109/ICDS.2009.42

CRUZ, Vitor – Ginga-NCL para Dispositivos Portáteis / Vitor Medina Cruz; orientador: Luiz Fernando Gomes Soares – 2008. Dissertação (mestrado) – Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2008.

CRUZ, Vitor Medina; MORENO, Marcio Ferreira; SOARES, Luiz Fernando Gomes. Ginga-NCL: Implementação de Referência para Dispositivos Portáteis. WebMedia'08, Outubro 26/29/2008, Vila Velha, ES, Brasil.

DTV-BR. Site Oficial da TV Digital. http://www.dtv.org.br/index.php/informacoes-tecnicas/paises-que-adotaram-o-isdbtb/
Último acesso em 02/12/2011.

EMMERICH, Paul. Beginning Lua with World of Warcraft Addons. Publisher: Apress. Pub. Date: July 15, 2009. Print ISBN: 978-1-4302-2371-9. Web ISBN: 1-4302-2371-5.

GARTNER, Inc. (NYSE: IT) is the world's leading information technology research and advisory company – http://www.gartner.com/it/page.jsp?id=1372013.
Último acesso em 05/06/2011.

GUIMARÃES, Rodrigo – Composer: um ambiente de autoria de documentos NCL para TV digital interativa / Rodrigo Laiola Guimarães; orientador: Luiz

Fernando Gomes Soares. – Rio de Janeiro: PUC, Departamento de Informática, 2007. Dissertação (mestrado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

IAB, *Interactive Advertising Bureau* Brasil http://www.iabbrasil.org.br/arquivos/doc/Indicadores/Indicadores-de-Mercado-IAB-Brasil.pdf. Acesso em 08/12/2011.

LUA, A Linguagem de Programação. Site Oficial da Linguagem. http://code.google.com/p/alua/.
Último acesso em 01/06/2010

MANHÃES, Marcus; SHIEHS, Pei. Canal de Interatividade: Conceitos, Potencialidades e Compromissos. 23 Agosto 2005. http://www.wirelessbrasil.org/wirelessbr/colaboradores/manhaes_e_shieh/canal_de_interatividade.html.
Acesso em 17/04/2011.

MARGALHO, Mauro; FRANCES, Renato; WEYL, João - Canal de Retorno para TV Digital com Interatividade Condicionada por Mecanismo de Sinalização Contínua e Provisionamento de Banda Orientado a QoS. Latin America Transactions, IEEE (Revista IEEE America Latina), 2007. DOI = 10.1109/TLA.2007.4378530

Ministério das Comunicações – Portal das Comunicações http://www.mc.gov.br/. Acesso em 13/07/2010.

MONTEZ, Carlos; BECKER, Valdecir. TV Digital Interativa: conceitos, desafios e perspectivas para o Brasil. Florianópolis: Ed. da UFSC, 2005. 2ª edição.

MORENO, Marcio – Um Middleware Declarativo para Sistemas de TV Digital Interativa / Marcio Ferreira Moreno; orientador: Luiz Fernando Gomes Soares – Rio de Janeiro: PUC, Departamento de Informática, 2006.

NCL. Site Oficial da Linguagem *Nested Context Language*. http://www.ncl.org.br/pt-br/inicio.
Último acesso em 21/08/2011.

NCLECLIPSE. Site Oficial da Ferramenta NCL Eclipse. http://laws.deinf.ufma.br/ncleclipse.
Último acesso em 13/09/2011.

SANT'ANNA, Francisco; CERQUEIRA, Renato; SOARES, Luiz – NCLua - Objetos Imperativos Lua na Linguagem. ACM - Proceedings of the 14th Brazilian Symposium on Multimedia and the Web 2008, Vila Velha, Brazil, October, 2008.

SCHWALB, Edward M. iTV Handbook: Technologies and Standards. Pub. Date: July 25, 2003. Print ISBN-10: 0-13-100312-7. Print ISBN-13: 978-0-13-100312-5

SOARES, Luiz Fernando Gomes; BARBOSA Simone Diniz Junqueira. Programando em NCL 3.0: desenvolvimento de aplicações para middleware Ginga, TV Digital e Web. Rio de Janeiro: Elsevier, 2009. ISBN: 978-85-352-3457-2

SOARES, Luiz Fernando Gomes; RODRIGUES, Rogério Ferreira e MORENO, Márcio Ferreira. *Ginga-NCL: the declarative environment of the Brazilian digital TV system*. J. Braz. Comp. Soc. [online]. 2007, vol.12, n.4, pp. 37-46. ISSN 0104-6500. http://dx.doi.org/10.1590/S0104-65002007000100005.

SOARES, Luiz; COSTA, Romualdo; MORENO Marcio; MORENO, Marcelo – Multiple Exhibition Devices in DTV Systems – Proceedings of the seventeen ACM international conference on Multimedia 2009, Beijing, China October, 2009.

SOARES, Luiz; MORENO, Marcelo; SANT`ANNA Francisco – Relating declarative hypermedia objects and imperative objects through the NCL glue language. ACM – Proceedings of the 9th ACM symposium on Document engineering, Munich, Germany, 2009. DOI = http://doi.acm.org/10.1145/1600193.1600243.

