

Produção de Conteúdo Declarativo para TV Digital

Rogério Ferreira Rodrigues

rogerio@telemidia.puc-rio.br

Luiz Fernando Gomes Soares

lfgs@inf.puc-rio.br

Depto. de Informática - PUC-Rio

Rua Marquês de São Vicente, 225, Rio de Janeiro, RJ - 22453-900.

***Resumo.** O direito à geração e disseminação de informação (conteúdo digital), permitindo ao cidadão que sua produção cultural seja registrada e divulgada, conseqüentemente diminuindo as desigualdades regionais e sociais, pode estar mais perto com o advento da TV digital. Este artigo discute a produção de programas não-lineares, para o sistema de TV digital brasileiro, utilizando a linguagem NCL. Ao mesmo tempo que os conceitos da linguagem são introduzidos, uma metodologia para concepção dos programas é apresentada.*

***Abstract.** The right to generate and disseminate information (digital content), allowing citizens to register and publish their cultural production, consequently reducing the regional and social disparities, can be closer with the digital TV introduction. This paper discusses the composition of non-linear programs, for the Brazilian digital TV system, using the NCL language. The paper explains the main concepts of the language and, at the same time, presents a methodology for NCL program authoring.*

1. Introdução

Muito esforço tem sido despendido no sentido da universalização do acesso às informações. Inúmeros projetos de governo têm tratado do assunto, tanto no âmbito de um equilíbrio regional, quanto no da inserção de cidadãos de baixa renda. No entanto, esse é apenas um viés da questão. Em outra direção, é importante garantir a todos o direito de geração e disseminação de informação (conteúdo digital), permitindo ao cidadão que sua produção cultural seja registrada e divulgada, tanto localmente quanto para o público em geral.

É sabido que grande parte da cultura nacional é gerada em comunidades de baixa renda e em regiões menos privilegiadas economicamente. A apropriação deste conteúdo por terceiros não é justa, não apenas do ponto de vista econômico, mas também por perpetuar uma desigualdade, pois dá a esses terceiros o direito da escolha do que será difundido como “cultura nacional”.

Se antes a apropriação da cultura e a geração de conteúdo de alta qualidade exigia um grande capital, hoje isso não é mais verdade. A evolução das tecnologias da informação deu início a uma mudança neste cenário, trazendo novas formas “mais democráticas” de produção de conteúdo. No entanto, a realização de ações de estímulo, valorização e difusão da cultura gestada, vivida e produzida em regiões marginalizadas brasileiras certamente exige políticas governamentais conseqüentes, que contemplem também o domínio tecnológico de ferramentas que as propiciem. Um exemplo de tais políticas começa a ser esboçado nos primeiros passos rumo à TV digital brasileira.

Sem dúvida a televisão é hoje, para nós brasileiros, a forma mais abrangente de distribuição de conteúdo, atingindo mais de 98% da população [Mina05]. A TV digital amplia mais ainda este espectro, uma vez que não apenas o áudio e vídeo principal de um programa televisivo poderão ser difundidos, como também outros dados. É assim imperativo que o Brasil tenha o domínio das tecnologias da informação nas inúmeras atividades que envolvem o acesso e a capacidade de gerar e apresentar informações televisivas, e que a absorção dessas tecnologias sirva para diminuir o desequilíbrio regional e social do país.

A TV digital possui várias características que possibilitam uma maior democratização na geração e distribuição de conteúdo. Começa pelo potencial aumento no número de canais em um mesmo espectro de frequência, o que permite a presença de novos atores, tais como TVs universitárias, TVs comunitárias etc. Passa pela alternativa de conteúdo, que permite que um mesmo programa possa ter seu conteúdo modificado, dependendo do usuário, de seu aparelho receptor ou da região onde é exibido. Entre inúmeras outras possibilidades, termina pelo oferecimento de “aplicações cidadãs”, tais como governo eletrônico e aplicações na área de educação e saúde.

Programas de TV digital interativa podem ser entendidos como aplicações hipermídia/multimídia. Nesse cenário, sistemas hipermídia (ou multimídia interativos) irão se constituir em uma das ferramentas mais importantes a serem dominadas. Sistemas de autoria hipermídia são o suporte para a geração de informação, não se restringindo apenas à concepção dos conteúdos em si, mas incluindo também a concepção de como eles devem ser apresentados. Sistemas de exibição hipermídia (núcleo central dos chamados *middlewares* para TV) são os responsáveis pela apresentação especificada. Todos esses sistemas têm por base alguma linguagem de especificação.

Conteúdos para TV digital interativa são usualmente concebidos usando uma linguagem declarativa (aplicações essas que para serem exibidas têm o suporte do chamado *middleware* declarativo), ou uma linguagem imperativa (a linguagem Java predomina e, nesse caso, as aplicações têm o suporte do chamado *middleware* procedural¹). Além do suporte à criação de conteúdos, o *middleware* tem a função de “virtualizar” os aparelhos de televisão dos diferentes fabricantes, definindo para os que produzem conteúdo uma visão única de plataforma. Esse papel confere ao *middleware* fundamental importância, pois é ele quem regula as relações entre duas indústrias estratégicas para o país: a de produção de conteúdo e a de fabricação de aparelhos receptores.

Quando projetadas com um foco específico a resolver, as linguagens declarativas são muito mais fáceis de serem utilizadas no desenvolvimento de aplicações que têm esse mesmo foco. Como discutido na próxima seção, esse é o caso dos chamados programas não-lineares, que se constituem na maioria dos conteúdos gerados por não especialistas.

