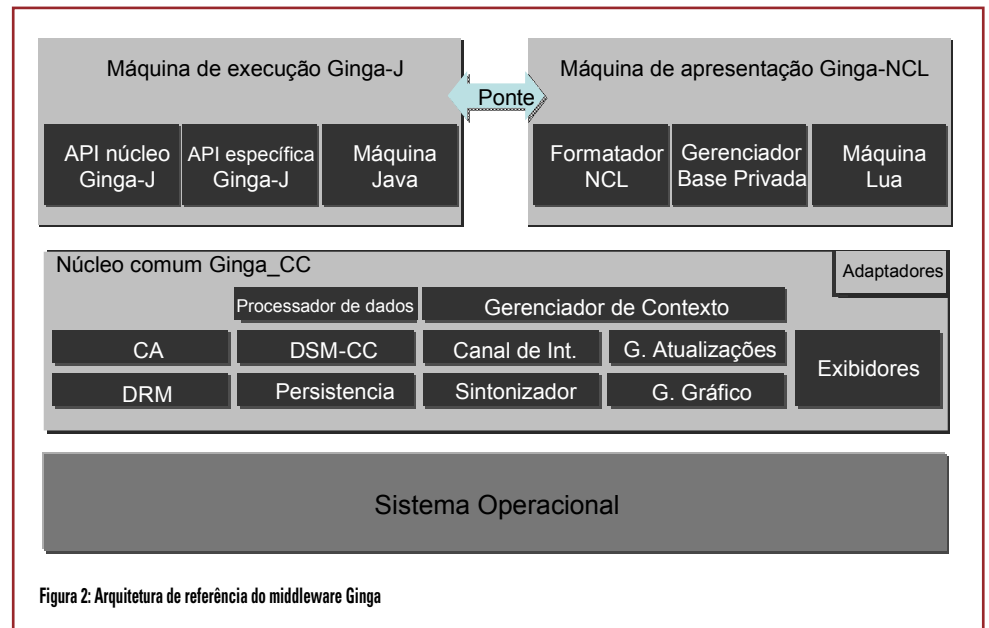


em risco a portabilidade de uma aplicação, e o controle da aplicação é muito mais sujeito a erros cometidos pelo programador. No entanto, nos casos onde o foco de realização de uma tarefa não casa com o foco da linguagem declarativa, o paradigma imperativo é, em geral, a melhor escolha.

Por tudo o mencionado acima, os middlewares para TV digital provêm suporte para o desenvolvimento tanto seguindo o paradigma declarativo quanto o imperativo. Muitas vezes, como é o caso do sistema Japonês, a entidade inicial de uma aplicação é sempre declarativa, mas as outras entidades podem ser codificadas segundo o paradigma imperativo. Muitas vezes, como é o caso do sistema americano e europeu, é oferecido suporte tanto para aplicações declarativas, quanto para aplicações imperativas, mas em ambos os casos, entidades seguindo um paradigma diferente da entidade inicial podem ser definidas.

O ambiente declarativo de um middleware dá o suporte necessário às aplica-



ções declarativas, enquanto o ambiente imperativo dá o suporte necessário às aplicações imperativas. No caso do middleware do padrão brasileiro, os dois ambientes são exigidos nos receptores fixos e móveis, enquanto apenas o ambiente

declarativo é exigido nos receptores portáteis.

O Sistema Brasileiro de TV Digital Terrestre (SBTVD) trouxe como principal inovação seu middleware, denominado Ginga. Por que o nome Ginga? Ginga é

uma qualidade de movimento e atitude que os brasileiros possuem e que é evidente em tudo o que fazem. A forma como caminham, falam, dançam e se relacionam com tudo em suas vidas. Ginga é flexibilidade, é adaptação, qualidades inerentes ao middleware brasileiro.

Arquitetura de Referência

Arquitetura do Ginga pode ser dividida em três módulos principais: Ginga-CC, Ginga-NCL e Ginga-J, como mostra a Figura 2. Os dois últimos módulos compõem a camada de Serviços Específicos do Ginga.

