PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Rafael Ferreira Rodrigues

Rodrigo Laiola Guimarães

Relatório comparativo da produção de programas hipermídia interativos em XHTML, SMIL/GRiNS e NCL/Maestro

MONOGRAFIA DA DISCIPLINA DE FUNDAMENTOS DE SISTEMAS MULTIMÍDIA

DEPARTAMENTO DE INFORMÁTICA

Programa de Pós-Graduação em Informática

Rio de Janeiro

Dezembro de 2005

PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Rafael Ferreira Rodrigues

Rodrigo Laiola Guimarães

Relatório comparativo da produção de programas hipermídia interativos em XHTML, SMIL/GRiNS e NCL/Maestro

Monografia da Disciplina de Fundamentos de Sistemas Multimídia

Monografia apresentada como requisito parcial para aprovação na disciplina de Fundamentos de Sistemas Multimídia do Programa de Pós-Graduação em Informática da PUC-Rio.

Orientador: Luiz Fernando Gomes Soares

Rio de Janeiro, dezembro de 2005

PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Rafael Ferreira Rodrigues

Rodrigo Laiola Guimarães

Relatório comparativo da produção de programas hipermídia interativos em XHTML, SMIL/GRiNS e NCL/Maestro

Monografia apresentada como requisito parcial para aprovação na disciplina de Fundamentos de Sistemas Multimídia do Programa de Pós-Graduação em Informática da PUC-Rio.

Luiz Fernando Gomes Soares
Orientador
Departamento de Informática - PUC-Rio

Rio de Janeiro, 12 de dezembro de 2005

Rafael Ferreira Rodrigues

Graduado em Engenharia de Computação pelo Instituto Militar de Engenharia (IME) em 2004. Atualmente, integra o grupo de pesquisadores do Laboratório TeleMídia da PUC-Rio, desenvolvendo pesquisa na área de Redes de Computadores e Sistemas Hipermídia.

Rodrigo Laiola Guimarães

Graduado em Engenharia de Computação pela Universidade Federal do Espírito Santo (UFES) em 2004. Atualmente, integra o grupo de pesquisadores do Laboratório TeleMídia da PUC-Rio, desenvolvendo pesquisa na área de Redes de Computadores e Sistemas Hipermídia.

Ficha Catalográfica

Rodrigues, Rafael Ferreira; Guimarães, Rodrigo Laiola

Descrição da Ficha Catalográfica

Informação Técnica da Ficha Catalográfica

Natureza da Ficha Catalográfica

Inclui referências bibliográficas.

Autoria; Conteúdo hipermídia interativo; Linguagens declarativas

Dedicamos este trabalho a todos àqueles que acreditam que a ousadia e o erro são caminhos para as grandes realizações.

Agradecimentos

Nossa sincera gratidão e admiração pelo nosso orientador Luiz Fernando Gomes Soares por sua tamanha dedicação e incessante esforço em nos ajudar durante o desenvolvimento deste trabalho de pesquisa científica.

Aos nossos colegas do TeleMídia, pela amizade, companheirismo e ajuda prestados. Em especial a Rogério e Rogerinho (Jr.) pela atenção, paciência e boa vontade em nos ensinar, tendo sido verdadeiros guias para o desenvolvimento deste trabalho, suas participações foram fundamentais.

À CAPES, ao CNPq e ao TeleMídia pelo apoio financeiro.

Resumo

Rodrigues, Rafael Ferreira; Guimarães, Rodrigo Laiola. **Relatório comparativo da produção de programas hipermídia interativos em XHTML, SMIL/GR*i*NS e NCL/Maestro.** Rio de Janeiro, 2005. 27p Monografia da Disciplina de Fundamentos de Sistemas Multimídia - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Com o advento da adoção de um padrão de TV digital pelo Brasil, tem crescido o interesse pela análise das possíveis alternativas nas mais diversas áreas que irão compor esse sistema. Nesse contexto, existe uma grande variedade de linguagens que podem ser utilizadas no desenvolvimento de programas multimídia/hipermídia interativos. Dentre os paradigmas de programação existentes, as linguagens declarativas estão despontando como a nova aposta para a autoria em TV digital interativa. Cabe então a análise rigorosa para que a melhor decisão seja tomada visando atender aos requisitos e necessidades desse novo sistema que será implantado no país. Esta monografia tem como objetivo levantar os pontos positivos e negativos encontrados na elaboração de conteúdo hipermídia interativo em XHTML, SMIL/GR*i*NS e NCL/Maestro, bem como fazer um estudo comparativo entre essas linguagens.

Palavras-chave

Autoria; Conteúdo hipermídia interativo; Linguagens declarativas

Sumário

Lista de figuras

1 Introdução

Com o advento da futura adoção de um padrão de TV digital interativa pelo Brasil, tem crescido o interesse pela análise das possíveis alternativas nas mais diversas áreas que compõem um sistema de TV digital.