A linguagem declarativa NCL [SoRo05] foi proposta como a linguagem do *middleware* declarativo Ginga-NCL para o Sistema Brasileiro de TV Digital (SBTVD). Este artigo procura realçar algumas das funcionalidades dessa linguagem, ao mesmo

¹ Embora Java não seja considerada como seguindo o paradigma procedural, e sim orientado a objetos, ficou consagrado o termo procedural para a máquina que dá suporte a aplicações Java.

tempo em que apresenta uma metodologia para o desenvolvimento de um programa não-linear usando NCL. Mesmo sem o uso do editor gráfico NCL [CoRS04], pretende-se mostrar a facilidade do desenvolvimento de programas por não especialistas. Seguindo esta metodologia, um curso de 8 horas foi dado para alunos do ensino médio em uma comunidade de baixa renda e, ao final do curso, já se tinha a produção de um vídeo clipe com uma banda da comunidade.

Este artigo encontra-se organizado da seguinte forma. Na Seção 2 são apresentados os vários tipos de aplicações encontradas no ambiente de TV digital interativa. A Seção 3 introduz alguns conceitos básicos sobre linguagens declarativas com o foco em sincronismo de mídia. A Seção 4 apresenta um exemplo de programa não-linear desenvolvido, passo a passo, em NCL. Nessa seção, os diversos conceitos e funcionalidades da linguagem NCL são discutidos. A geração de programas não-lineares ao vivo é o assunto da Seção 5. Na Seção 6, tecnologias relacionadas, presentes em outros sistemas de TV digital, são apresentadas e comparadas com a opção pelo *middleware* declarativo proposto para o SBTVD. A Seção 7 é reservada às conclusões.

2. Aplicativos para TV Digital

Esta seção discute a necessidade de *middlewares* declarativos e procedurais; caracteriza as várias aplicações encontradas no ambiente de TV digital interativa; e discute o foco das linguagens declarativas propostas para esse ambiente.

2.1. Middlewares Declarativos e Procedurais

Numa programação procedural deve-se informar ao computador cada passo a ser executado. Pode-se afirmar que, em linguagens procedurais, o programador possui um maior controle do código, sendo capaz de estabelecer todo o fluxo de controle e execução de seu programa. Entretanto, para isso, ele deve ser bem qualificado e conhecer bem os recursos de implementação.

Nas linguagens declarativas, o programador fornece apenas o conjunto das tarefas a serem realizadas, não estando preocupado com os detalhes de como o executor da linguagem (interpretador, compilador ou a própria máquina real ou virtual de execução) realmente implementará essas tarefas. Em outras palavras, a linguagem enfatiza a declaração descritiva de um problema ao invés de sua decomposição em implementações algorítmicas, não necessitando, em geral, de tantas linhas de código para definir uma certa tarefa.

Middlewares declarativos são mais adequados para aplicações cujo foco casa com o objetivo específico para o qual a linguagem foi desenvolvida, em caso contrário, o uso de *middlewares* procedurais é mais apropriado. Como na TV Digital os dois tipos de aplicações irão coexistir, é conveniente que o dispositivo receptor integre os dois tipos de *middleware*.

2.2. Sincronismo e Interatividade

A necessidade de sincronização em um Sistema de TV Digital está presente mesmo em sua aplicação mais primária: a exibição temporalmente sincronizada do fluxo de vídeo e áudio principal de um programa de TV. Aplicações para TV digital muitas vezes devem lidar com a sincronização, espacial e temporal, de objetos de diferentes tipos de mídia, além dos objetos de vídeo e áudio que compõem o fluxo principal. Tome como exemplo

uma aplicação que, no momento preciso no qual o ator de um filme em exibição tira seus óculos, um vídeo-propaganda de uma ótica é exibido. Note que, nesse caso, o sincronismo foi baseado em um evento previsível do tempo (por exemplo, a retirada dos óculos ocorre depois de transcorridos 10,5 minutos do início do filme).

O sincronismo presente em documentos multimídia pode não ser apenas baseado em ocorrências (eventos) previsíveis, ou seja, em eventos onde é possível se prever o tempo e o local de suas ocorrências, relativos a uma outra ocorrência (outro evento) qualquer. A interatividade do usuário é um exemplo de evento imprevisível. Note, assim, que a interatividade é apenas um caso particular, embora importante, de sincronismo de mídias.

O foco dos *middlewares* declarativos presentes nos três principais sistemas de TV digital (europeu [ETSI05], americano [ATSC05] e japonês [ARIB04]) é a interatividade, isto é, eles têm como objetivo facilitar o desenvolvimento das aplicações com interação do usuário telespectador. Por isso, tais *middlewares* adotaram, como linguagem, derivações da linguagem HTML. Esse foco é, no entanto, bastante restritivo. Para contornar suas restrições, as derivações de HTML, nos sistemas de TV, permitem a inclusão de objetos especificados na linguagem ECMAScript [ECMA99]. Qualquer outro tipo de sincronismo de mídias mais complexo terá de ser descrito através de scripts, ou seja, de forma mais complexa e mais longe dos objetivos do projetista da aplicação.

2.3. Caracterização das Aplicações

Nem todas as aplicações para TV digital têm uma relação semântica com o conteúdo do áudio e vídeo principal do programa, ou seja, com o significado da informação sendo exibida a partir do áudio e vídeo principal. Um exemplo típico é uma aplicação de correio eletrônico, que pode ser requisitada a qualquer instante e não tem a menor relação com o que está sendo exibido pela TV.

Outras aplicações têm relação semântica com o conteúdo do programa sendo exibido, mas não têm nenhuma relação de sincronismo com o áudio ou o vídeo principal. Tome como exemplo a aplicação de saúde “realização de um teste de estresse”, que pode ser acionada a qualquer instante que se queira dentro de um programa televisivo sobre “doenças do coração”.

É importante frisar que, embora os tipos de aplicações descritos nos dois parágrafos anteriores não tenham nenhuma relação de sincronismo com o áudio e vídeo principal, eles podem ser compostos por objetos de mídia cujas apresentações devam ser sincronizadas no tempo ou no espaço.