Ginga-J é o subsistema lógico do middleware Ginga responsável pelo processamento de aplicações imperativas escritas utilizando a linguagem Java. A especificação desse subsistema caberá à Norma ABNT NBR 15606-4. Ginga-NCL é o subsistema lógico do middleware Ginga responsável pelo processamento de aplicações declarativas NCL. NCL (Nested Context Language) e sua linguagem de script Lua compõem a

base para o desenvolvimento de aplicações declarativas no SBTVD. Ginga-CC (Ginga Common Core) é o subsistema lógico que provê todas as funcionalidades comuns ao suporte dos ambientes declarativo, Ginga-NCL, e imperativo, Ginga-J. A arquitetura do sistema garante que apenas o módulo Ginga-CC deva ser adaptado à plataforma onde o Ginga será embarcado. Ginga-CC provê, assim, um nível de abstração da plataforma de hardware e sistema operacional, acessível através de APIs (Application Program Interfaces) bem definidas.

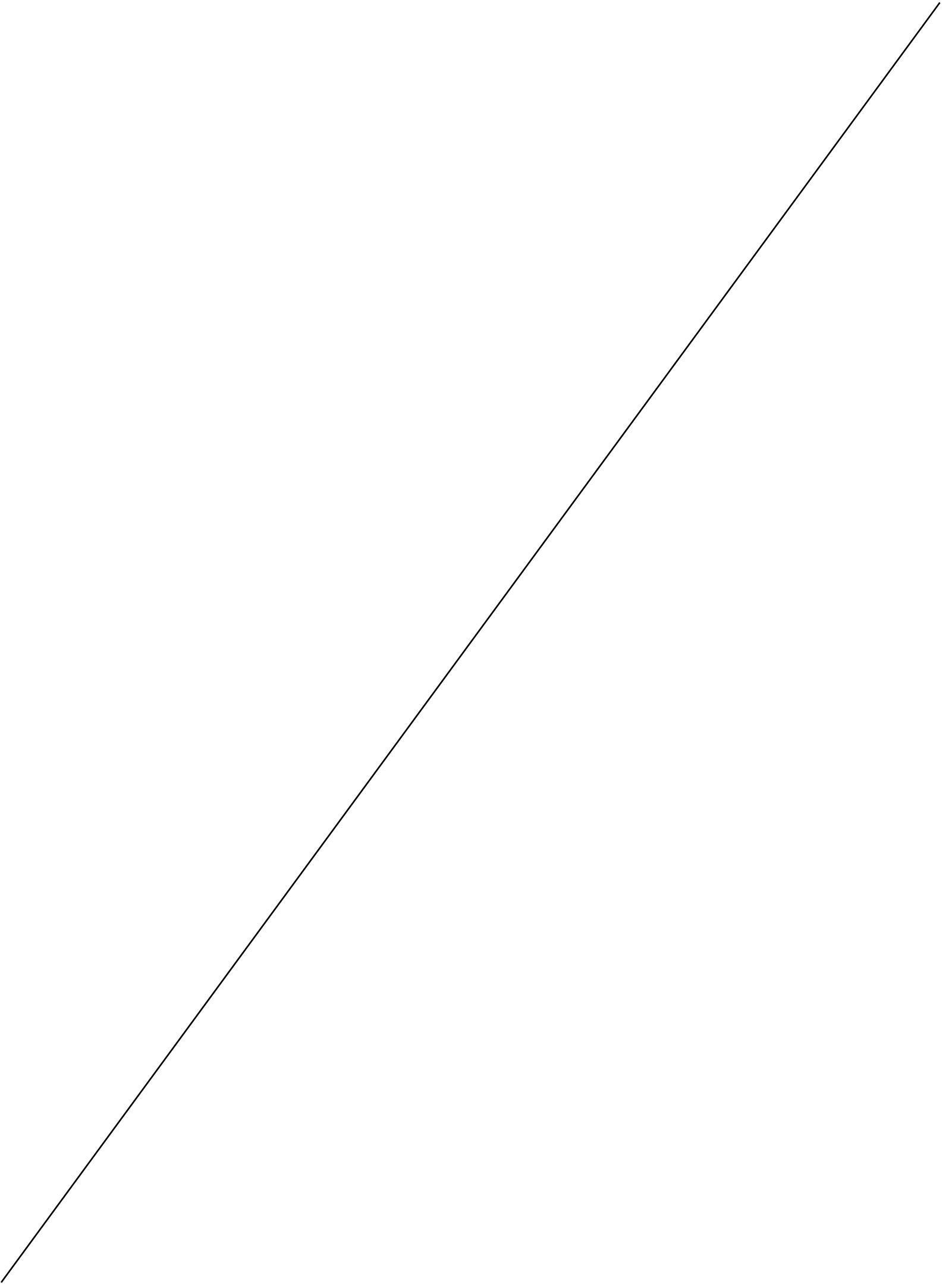
Um conjunto de exibidores monomídia comuns faz parte dos componentes do Ginga-CC. As características de tais exibidores são definidas na Norma ABNT NBR 15606-1. Eles são exibidores de áudio, vídeo, texto e imagem, incluindo entre eles o exibidor MPEG-4/H.264, implementado por hardware. O acesso a tais exibidores se dá através de adaptadores, responsáveis por notificar eventos de apresentação e seleção