Este capítulo descreve as motivações e os objetivos desta monografia, assim como apresenta sua estrutura.

1.1. Motivação

Nesse mesmo contexto, nota-se a grande variedade de linguagens que podem ser utilizadas no desenvolvimento de programas multimídia/hipermídia interativos. Dentre os paradigmas de programação existentes, as linguagens declarativas são apontadas por parte da comunidade científica como a nova aposta para a área de autoria em TV digital. Cabe então a análise rigorosa das alternativas para que a melhor decisão seja tomada visando atender aos requisitos desse novo sistema que está para ser implantado no país.

1.2. Objetivos

Esta monografia tem como principal objetivo levantar os pontos positivos e negativos encontrados durante a elaboração de conteúdo hipermídia interativo em XHTML, SMIL/GRiNS e NCL/Maestro, bem como fazer um estudo comparativo entre essas linguagens.

A análise é direcionada aos requisitos que previmos de antemão, os quais, em alguns casos puderam, e em outros não, ser satisfeitos com os recursos e funcionalidades existentes nessas linguagens. Além disso, são apontadas as facilidades e dificuldades com as quais nos deparamos na utilização das linguagens e ferramentas de autoria.

1.3. O programa

Para a elaboração do programa partiu-se da idéia de se desenvolver um conteúdo de cunho jornalístico. A idéia evoluiu e chegou a um formato que consiste de três canais sendo: um jornalístico, um de esportes e o último musical. É possível a qualquer momento realizar a troca entre eles, emulando uma programação ao vivo.

O canal jornalístico consiste de dois vídeos em seqüência, sendo que apenas o primeiro programa apresenta interatividade. Esta interatividade se apresenta de duas formas: no primeiro caso, uma opção de acessibilidade permite que usuários surdos possam entender o conteúdo da notícia através de uma legenda visual com interprete da linguagem de sinais; no segundo caso, a interatividade está na forma da apresentação de um ícone indicando a presença de uma publicidade.

No canal musical encontra-se um clipe que irá apresentar um ícone publicitário em sincronismo com a música, e que ao ser clicado, causará a reprodução do vídeo relativo ao anunciante, sendo este sem a presença de efeitos sonoros.

Por fim, no canal de esportes é realizada uma votação para a eleição de um vídeo dentre três apresentados, que será reapresentado ao final da programação. É exibido ainda um programa que consiste de um jogo de futebol onde são ilustrados, por meio de eventos selecionáveis, um gol ocorrido em outra partida e um anúncio publicitário.

1.4. Estrutura da Monografia

Esta monografia encontra-se organizada como a seguir. O Capítulo 2, 3 e 4 apresentam os pontos positivos e negativos encontrados durante o desenvolvimento do trabalho em XHTML, SMIL/GR*i*NS e NCL/Maestro, respectivamente. No início desses capítulos é feita uma pequena introdução a cada uma dessas linguagens, e ao final é dada a impressão geral da sua utilização.

Por fim, o capítulo 5 tece as conclusões, e faz uma análise comparativa entre as três linguagens utilizadas, apontando possíveis trabalhos futuros.

2 XHTML

O XHTML (e*Xtensible HyperText Markup Language*) [PAAÇ02] é uma linguagem de marcação especificada pelo W3C (*World Wide Web Consortium*), um consórcio que desenvolve tecnologias interoperáveis (especificações, guias, softwares e ferramentas) para a Web.

O XHTML consiste num modelo elaborado a partir do HTML (*HyperText Markup Language*) na sua versão 4.01 [W3C99a]. Entretanto, o HTML possui uma sintaxe muito mais flexível, enquanto o XHTML, que é baseado em XML (*eXtensible Markup Language*) [W3C04a], possui uma sintaxe muito mais estrita. Portanto, o XHTML possui certas restrições como a necessidade de *tags* fechadas ou nomenclatura de tags e atributos *case sensitive*.

Com relação ao modelo conceitual hipermídia, as restrições do XHTML são as mesmas do HTML, uma vez que ambas as linguagens seguem o mesmo modelo de documentos [SoCR04].

Nas próximas seções é feita uma análise dos pontos positivos e negativos encontrados na realização do trabalho em XHTML.

2.1. Pontos positivos

Algumas facilidades do XHTML notadas durante o trabalho são:

- Muito parecida com HTML: isso facilita bastante o desenvolvimento de um documento por quem já conhece o HTML;
- A programação utilizando scripts é bastante simples para quem já tem uma noção de programação;
- O script apesar de ser uma ferramenta de difícil aplicação sob a ótica de um usuário leigo constiui uma ferramenta poderosa. Este possibilita a construção de uma grande variedade de meios de interação.