Ainda um terceiro tipo de aplicação é composto por aplicações em que existe não só uma relação semântica entre seus objetos de mídia e o áudio e vídeo principal do programa televisivo, mas também uma relação de sincronismo. Esse é exatamente o caso dos chamados *programas não-lineares*. O termo vem em contraposição à forma sequencial – linear – que caracteriza os programas para a TV analógica. Nesses últimos, existe um e apenas um caminho, sequencial, de exibição. Ao contrário, os programas não-lineares são compostos de múltiplas cadeias de exibição, algumas exibidas em paralelo e outras como alternativas (ou seja, ou uma cadeia ou a outra) que dependem da escolha do usuário, do terminal onde o programa será exibido, da região onde o telespectador está inserido etc. O exemplo mais simples de um programa não-linear é

aquele onde, em um dado instante de exibição, o usuário telespectador pode escolher entre formas alternativas de sua continuação. Note assim que o programa deixa de poder ser representado por uma linha de tempo e passa a ter um fluxo de exibição que pode ser representado por um grafo.

Em geral, aplicações onde o sincronismo exerce papel preponderante são mais fáceis de serem especificadas usando uma linguagem declarativa orientada a esse tipo de foco. Porém, quando a exigência de sincronismo é apenas eventual, o melhor suporte é dado por uma linguagem procedural de propósito geral. Na grande maioria dos casos, a linguagem declarativa tende a ser a preferencial no desenvolvimento dos programas não-lineares. Mais ainda, como em programas não-lineares o sincronismo intermídia sem a interação do usuário deve ser tão ou mais importante que a interatividade, o sincronismo de mídias em sua forma mais ampla, e não a interatividade, deve ser o foco das linguagens declarativas, como é o caso da linguagem NCL, proposta para o SBTVD, e outras linguagens em estudo para inclusão (ou substituição) nos outros sistemas de TV digital já mencionados.

3. Conceitos Básicos de Linguagens Declarativas para Produção de Programas Não-Lineares

No âmbito de documentos hipermídia, sincronismo é definido como o relacionamento, no tempo ou espaço, entre eventos. Evento é uma ocorrência no tempo, de duração finita ou infinitesimal, como, por exemplo, a apresentação de um trecho de um objeto de mídia ou de todo o seu conteúdo (evento de apresentação); a seleção de um trecho de um conteúdo de mídia (evento de seleção) através de qualquer dispositivo seletor (mouse, controle remoto etc.); a atribuição de um valor a uma propriedade de um objeto de mídia (evento de atribuição); etc.

O relacionamento entre eventos pode ser realizado de forma absoluta ou relativa. A especificação do relacionamento, de forma absoluta, através do posicionamento dos eventos com o uso de eixos ou selos de tempos ou espaço não contempla o sincronismo não determinístico (com algum evento não determinístico, como os de seleção) e nem a apresentação de eventos alternativos, e por isso mesmo não é usado em nenhuma linguagem declarativa para TV digital. A definição relativa dos relacionamentos pode ser obtida através de várias técnicas.

Uma das técnicas representa os relacionamentos através de composições com semântica temporal ou espacial embutida. Como exemplo, tem-se as composições temporais *seq* e *par* em SMIL [W3C05] e XMT-O [ISO01], com a semântica que todos os objetos de mídia que elas contêm devam ser exibidos em seqüência ou em paralelo, respectivamente. A técnica facilita bastante o trabalho do autor do programa não-linear. Por outro lado, nesse tipo de abordagem, relacionamentos mais complexos têm que ser descritos por uma hierarquia de composições usando os tipos básicos. Isso nem sempre é desejável, pois obriga que a estrutura lógica do documento seja igual à sua estrutura de apresentação temporal dada pelas composições. Essa estruturação pode se tornar ainda mais difícil de ser concebida quando existe um objeto mestre (o fluxo de áudio e vídeo principal) no qual se baseiam a grande maioria dos relacionamentos de sincronização, como é o caso usual de programas não-lineares.

A especificação de relacionamentos de sincronização espaço-temporais através de elos, que é a abordagem adotada por NCL, permite a definição de relacionamentos

causais e de restrição. Relacionamentos causais estabelecem condições (origem do elo), sobre o estado de eventos ou valores de propriedade de objetos de mídia, que quando satisfeitas disparam ações de mudanças de estado em outros eventos ou ações de atribuição de valores a propriedades (destino dos elos). Relacionamentos de restrição, sem nenhuma causalidade envolvida, também podem ser especificados por elos. Considere, por exemplo, a restrição especificando que um objeto de mídia deve terminar sua apresentação ao mesmo tempo que outro começa a dele. A ocorrência de uma apresentação sem a ocorrência da outra também satisfaz a restrição, que especifica que, se e somente se esses dois objetos forem apresentados, seus tempos de fim e início, respectivamente, devem coincidir.

A especificação de relacionamentos através de elos permite que o uso de composições possa ser dedicado à criação de relações de estruturação entre componentes de um programa não-linear. Além disso, o uso de elos para definir o sincronismo espaço-temporal permite que um autor defina qualquer tipo de relação de sincronização. Essa abordagem oferece mais flexibilidade, já que o autor não precisa ficar limitado a alguns tipos de composição pré-definidos pela linguagem e pode estruturar seu documento de forma independente de suas relações de sincronização. Entretanto, para relacionamentos simples, pode ser mais trabalhosa a especificação utilizando elos.

Dadas as vantagens e desvantagens do uso de composições e elos para representar relacionamentos, o ideal é oferecer ao autor todas as possibilidades: o uso de composições para estruturação lógica, o uso de elos para especificar sincronização e o uso de composições com semântica, por exemplo temporal, fazendo a estrutura lógica casar com a estrutura de apresentação, quando apropriado. Melhor ainda se a linguagem de autoria permitir a criação de tipos de composição com a semântica desejada pelo autor e que possam ser utilizados (e reutilizados) de acordo com sua necessidade, e não apenas limitar-se a um conjunto fixo de tipos. A linguagem NCL é a única linguagem que oferece ao autor essas várias possibilidades, através do conceito de *template* de composição. Dessa forma, tipos pré-definidos de composições, como os oferecidos pela linguagem SMIL, podem ser especificados como *templates* NCL e, mais ainda, novos *templates* de composição podem ser criados, generalizando os tipos de composição que a linguagem de autoria oferece.