SOARES, Luiz; RODRIGUES, Rogérios – Nested Context Language 3.0 Part 8 - NCL Digital TV Profiles. Monografias em Ciências da Computação No. 35/06. Departamento de Informática - PUC-Rio, Outubro, 2006. http://www.ncl.org.br/documentos/NCL3.0-DTV.pdf

SOFTWAREPUBLICO. Portal do Software Público Brasileiro. http://www.softwarepublico.gov.br/
Último acesso em 23/02/2012.

SOUZA FILHO, Guido Lemos de; LEITE, Luiz Eduardo Cunha e BATISTA, Carlos Eduardo Coelho Freire. Ginga-J: the procedural middleware for the Brazilian digital TV system. J. Braz. Comp. Soc. [online]. 2007, vol.12, n.4, pp. 47-56. ISSN 0104-6500. http://dx.doi.org/10.1590/S0104-65002007000100006.

SOUZA JÚNIOR, P. J. – LuaComp: uma ferramenta de autoria de aplicações para TV digital. Dissertação de Mestrado em Engenharia Elétrica, Publicação PPGENE.DM 375/09, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 143 p.

TELECO, Inteligência em Telecomunicações – Portal http://www.teleco.com.br/ncel.asp. Acesso em 01/02/2012.

TELECO. Inteligência em Telecomunicações. Seção: Tutoriais Rádio e TV http://www.teleco.com.br/tutoriais/tutorialinteratividade/pagina_4.asp
Acesso em 12/07/2010.

YAMAKAMI, Toshihiko – An Interactivity Model of Mobile Interactive TV: A Oneseg case for Mobile Glue. ACM – Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human 2009, Seoul, Korea, 2009.

Apêndice A – Publicações

Abaixo seguem publicações que foram resultados da pesquisa desenvolvida neste trabalho de dissertação, em ordem cronológica de publicação.

A.1 *GEmPTV: Ginga-NCL Emulator for Portable Digital TV*

SOUZA, Fábio Gomes De; PINTO Luiz Filipe da Silva Souza; LUCENA JR, Vicente Ferreira De. *GEmPTV: Ginga-NCL Emulator for Portable Digital TV*. Second Euro ITV Workshop on Interactive Digital TV in Emergent Economies at EuroITV 2011 in Lisbon, Portugal.

A.2 *SMS as interactive channel for portable digital TV receivers*

PINTO Luiz Filipe da Silva Souza; SOUZA, Fábio Gomes De; LUCENA JR, Vicente Ferreira De. *SMS as interactive channel for portable digital TV receivers*. EuroITV 11 Proceddings of the 9th international interactive conference on Interactive television. DOI: 10.1145/2000119.2000160.

Apêndice B – Mídias Suportadas

Tipo MIME	Extensão de Arquivo	Observações
text/html	htm, html, xhtml	-
text/css	css	a
text/XML	xml	a
text/plain	txt	-
image/bmp	bmp	-
image/png	png	-
image/jpeg	jpg, jpeg	-
image/gif	gif	b
audio/basic	wav	c
audio/mpeg	mpeg, mpg	c
audio/mpeg4	mp4, mpg4	c
audio/aac	aac, mp4, 3gp	c
audio/mp3	mp3	c
audio/mp2	mp2	c
video/mpeg	mpeg, mpg, mpe	d
video/mp4	mp4, mpg4	d
application/x-ncl-NCL	ncl	-
application/x-ginga-NCL	ncl	-
application/x-ginga-NCLua	lua	-
application/x-ginga-settings	(source)	e
application/x-ginga-time	(source)	e
application/x-ncl-time	(source)	e

Tabela 10: Mídias Suportadas

Observações:

a. Carregado se referenciado por documentos html/xhtml.
b. Estático e animado;
c. Limitado pelo codec de áudio de cada dispositivo.
d. Limitado pelo codec de vídeo de cada dispositivo.
e. Elemento definido em *source*

Apêndice D – Manual de Execução do SGDP

Para testar uma aplicação interativa, é necessário instalar o simulador SGDP gratuitamente a partir do site http://store.ovi.com/, e seguir os seguintes passos:

1. Copie os arquivos que compõem a aplicação interativa NCL para algum diretório no celular.
2. Abra o aplicativo de simulação digital, e em seguida pressione o botão “PESQUISAR”.
3. Selecione o arquivo principal correspondente ao seu aplicativo Ginga.

Figura 32: Selecionando a aplicação interativa para apresentação

4. Escolha a orientação de tela desejada através dos botões “VERTICAL” ou “HORIZONTAL”.
5. Pressione o botão “EXECUTAR”.
6. Neste ponto a aplicação interativa é exibida, conforme ilustrado na Figura 33.

Figura 33: Executando a aplicação NCL

7. Pressione o botão de ativação do teclado virtual para interagir com a aplicação NCL.
8. Pressione o botão indicado para acessar as funcionalidades de teclas direcionais e coloridas.
9. Pressione o botão indicado para acessar as funcionalidades de teclas numéricas.

Figura 34: Acionando o teclado virtual

Apêndice E – Dispositivos Compatíveis

Abaixo estão listados os dispositivos compatíveis com o simulador de interatividade portátil SGDP. As especificações dos dispositivos podem ser encontradas em: http://europe.nokia.com.

- Nokia N85
- Nokia 5230
- Nokia 5233
- Nokia 5235
- Nokia 5530 XpressMusic
- Nokia 5800 XpressMusic
- Nokia C5-03
- Nokia C6-00
- Nokia N97
- Nokia N97 mini
- Nokia X6
- Nokia C7-00
- Nokia N8-00
- Nokia E7