2.2. Pontos negativos

Além das limitações do HTML, podemos destacar como dificuldades encontradas durante o desenvolvimento do trabalho:

- Sintaxe muito estrita: o “preciosismo” demandado na elaboração de um documento pode vir a atrapalhar o processo de autoria. Uma sugestão é utilizar uma ferramenta que permita ao autor abstrair essas necessidades para que ele se concentre apenas na criação;
- Reaproveitamento de elementos e seus eventos associados: durante o desenvolvimento do trabalho gostaríamos de configurar dinamicamente ações a serem executadas quando o usuário clicasse sobre um vídeo. Porém não conseguimos fazer isso. Como alternativa, tivemos que criar inúmeros objetos que funcionavam da mesma forma, sendo que executavam ações diferentes para o mesmo evento de mouse;
- Manipulação de elementos `DIV`: foi através desses elementos que conseguimos utilizar a idéia de camadas, colocando mídias visuais umas sobre as outras. Nem sempre obtivemos o efeito esperado, e para que o comportamento desses elementos fosse satisfatório tivemos que utilizar vários truques, como, por exemplo, desaparecer com um vídeo por algum tempo para que o mesmo conseguisse sobrepor outro vídeo;
- Garantia de Qualidade de Serviço: o *player* (*browser*) utilizado aparentemente não dá nenhuma garantia de QoS (*Quality of Service*). Em alguns casos, só depois de alguns segundos do tempo de execução especificado que a mídia era iniciada. Por conseguinte, é perdido todo o sincronismo outrora desejado. Cabe ressaltar que para que tal afirmativa fosse feita com segurança, precisaríamos assegurar que nenhum outro aplicativo estivesse “roubando” processamento da máquina;
- Utilização de scripts: apesar de ser muito fácil utilizar scripts, quando se deseja fazer algo um pouco mais “interessante” se tem um trabalho imenso. Um exemplo disso é a dificuldade para se sincronizar as mídias. Outro ponto a se ressaltar é a dificuldade de reuso de funções quando se utiliza a função `setTimeout` do JavaScript (Netscape Communications Corporation). Como parâmetros dessa função é passada a ação a ser executada e o momento de execução. Sendo essa ação uma outra função

que possui variáveis de entrada, o *bind* entre essa variável e a função só é feito quando ela é chamada. Dessa forma, é complicado utilizar uma variável nessa situação e isso dificulta muito o reuso de funções.

- O tratamento de mídias do tipo vídeo requer recursos específicos da plataforma, como foi no caso do programa elaborado onde se usou o ActiveX. Não existe uma padronização para a execução de tal mídia, apenas para imagens estáticas através do elemento `img`.

2.3. Outras considerações

O XHTML se mostra como uma linguagem bastante versátil. Por um lado ela proporciona uma forma simples de criar uma apresentação. Ela usa recursos avançados como XForms (para a criação de formulários) e XFrames (para a criação dos frames, já presentes no HTML 4.01). Porém, toda lógica por trás da apresentação se mostra complexa devido ao uso de scripts, os quais exigem uma maior capacidade técnica do usuário.

Existe ainda uma vasta gama de ferramentas para construir uma apresentação XHTML, contudo nenhuma para a construção da lógica dos scripts.

Dada a complexidade de se construir documentos hipermídia em XHTML, durante o trabalho foi desenvolvido o XiTV – XHTML iTV Framework [RoGui05], um *framework* que visa abster do autor toda a complexidade da criação de seu programa hipermídia interativo. A Figura 1 apresenta a arquitetura do XiTV.