Em NCL a definição de eventos (apresentação ou seleção de trechos de conteúdos marcados etc.), bem como o relacionamento entre eventos (elos) são realizados antes da exibição do programa não-linear. Entretanto, muitos programas são gerados ao vivo. Para dar suporte à concepção de programas não-lineares ao vivo, o *middleware* declarativo Ginga-NCL permite a definição de eventos e elos em tempo de exibição (através de eventos de sincronismo DSM-CC [ISO98]). Essa característica é mais uma vantagem que a especificação de relacionamentos por elos tem sobre a especificação por composições, onde é extremamente difícil a definição de sincronismo em objetos de mídia gerados ao vivo. A Seção 5 discute com um pouco mais de detalhes a edição ao vivo de um programa não-linear.

4. Geração de Programas em NCL

NCL é uma linguagem declarativa baseada em XML para autoria de documentos hipermídia. Seguindo a abordagem de outras linguagens declarativas [ISO01, W3C05], NCL foi concebida de forma modular. Módulos agrupam, de modo coerente, elementos

e atributos XML que possuam alguma relação semântica entre si. Essa estruturação permite que as funcionalidades da linguagem sejam reunidas de acordo com as necessidades de uma determinada aplicação. Sendo assim, um subconjunto das funcionalidades de NCL foi reunido para compor um perfil apropriado ao desenvolvimento de programas de TV não-lineares [SoRo05].

Com o intuito de ao mesmo tempo oferecer uma visão geral das funcionalidades da linguagem NCL e delinear uma metodologia para seu uso, esta seção foi organizada de tal forma a construir um exemplo de programa não-linear passo a passo.

4.1. Apresentação do exemplo e sua divisão lógica (Roteirização)

O programa de TV interativo escolhido como exemplo é constituído de um vídeo principal contendo uma cena do filme Matrix, onde ocorre um diálogo entre os personagens Morpheus e Neo. Em um determinado momento do diálogo, Morpheus suspende e mostra a Neo uma pilha. Nesse exato instante, um outro vídeo contendo a propaganda de uma marca de bateria deverá ser exibido em uma janela menor, sobre a exibição do vídeo principal.

Em um outro ponto do diálogo, deseja-se fazer uma propaganda interativa dos óculos escuros utilizados por Morpheus. Durante um trecho da fala desse personagem, um ícone de óculos, de cor amarela, deverá ser exibido sobre os seus óculos. Se o usuário selecionar o botão amarelo do controle remoto enquanto o ícone estiver sendo exibido, o ícone deverá sumir, o vídeo principal do filme deverá ser redimensionado para $\frac{1}{4}$ da tela e pausado, ao mesmo tempo, um vídeo com a propaganda de uma ótica será exibido em uma janela menor, ocupando outro $\frac{1}{4}$ da tela. Em seguida à exibição da propaganda, um formulário HTML deverá ser exibido para que, se desejar, o usuário proceda com a compra. Ao fechar o formulário, o diálogo do filme principal deverá ser redimensionado para a tela cheia e a exibição do filme retomada.

4.2. Estrutura básica de um documento NCL

A estrutura básica de um documento NCL é formada por um elemento *ncl* que contém um cabeçalho (elemento *head*) e um corpo (elemento *body*).

Conforme ilustrado na Figura 1, a declaração de um elemento *ncl* deve ser acompanhada de um identificador opcional do documento (atributo *id*) e da especificação do *namespace* para declaração dos elementos da linguagem NCL [W3C99]. O leitor interessado poderá encontrar no endereço <http://www.telemidia.puc-rio.br/specs/xml/NCL23/> os arquivos contendo a especificação completa e livre de ambigüidades da linguagem NCL fazendo uso de XML Schema [W3C04b]. Qualquer documento NCL começa com as seis primeiras linhas indicadas na figura.

No corpo de um documento NCL são definidos os objetos de mídia (vídeos, imagens, textos etc.) que formam o conteúdo da apresentação, assim como os relacionamentos de sincronismo entre os objetos de mídia, conforme será explicado nas Seções 4.2 e 4.4, respectivamente. No cabeçalho de um documento NCL encontram-se as informações referentes à forma que a apresentação assumirá, como o leiaute espacial dos objetos e as definições dependentes da plataforma de exibição, conforme será visto na Seção 4.3.

```
<?xml version="1.0"?>
```

```

<ncl id="matrixExample"
xmlns="http://www.telemidia.puc-rio.br/specs/xml/NCL23/profiles"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.telemidia.puc-rio.br/specs/xml/NCL23/profiles
http://www.telemidia.puc-rio.br/specs/xml/NCL23/profiles/NCL23.xsd">
<head>
...
</head>
<body>
...
</body>
</ncl>

```

Figura 1 – Estrutura básica de um documento NCL.

4.3. Nós de Mídia, Nós de Contextos e Interfaces dos Nós

O elemento básico de informação em um programa não-linear de TV é chamado de um *nó de mídia* (ou *objeto de mídia*). Objetos de mídia podem ser de diversos tipos, como vídeo, áudio, texto, imagem estática, fragmentos de código executável etc., e são declarados na linguagem NCL através do elemento *media*. Um objeto de mídia NCL deve especificar o seu *tipo* e a localização de seu conteúdo propriamente dito (a URI do conteúdo, por exemplo). Além disso, objetos de mídia definem, opcionalmente, fragmentos de informação (*âncoras*) do seu conteúdo que podem vir a ser usados em relacionamentos com outros objetos. Por exemplo, no nosso caso de estudo, o trecho do Matrix em que o Morpheus levanta e mostra uma pilha deve se relacionar com o objeto contendo a propaganda da bateria. Em NCL, âncoras são descritas através do elemento *area*. O detalhamento de uma âncora depende do tipo de mídia em questão; por exemplo, para um objeto de mídia do tipo vídeo, uma âncora pode definir um intervalo de tempo na exibição do objeto. As âncoras darão origem a alguns tipos de eventos, mencionados na Seção 3.