(interação do usuário).

Entre os exibidores também se encontra o exibidor (agente do usuário) HTML, especificado nas Normas ABNT NBR 15606-2 e ABNT NBR 15606-5. Na Figura 2, o Gerenciador Gráfico é o responsável pelo gerenciamento do modelo conceitual do plano gráfico de apresentação. É ele que define o plano de exibição do vídeo principal H.264, os planos de exibição dos outros objetos de mídia que compõem uma aplicação TVD, e como esses planos se superpõem. A Norma ABNT NBR 15606-1 é responsável também por tal definição.

Todo o acesso a dados obtidos através do canal de retorno (ou canal de interatividade) é também de responsabilidade do Ginga-CC. As diversas possibilidades de canal de interatividade são especificadas na Norma ABNT NBR 15607.

Ainda na Figura 2, os componentes DSM-CC e Processador de Dados oferecem o suporte para obtenção de dados, obtidos através de carrosséis de



objetos DSM-CC, especificados na Norma ABNT NBR 15606-3. O componente de Persistência é o encarregado pelo gerenciamento do armazenamento de dados requisitados pelas aplicações; enquanto o componente Sintonizador é o responsável pela sintonização e controle do canal de rádio frequência.

Os demais componentes do Ginga-CC são opcionais e dependem da implementação particular de cada receptor. O Gerenciador de Contexto é o encarregado de colher informações do dispositivo receptor, informações sobre o perfil do usuário e sua localização, e torná-las disponíveis ao Ginga-NCL e Ginga-J, para que eles possam efetuar adaptação de conteúdos ou da forma como conteúdos deverão ser apresentados, conforme determinado pelas aplicações.

Ao Gerenciador de Atualizações cabe o controle das atualizações de todo o software residente e do middleware Ginga, durante o ciclo de vida de um dispositivo receptor. Os componentes CA (Conditional Access) e DRM (Digital Right Management) são os responsáveis por determinar os privilégios de acesso às diversas mídias que compõem uma aplicação (programa) TVD.

Ambientes Declarativo e Imperativo do Middleware Ginga

Diferente de outros sistemas, por exemplo o sistema Europeu, não existe qualquer relacionamento mestre-escravo entre os diversos ambientes de aplicação do Ginga. Ao contrário, eles são ambientes pares com processo muito bem definido de comunicação, especificado por APIs simbolizadas na Figura 2 pela Ponte.

O Ambiente Ginga-J

Como já mencionado, o ambiente imperativo Ginga-J oferece suporte a aplicações desenvolvidas usando a linguagem Java.

Ginga-J é dividido em três módulos, conforme ilustra a Figura 2: a máquina virtual Java; o núcleo e suas APIs, também chamadas APIs verde do Ginga-J; e o módulo responsável pelo suporte às APIs específicas do Ginga-J, chamadas de APIs amarela e vermelha do Ginga-J.