A Figura 2 ilustra a especificação dos objetos de mídia que compõem o exemplo descrito na Seção 4.1 (*matrix*, *battery*, *glassesIcon*, *glasses* e *ptForm*). Como pode ser observado, duas âncoras temporais (*anchorBattery* e *anchorGlasses*) são também definidas para o objeto de mídia *matrix*, que modela o vídeo principal.

Além das âncoras, um nó de mídia pode exportar atributos para que participem também na especificação dos relacionamentos. No exemplo, o vídeo principal define um atributo (*bounds*) que é uma abstração da área de exibição do vídeo. Como será visto na Seção 4.4, esse atributo será útil na definição dos relacionamentos que fazem o redimensionamento do vídeo principal.

Uma das mais importantes contribuições da linguagem NCL é a definição de um tipo especial de nó, chamado *nó de contexto* (elemento *context*), que permite definir uma estruturação para os documentos (as composições comentadas na Seção 3). Um contexto contém nós e relacionamentos entre os nós. Os nós de um contexto, por sua vez, podem ser nós de mídia ou outros nós de contexto.

No exemplo, foi definido um contexto para agrupar os objetos de mídia (vídeo da propaganda da ótica e formulário para compra) referentes ao comercial dos óculos (*glassesContext*). Cabe observar que o próprio corpo (elemento *body*) de um documento NCL é um contexto representando a estruturação do programa não-linear.

```

<ncl ...>
<head>
  ...
</head>
<body>
  <port id="docIn" component="matrix" />
  <media id="matrix" type="video" descriptor="matrixDesc" src="matrix.mpg">
    <area id="anchorBattery" begin="18s" />
    <area id="anchorGlasses" begin="27s" end="32s" />
    <property id="bounds" name="bounds" />
  </media>
  <media id="battery" type="video" descriptor="pubDesc" src="battery.mpg" />
  <media id="glassesIcon" type="image" descriptor="glassesIconDesc"
src="glassesIcon.jpg" />
  <context id="glassesContext">
    <port id="glassesAdIn" component="glasses" />
    <port id="glassesAdOut" component="ptForm" />
    <media id="glasses" type="video" descriptor="pubDesc" src="glasses.mpg" />
    <media id="ptForm" type="text" descriptor="formDesc" src="ptForm.html" />
    ...
  </context>
  ...
</body>
</ncl>

```

Figura 2 – Objetos de mídia, âncoras, atributos, contextos e portas em NCL.

Da mesma forma que os objetos de mídia, contextos podem possuir interfaces, que exportam fragmentos de informação de um contexto para a criação de relacionamentos. Além de âncoras e atributos, contextos possuem um tipo especial de interface que é chamado de porta. Uma porta (elemento *port*) permite criar um mapeamento entre o contexto e a interface de um componente interno ao contexto. As portas permitem restringir os pontos de entrada e saída dos contextos, o que é importante para garantir a propriedade de composicionalidade da linguagem NCL. Dessa forma, nós e interfaces de nós que não estejam mapeados através de portas não são visíveis para nós que estejam externos ao contexto. Como consequência, relacionamentos de sincronismo não podem ser estabelecidos com esses elementos. No exemplo, uma porta no corpo do documento exporta o ponto de entrada do programa (*docIn*), enquanto duas outras portas no contexto da propaganda da ótica (*glassesAdIn* e *glassesAdOut*) exportam os elementos que integram o anúncio da ótica.

4.4. Descritores e Leiaute Espacial

A linguagem NCL separa o conteúdo e a estruturação de um programa das características de apresentação, buscando favorecer reuso e manutenção. Sendo assim, descritores (elementos *descriptor*) devem ser especificados para estabelecer a associação entre os nós e as suas características de exibição. Uma importante informação de exibição é a disposição espacial dos objetos (leiaute da apresentação). Em NCL, essa especificação é feita através de uma base de regiões (elemento *regionBase*). Cada região (elemento *region*) especifica uma área espacial onde um objeto pode ser exibido. A Figura 3 ilustra a especificação dos descritores e regiões, feita no cabeçalho do documento NCL, para o exemplo do filme Matrix.

```

<ncl ...>
<head>
  <regionBase>
    <region id="screenRegion" width="800" height="600">
      <region id="mainRegion" width="100%" height="100%" zIndex="1" />
      <region id="pubRegion" top="50%" left="50%" width="50%" height="50%" />
      <region id="glassesRegion" top="27%" left="15%" width="10%" height="8%" />
      <region id="formRegion" left="50%" width="50%" height="100%" />
    </region>
  </regionBase>
  <descriptorBase>
    <descriptor id="matrixDesc" region="mainRegion">
      <descriptorParam name="soundLevel" value="1" />
    </descriptor>
    <descriptor id="pubDesc" region="pubRegion">
      <descriptorParam name="soundLevel" value="0.8" />
    </descriptor>
    <descriptor id="glassesIconDesc" region="glassesRegion" />
    <descriptor id="formDesc" region="formRegion" explicitDur="120s" >
      <descriptorParam name="scroll" value="never" />
    </descriptor>
  </descriptorBase>
  ...
</head>
<body>
  ...
</body>
</ncl>

```

Figura 3 – Definição do sincronismo espacial.

Como pode ser observado na Figura 2, os objetos de mídia possuem um atributo que especifica o descritor a ser utilizado. Mais de um objeto, como acontece com as propagandas (*battery* e *glasses*), podem usar o mesmo descritor.

Além de definir a região de apresentação, o descritor também pode definir outras características de uma apresentação, independentes do conteúdo sendo apresentado. No exemplo, o descritor do filme (*matrixDesc*) estabelece que o volume do som deve estar igual ao volume escolhido pelo telespectador, enquanto as propagandas (*pubDesc*) deverão ser exibidas com 80% do volume. O tempo máximo que o formulário ficará na tela (2 minutos) e a opção de barra de rolagem para a página HTML também encontram-se especificados no descritor *formDesc*. Os parâmetros que se aplicam a qualquer descritor são definidos como atributos do elemento *descriptor* (região espacial, duração etc.). Já os parâmetros que são dependentes do tipo de mídia e da ferramenta que será usada na exibição são especificados através do elemento *descriptorParam*, e podem ser facilmente estendidos sem que a linguagem NCL precise ser alterada.

Regiões podem ser especificadas hierarquicamente e de modo proporcional, o que favorece uma rápida reconfiguração da disposição espacial da apresentação. No exemplo, para adaptar o programa para HDTV, basta modificar a resolução principal (atributos *width* e *height* de *screenRegion*).

4.5. Elos e Conectores

Uma vez definidos o conteúdo da apresentação, a sua estruturação e a disposição espacial inicial dos objetos, resta especificar os relacionamentos de sincronização. Como mencionado na Seção 3, em NCL esses relacionamentos são especificados através de elos. Um elo (elemento *link*) deve ser definido dentro de (pertencendo a) um contexto (podendo esse contexto ser o elemento *body*) e relacionar interfaces (âncoras, atributos ou portas) de objetos diretamente contidos no contexto. Evidentemente, ao fazer uso dos mapeamentos das portas, podem ser estabelecidos relacionamentos entre objetos de mídia que estejam contidos em contextos distintos. A Figura 4 destaca dois relacionamentos definidos diretamente no corpo do documento representante do nosso caso de estudo.

```
<ncl ...>
<head>
  ...
  <connectorBase>
    <importBase alias="connBase" baseURI="matrix.conn" />
  </connectorBase>
</head>
<body>
  ...
  <link id="link01" xconnector="connBase#onBeginStart">
    <bind component="matrix" interface="anchorBattery" role="onBegin" />
    <bind component="battery" role="start" />
  </link>
  ...
  <link id="link04" xconnector="connBase#onSelectionSetVarPauseStopStart">
    <bind component="glassesIcon" role="onKeySelection">
      <bindParam name="keyCode" value="YELLOW" />
    </bind>
    <bind component="matrix" interface="bounds" role="setBounds">
      <bindParam name="bounds" value="0%,0%,50%,50%" />
    </bind>
    <bind component="matrix" role="pause" />
    <bind component="glasses" role="stop" />
    <bind component="glassesContext" interface="glassesAdIn" role="start" />
  </link>
  ...
</body>
</ncl>
```

Figura 4 – Definição do sincronismo através de elos.

O primeiro relacionamento (*link01*) especifica o disparo da apresentação da propaganda da pilha (ação *start*) no momento em que a bateria é apresentada (condição *onBegin*) por Morpheus no filme (evento de apresentação da âncora *anchorBattery*).

O segundo exemplo de relacionamento (*link04*) ilustra o sincronismo baseado em interatividade. Esse relacionamento determina que se o botão amarelo do controle remoto for pressionado enquanto o ícone dos óculos estiver sendo exibido (condição *onKeySelection*), deve o vídeo *matrix* ser redimensionado para ocupar o ¼ superior esquerdo da tela (ação *setBounds*) e após pausado (ação *pause*), o ícone dos óculos ser

removido da tela (ação *stop*) e o vídeo com a propaganda da ótica ser exibido (ação *start*).

Como pode ser observado, os elos NCL referem-se a um conector (atributo *xconnector*). Os conectores NCL são definidos em um módulo à parte da linguagem [SoRo05] e oferecem recursos para que relações de sincronismo, tanto de causalidade como de restrição, sejam especificadas independentemente dos objetos que participem da relação. Papéis (*roles*) são especificados em cada conector para descrever como objetos que façam seu uso podem participar da relação. Os conectores criados por um autor especialista (não necessariamente o autor do programa) podem ser colocados em uma biblioteca (base) separada e reusados em diferentes documentos. No exemplo, a base denominada *maestro.conn* foi importada. Dessa forma, um autor não especialista de um programa não-linear só precisa especificar em um elo, quais nós e interfaces desempenham os papéis do conector selecionado, como ocorreu no exemplo.

4.6. Alternativas, Importação e Reuso

Além das entidades básicas apresentadas nas seções anteriores, NCL define um conjunto de elementos e atributos que oferecem suporte à construção de documentos adaptativos. O principal suporte à adaptação é a possibilidade do autor definir alternativas de nós (objetos de mídia e contexto) e de descritores, através dos elementos *switch* e *descriptorSwitch*, respectivamente. Para especificar alternativas, o autor deve antes definir uma base de regras de seleção no cabeçalho do documento. As regras podem ser simples ou compostas, e constituem-se em expressões que comparam atributos e valores. Os atributos podem representar parâmetros do terminal de acesso, preferências do usuário e outras informações que permitam contextualizar o programa NCL.

Um *switch* NCL possui uma seqüência de nós, estando cada nó associado a uma determinada regra. O nó selecionado será o primeiro da seqüência que tiver a regra associada satisfeita. Similarmente, regras são associadas a descritores para compor um *descriptorSwitch*. Enquanto *switches* de nós oferecem adaptação da estrutura e do conteúdo, *switches* de descritores auxiliam na adaptação da forma da apresentação dos documentos.

A Figura 5 destaca as mudanças no exemplo Matrix para que, ao invés de uma única opção de formulário, seja dada ao telespectador a opção de visualizar o formulário de compra em função do idioma preferencial.

```
<ncl ...>
<head>
  ...
  <ruleBase>
    <rule id="rEn" var="language" op="eq" value="english" />
    <rule id="rPt" var="language" op="eq" value="portugues" />
  </ruleBase>
</head>
<body>
  ...
  <context id="glassesContext">
    ...
    <switch id="form">
      <bindRule rule="rPt" component="ptForm"/>
    </switch>
  </context>
</body>
</ncl>
```

```
<bindRule rule="rEn" component="enForm"/>
<media id="ptForm" type="text" descriptor="formDesc" src="ptForm.html"/>
<media id="enForm" type="text" descriptor="formDesc" src="enForm.html"/>
</switch>
...
</context>
...
</body>
</ncl>
```

Figura 5 – Definição de alternativas de conteúdo para um programa.