Ginga-J seguirá a especificação da Norma ABNT NBR 15606-4. As APIs verde do núcleo são as responsáveis por manter o sistema compatível o máximo possível com os sistemas americano e europeu.

As APIs específicas do Ginga que podem ser exportadas para outros sistemas são chamadas de amarelas. Entre elas estão aquelas que provêm suporte a múltiplos usuários, a múltiplos dispositivos e a múltiplas redes. Estão também aquelas que oferecem suporte às aplicações que podem ser recebidas, armazenadas e executadas em um tempo futuro.

O suporte para as necessidades específicas de aplicações voltadas para o Brasil, em especial aplicações de inclusão social, são endereçadas pela API vermelha do Ginga-J. Ginga-J tem por base um conjunto de pacotes Java, comuns a diversos middlewares imperativos, conforme será especificado na Norma ABNT NBR 15606-4.

Entre as APIs específicas, cabe ainda ressaltar aquelas para a comunicação com o ambiente declarativo Ginga-NCL. Uma aplicação Java pode agir como entidade filha de uma aplicação

declarativa, ou como uma entidade inicial controlando o ciclo de vida de uma entidade filha declarativa.

Quando a entidade Java é a entidade inicial, ela pode criar, modificar e destruir documentos declarativos NCL através das APIs de comandos de edição Ginga, conforme especificado na Norma ABNT NBR 15606-2. Quando a entidade Java é uma entidade filha, ela atua como um objeto de mídia NCL, podendo se registrar para receber eventos NCL. Eventos NCL poderão, a partir de então, acionar métodos das classes Java do objeto. Objetos de mídia NCL com códigoimperativo Java podem também comandar condições de disparos de relacionamentos NCL, usados no sincronismo temporal e espacial da apresentação de conteúdos. Podem também manipular variáveis globais de aplicações declarativas, responsáveis pela determinação da adaptação de conteúdos ou da forma como conteúdos são apresentados.

As APIs amarela e vermelha, e a as APIs da ponte são inovações brasileiras do ambiente imperativo Ginga-J, que o distingue dos demais ambientes imperativos dos middlewares do sistema europeu e americano.

Ginga-NCL é a inovação totalmente brasileira do SBTVD. O ambiente tem por base a linguagem NCL (uma aplicação XML) e sua linguagem de script Lua, ambas desenvolvidas nos laboratórios da Pontifícia Universidade Católica do Rio de Janeiro.

Os ambientes declarativos dos sistemas americano (ACAP-X), europeu (DVB-HTML) e japonês (BML-ARIB) têm por base a linguagem XHTML.

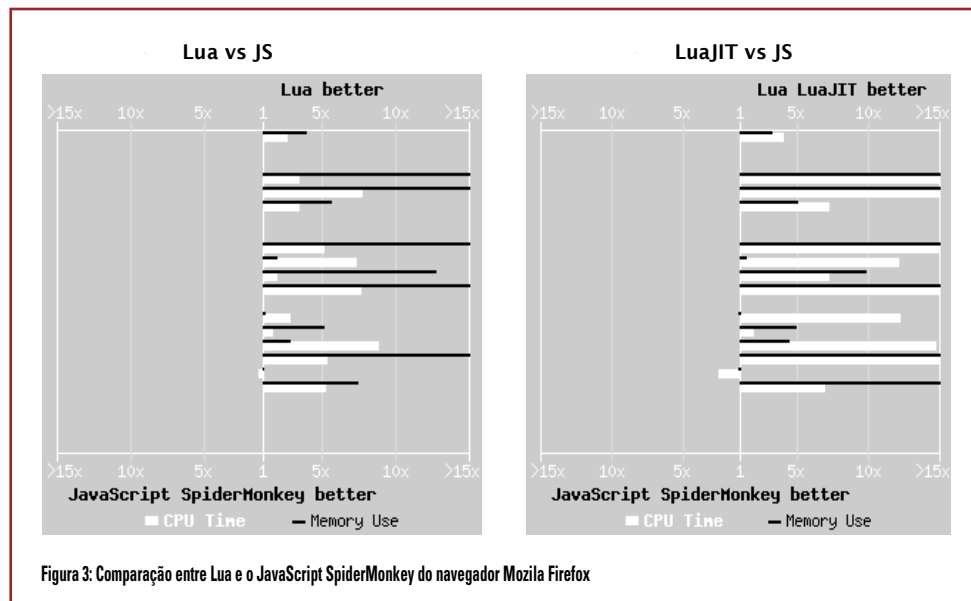


Figura 3: Comparação entre Lua e o JavaScript SpiderMonkey do navegador Mozilla Firefox