Um outro recurso importante de NCL é a facilidade de reuso das especificações. Todas as bases especificadas nos cabeçalhos (regiões, descritores, regras, conectores etc.) podem importar elementos de outros documentos NCL, como ocorreu na definição dos conectores (Figura 4). Um elemento especial denominado *importedDocumentBases* também pode ser incluído no cabeçalho, permitindo que um documento NCL reuse elementos de outros documentos NCL. Elementos NCL podem fazer reuso das especificações realizadas em elementos das bases importadas ou em outros elementos do próprio documento através do atributo *refer*. O elemento que referencia e o elemento referenciado passam então a ser considerados como o mesmo elemento (o mesmo *id*).

5. Edição em Tempo de Exibição

O *middleware* Ginga é capaz de receber, via carrossel de dados do padrão DSM-CC [ISO98], a especificação de um programa NCL e controlar a exibição sincronizada de seus objetos.

Através do mecanismo de sincronização baseado em elos da NCL, e de um protocolo de sinalização baseado em eventos DSM-CC, foi possível construir um modelo de edição de programas de TV em paralelo com suas exibições [MCRS05]. Conforme mencionado na Seção 3, a edição de programas não-lineares durante suas exibições é uma facilidade desejável, em especial para programas gerados ao vivo.

Comandos de edição foram definidos, permitindo que o *middleware* seja notificado para adicionar ou remover uma entidade NCL (objeto de mídia, âncora, elo, contexto etc.) em tempo de exibição. Esses comandos são enviados através de eventos de *sincronização* (*stream events*) oferecidos pelo padrão DSM-CC. O *middleware* Ginga efetua a operação e recalcula as suas estruturas de dados de execução para que a mudança surta o efeito na exibição do programa.

Incluir e retirar objetos e elos de contextos são operações triviais em documentos NCL, uma vez que não implicam em mudanças na estrutura do documento. Note que a inclusão de tal facilidade em modelos de sincronismo baseados em composições seria muito difícil.

6. Tecnologias Relacionadas – Comparação

Como comentado na Seção 2.2, os principais *middlewares* declarativos existentes [ETSI05, ATSC05, ARIB04] são baseados em tecnologias desenvolvidas para a Web, mais especificamente XHTML [W3C02] com suporte a CSS [W3C98], DOM [W3C04a] e ECMAScript [ECMA99].

De um modo geral, XHTML favorece a autoria em função da sua simplicidade e da sua disseminação como linguagem para especificação de documentos na Web. Entretanto, as funcionalidades dessa linguagem são restritas quando existe a necessidade de especificar o sincronismo na sua forma mais ampla, uma vez que XHTML restringe-se à formatação para apresentação e à definição de relações de referência entre documentos. Para superar essa limitação, as linguagens DVB-HTML (sistema europeu) [ETSI05], XDMML (sistema americano) [ATSC05] e BML (sistema japonês) [ARIB04] adotam XHTML em conjunto com a linguagem ECMAScript. Todas essas linguagens acrescentam pequenas extensões à XHTML, mas apenas BML define extensões para descrever alguns tipos de sincronismo. Entretanto, mesmo as novas marcações definidas precisam, geralmente, ser complementadas com códigos ECMAScript para descrever a ação a ser executada no relacionamento de sincronização. Além disso, certos relacionamentos de sincronização espaço-temporal, que não se encaixam nas extensões definidas por BML, também devem ser definidos através do uso de ECMAScript, por meio de uma especificação embutida, direta ou indiretamente, em um objeto XHTML. ECMAScripts apresentam um propósito mais geral que apenas a especificação de sincronismo, oferecendo uma maior flexibilidade para construção dos programas. No entanto, o nível de abstração oferecido para os autores é baixo, obrigando que os criadores dos programas de TV interativa trabalhem, na realidade, como programadores de software. Por ser uma linguagem orientada a sincronismo de mídias, NCL oferece, exclusivamente através do paradigma declarativo, um nível mais alto de abstração para a autoria de programas não-lineares, ao mesmo tempo que coloca uma rica expressividade para a descrição dos relacionamentos temporais e espaciais entre os objetos de mídia.

As linguagens DVB-HTML, XDMML e BML permitem a definição de características espaciais de apresentação de um componente em um objeto separado, utilizando CSS, similar ao modelo de regiões e descritores da linguagem NCL. Todavia, as características temporais de apresentação, tais como início, duração e fim da exibição, permanecem embutidas na definição dos objetos. Mais ainda, todas as linguagens, por serem baseadas em XHTML, implementam a sincronização temporal com interatividade por meio de elos embutidos no conteúdo dos documentos (exceto quando o sincronismo é realizado por eventos de sincronismo DSM-CC). Como desvantagem, isso faz com que a reutilização de um objeto (conteúdo) herde obrigatoriamente os seus relacionamentos. Em NCL, uma página XHTML é tratada como mais um objeto de mídia, e todas as informações temporais e de sincronização a ela relacionadas são descritas externamente ao objeto (através dos elos e descritores). Dessa forma, uma mesma página poderia assumir comportamentos distintos em programas NCL diferentes, favorecendo o reuso.

Recentemente, os grupos de padronização dos vários sistemas internacionais de TV digital estão dedicando uma atenção maior à importância do *middleware* declarativo e estão sinalizando com uma tendência a buscar linguagens com foco em sincronismo de mídias. O padrão W3C SMIL [W3C05] e a alternativa declarativa para especificação de documentos MPEG-4, denominada XMT [ISO01], são duas opções cogitadas, além da antiga proposição MHEG-5 [ISO97]. SMIL e XMT têm as limitações comentadas na Seção 3 ao explorar composições como mecanismo de sincronização, enquanto o MHEG-5 já foi criticado por sua complexidade. O paradigma de elos para sincronização utilizado pela linguagem NCL, com uso de composições para estruturação, não

necessariamente temporal, dos documentos, contorna as potenciais desvantagens das composições temporais. A separação entre elos (relacionamentos) e conectores (relações), por outro lado, procura não tornar complexa a tarefa de autoria.

7. Conclusões

As ferramentas e linguagens multimídia/hipermídia desempenham um papel importante nessa atual fase de convergência que presenciamos das redes e serviços, principalmente na atual transição do sistema de TV analógico para o sistema de TV digital. A utilização de tecnologias que não imponham custos adicionais de licença é um fator estratégico para favorecer a inclusão social. Além disso, a disseminação do uso dessas tecnologias junto aos responsáveis pela geração de conteúdo privilegia uma apropriação cultural, podendo fazer com que as oportunidades na área de entretenimento digital venham a estar melhor distribuídas geograficamente e socialmente no país.

NCL foi concebida como uma linguagem para autoria de documentos hipermídia. Sua organização modular permite que funcionalidades da linguagem sejam utilizadas de acordo com as necessidades das aplicações. Como resultado, um subconjunto das funcionalidades da linguagem foi selecionado para formar um perfil apropriado à composição de programas não-lineares de TV digital, conforme descrito neste artigo. Este perfil também foi proposto como alternativa de linguagem de especificação de programas para o *middleware* declarativo do Sistema Brasileiro de TV Digital.

É importante observar que NCL é uma linguagem para integração e sincronização de mídias. Dessa forma, as aplicações declarativas desenvolvidas para os principais sistemas de TV digital citados não precisam, necessariamente, ser adaptadas para NCL, uma vez que a linguagem HTML/XHTML é entendida apenas como mais um tipo de objeto de mídia em NCL. A vantagem de utilizar NCL é usufruir de construções declarativas para a especificação da estruturação e do sincronismo dos programas, dispensando em muitas vezes a programação por scripts (ou outras estratégias de programação algorítmica).

Um conjunto de ferramentas para autoria gráfica e controle das apresentações de documentos NCL foi desenvolvido e está disponível através de projetos de software aberto². O leitor interessado pode também encontrar no endereço <http://www.telemidia.puc-rio.br/papers/semish2006/> o arquivo NCL contendo o exemplo completo apresentado na Seção 4, assim como o código executável do formatador NCL para exibição do documento e um vídeo que ilustra a aplicação da linguagem em outros exemplos de programas de TV interativa.

Referências

[ARIB04] ARIB – Association of Radio Industries and Businesses “Data Coding and Transmission Specifications for Digital Broadcasting Volume 2: XML-Based Multimedia Coding Schema”, STD-B24 Versão 4, fevereiro de 2004.

² <http://opensource.telemidia.puc-rio.br>

- [ATSC05] ATSC - Advanced Television Systems Committee “ATSC Standard: Advanced Common Application Platform (ACAP)”, Padrão A/101, Washington, EUA, agosto de 2005.
- [CoRS04] Coelho R.M., Rodrigues R.F., Soares L.F.G. “Integração de Ferramentas Gráficas e Declarativas na Autoria de Arquiteturas Modeladas através de Grafos Compostos”, X Simpósio Brasileiro de Sistemas Multimídia e WEB (WebMedia), Ribeirão Preto, outubro de 2004. pp. 3-11.
- [ECMA99] ECMA “Standardizing Information and Communication Systems. ECMAScript Language Specification”, Padrão ECMA 262, 3a Edição, 1999.
- [ETSI05] ETSI – European Telecommunications Standards Institute. “Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.1.1”, Especificação Técnica ETSI TS 102 B12, maio de 2005.
- [ISO97] ISO/IEC. “Information Technology - Coding of multimedia and hypermedia information - Part 5: Support for base-level interactive applications”. Padrão ISO 13522-5, 1997.
- [ISO98] ISO/IEC - International Organization for Standardization. “Information technology – Generic coding of moving pictures associated audio information - Part 6: Digital Storage Media Command and Control”, 13818-6, 1998.
- [ISO01] ISO/IEC - International Organization for Standardization. “14496-1:2001, Coding of Audio-Visual Objects – Part 1: Systems”, 2a Edição, 2001.
- [Mina05] Minassian A.A. “A TV Digital – Convergência de Mídia”, Seminário Nacional de TV Digital, Belo Horizonte, Brasil, novembro de 2005. Disponível em http://www.anatel.gov.br/Tools/frame.asp?link=/acontece_anatel/palestras/comunicacao_massa/convergencia_midia.pdf
- [MCRS04] Moreno M.F., Costa R.M., Rodrigues R.F., Soares L.F.G. “Edição de Documentos Hiperídia em Tempo de Exibição”, XI Simpósio Brasileiro de Sistemas Multimídia e WEB (WebMedia), Poços de Caldas, novembro de 2005.
- [SoRo05] Soares L.F.G., Rodrigues R.F. “Nested Context Model 3.0 Part 5 – NCL (Nested Context Language) Main Profile”. Monografias em Ciência da Computação do Departamento de Informática da PUC-Rio, MCC 26/05, 2005.
- [W3C98] W3C. “Cascading Style Sheets, level 2 - CSS 2 Specification”, W3C Recommendation, 1998.
- [W3C99] World-Wide Web Consortium. “Namespaces in XML”. *W3C Recommendation*, janeiro de 1999.
- [W3C02] World-Wide Web Consortium. “XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition)”. W3C Recommendation, agosto de 2002.
- [W3C04a] World-Wide Web Consortium. “Document Object Model (DOM) Level 3 – Core Specification”. W3C Recommendation, 2004.
- [W3C04b] World-Wide Web Consortium. “XML Schema Part 1: Structure – 2nd Edition”. W3C Recommendation, outubro de 2004.
- [W3C05] World-Wide Web Consortium. “Synchronized Multimedia Integration Language (SMIL 2.1)”. W3C Recommendation, dezembro de 2005.