XHTML carrega o legado de tecnologias anteriormente desenvolvidas para navegação textual. Em sentido contrário, aplicações para TVD são usualmente centradas no vídeo. Além disso, o modelo da linguagem XHTML tem o foco no suporte à interação do usuário telespectador. Outros tipos de relacionamentos, como relacionamentos de sincronização espaço-temporal e relacionamentos para definição de alternativas (adaptação de conteúdo e de apresentação), são usualmente definidos através de uma linguagem imperativa, no caso de todos os três sistemas citados a linguagem ECMAScript.

Diferente das linguagens baseadas em XHTML, NCL define uma separação bem demarcada entre o conteúdo e a estrutura de uma aplicação, provendo um controle não invasivo da ligação entre o conteúdo e sua apresentação e leiaute. O modelo da linguagem NCL visa um domínio de aplicações mais amplo do que o oferecido pela linguagem XHTML. NCL visa não apenas o suporte declarativo à interação do usuário, mas o sincronismo espacial e temporal em sua forma mais ampla, tratando a interação do usuário como um caso particular. NCL visa também o suporte declarativo a adaptações de conteúdo e de formas de apresentação de conteúdo, o suporte declarativo a múltiplos dispositivos de exibição e a edição/produção da aplicação em tempo de exibição, ou seja, ao vivo. Esses são também os focos da maioria das aplica-

ções para TV digital, o que torna NCL a opção preferencial no desenvolvimento da maioria das aplicações de TVD. Para os poucos casos particulares, como por exemplo, quando a geração dinâmica de conteúdo é necessária, NCL provê o suporte de sua linguagem de script Lua. Alternativamente, as APIs da ponte com o Ginga-J podem ser usadas acionando o suporte imperativo oferecido pela linguagem Java.

Como a NCL tem uma separação mais acurada entre o conteúdo e a estrutura de uma aplicação, ela não define nenhuma mídia per si. Ao contrário, ela define a cola que prende as mídias em apresentações multimídia. NCL apenas define como objetos de mídia são estruturados e relacionados, no tempo e espaço. Como uma linguagem de cola, ela não restringe ou prescreve os tipos de conteúdo dos objetos de mídia. Nesse sentido, podemos ter objetos de imagem, de vídeo, de áudio, de texto, de código imperativo (Xlet e Lua, no SBTVD), entre outros, como objetos de mídia NCL. Quais objetos de mídia têm suporte depende dos exibidores de mídia que estão acoplados ao formatador NCL (na verdade, que têm suporte no Ginga-CC, vide Figura 2). No SBTVD, um desses exibidores é o decodificador/exibidor MPEG-4, implementado em hardware no receptor de televisão digital. Dessa forma, o vídeo e o áudio MPEG-4 são tratados como todos os demais objetos de mídia que podem ser relacionados utilizando NCL, em outras

palavras eles são simplesmente parte de uma aplicação de TVD.

Outro objeto de mídia NCL que deve obrigatoriamente ser suportado pelo Ginga-NCL é o objeto de mídia baseado em XHTML. A NCL não substitui, mas embute documentos (ou objetos) baseados em XHTML. Como acontece com outros objetos de mídia, qual linguagem baseada em XHTML tem suporte em um formatador NCL é uma escolha de implementação e, portanto, depende de qual navegador XHTML, incorporado no formatador NCL (na verdade suportado pelo Ginga-CC), atua como exibidor dessa mídia.

Como consequência, é possível ter navegadores BML, DVB-HTML e ACAP-X individualmente embutidos em um exibidor de documento NCL. É possível, ainda, ter todos eles. Assim, aplicações declarativas desenvolvidas para aqueles sistemas também executariam com o suporte oferecido pelo middleware Ginga. Resta mencionar que as Normas ABNT NBR 15606-2 e ABNT NBR 15606-5 definem apenas um conjunto de funcionalidades básicas para XHTML e suas tecnologias derivadas como obrigatórias. A escolha de outras funcionalidades adicionais é opcional.

Voltando nossa atenção para a Figura 2, o componente Formatador NCL já foi por demais citado, e tem como responsabilidade orquestrar toda a execução de uma aplicação NCL, garantindo que os relacionamentos espaço-temporais definidos pelo autor da aplicação sejam respeitados. A máquina de execução Lua é responsável pelo processamento do código imperativo Lua. Lua é uma linguagem de programação imperativa eficiente, rápida e leve, projetada para estender aplicações. Lua combina uma sintaxe simples para programação imperativa com construções poderosa para descrição de dados baseadas em tabelas associativas e em semântica extensível. Lua é tipada dinamicamente, é interpretada e tem gerenciamento automático de memória, com coleta de lixo incremental. Essas características fazem de Lua uma linguagem ideal para configuração, automação (scripting) e prototipagem rápida (geração rápida de aplicações). Lua é uma das linguagens de script mais

eficientes; muito mais rápida do que ECMAScript, e com um footprint de memória bem menor, como indica a Figura 3, obtida do site de avaliação de linguagens <http://shootout.alioth.debian.org/>: Lua é, em média 7 vezes mais rápida e com um uso de memória 40 vezes menor. Lua é hoje a linguagem mais importante na área de entretenimento.

O Formador NCL trata de aplicações recebidas pelo Ginga-CC e depositadas em uma estrutura de dados chamada “base privada”. Existe uma base privada por canal de radio frequência. Cabe ao componente Gerenciador de Bases Privadas a tarefa de receber comandos para ativação e manipulação dessas aplicações. Como anteriormente mencionado, no Ginga-NCL, uma aplicação de TVD pode ser gerada ou modificada ao vivo (em tempo real), através de comandos de edição. O conjunto de comandos de edição, especificados na Norma ABNT NBR 15606-2, pode ser dividido em três grupos.

O primeiro grupo de comandos é responsável pela ativação e desativação de

uma base privada, ou seja, a habilitação de aplicações de um determinado canal de TV. Em uma base privada, aplicações NCL podem ser ativadas, pausadas, retomadas e desativadas, através de comandos bem definidos pertencentes ao segundo grupo de comandos. O terceiro grupo define comandos para modificações de uma aplicação ao vivo.

Finalmente, como o Ginga-J, Ginga-NCL oferece suporte a múltiplos dispositivos de entrada e saída. Tal facilidade declarativa, juntamente com os comandos de edição ao vivo, únicos do sistema brasileiro, provê suporte para o grande domínio de aplicações interativas de TVD que se descortina: as aplicações para as chamadas TV em comunidade (Community ou Social TV), onde uma comunidade de usuários cria ao vivo, sobre o conteúdo e aplicações recebidas, novas aplicações (geração de novos conteúdos e informações personalizadas), que são trocadas entre seus membros, para exibição em tempo real ou sob demanda.

Conclusões

O Sistema Brasileiro é, atualmente, o mais avançado sistema de TV digital terrestre, não apenas por usar as tecnologias mais avançadas, mas, principalmente, por dispor de tecnologias inovadoras, como é o caso de seu middleware Ginga.

A implementação de referência do Ginga-NCL foi desenvolvida em código aberto e pode ser obtida em www.ncl.org.br, sob de licença GPLv2. A máquina Lua também se encontra disponível pelo mesmo site, através de licença MIT. Ferramentas de autoria para o desenvolvimento de aplicações NCL, também em código aberto, estão disponíveis ainda no mesmo site, bem como tutoriais, livros e exemplos de várias aplicações NCL e Lua. Várias listas de discussão e contribuições no desenvolvimento de aplicações, podem ser encontradas na comunidade Ginga, em www.softwarepublico.gov.br. Documentos, artigos e tutoriais a respeito do middleware Ginga, podem ser obtidos em www.ginga.org.br. **PP**