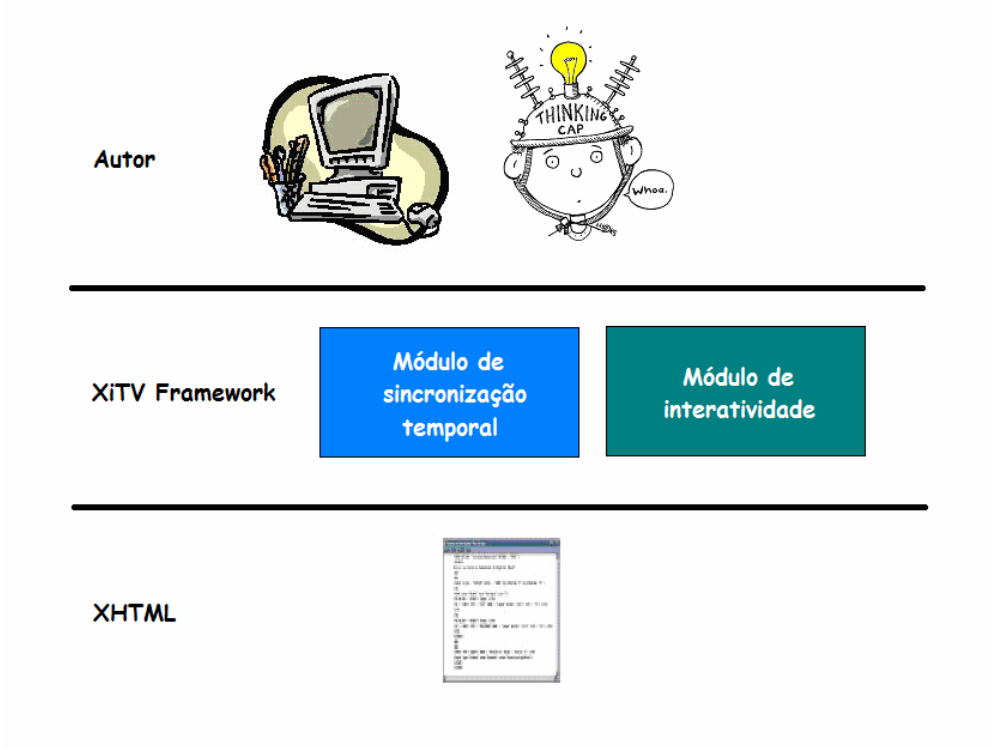

FIGURA 1. Arquitetura do XiTV.

O funcionamento desse framework é bem simples. Ao escrever o programa, o autor precisa especificar quais serão os vídeos que irão compor os canais. Entre canais, a semântica de sincronização temporal é paralela, enquanto que dentro de um mesmo canal a semântica é seqüencial. Essa sincronização é feita pelo módulo de sincronização temporal. Além disso, para cada vídeo é disponibilizada uma opção de acessibilidade que permite que uma legenda visual possa ser exibida juntamente com o áudio e vídeo principal. Outra funcionalidade fornecida é a de exibição de um plantão, que a qualquer momento da apresentação do documento, exibe a notícia em todos os canais, não deixando é claro de manter o vídeo principal do canal corrente em uma região menor, para que o telespectador possa continuar ligado no que estava acedendo. Ao término do plantão o áudio principal é restabelecido, e vídeo principal é novamente maximizado. Podemos entender o plantão como sendo um canal com prioridade sobre todos os demais, e o módulo de sincronização temporal é o responsável por sua exibição.

O framework dá ainda suporte a configuração de propagandas. Essas podem ser interativas ou não. No primeiro caso, é possível especificar quando a

propaganda irá aparecer e qual será a imagem ou vídeo que será exibido após a interação com o primeiro elemento audiovisual. Além disso, é possível configurar se ao ocorrer a interatividade o áudio do programa principal será ou não desabilitado. Isso é feito através do módulo de interatividade.

A idéia desse framework é permitir que pessoas criativas possam desenvolver programas interativos interessantes sem precisar conhecer profundamente XHTML. Por outro lado, usuários mais familiarizados com a linguagem podem estender essa aplicação para que efeitos mais interessantes possam ser suportados.

Um trabalho futuro pode ficar por conta da integração desse framework com uma ferramenta de edição que possibilite a edição de um documento XHTML em múltiplas visões, como acontece com o SMIL/GRiNS e NCL/Maestro (próximos capítulos).

3 SMIL

O SMIL (*Synchronized Multimidia Integration Language*) [BuRu04] é uma linguagem baseada em XML, e especificada pelo W3C [W3C01b], que permite a descrição de apresentações multimídia interativas.

Essa linguagem é bastante atraente para o desenvolvimento de apresentações, pois nela é possível se definir de forma simples e bem intuitiva como os objetos de mídia estarão dispostos no tempo. Tudo isso é feito através de um conjunto básico de composições que possuem semântica de sincronização. Com esses recursos é possível compor grande parte dos documentos desejados.

No desenvolvimento do trabalho foi utilizado o ambiente de autoria GR*i*NS Pro Editor for SMIL 2.0 [GR*i*NS/Pro], o qual possui várias visões integradas (visão de apresentação, temporal, de layout e textual). Embora projetado segundo o paradigma baseado na estrutura, essa ferramenta destaca-se por sua visão temporal para concepção de documentos.

Nas próximas seções é feita uma análise dos pontos positivos e negativos encontrados na realização do trabalho na linguagem SMIL.

3.1. Pontos positivos

São algumas facilidades do SMIL/GR*i*NS notadas durante a elaboração do trabalho:

- Modularização: separação da definição das regiões de exibição da parte responsável pela sincronização temporal. Essa distinção facilita muito a elaboração e edição de um documento multimídia/hipermídia;
- Com o conjunto básico de composições disponível é possível montar uma quantidade imensa de apresentações. Além disso, essas composições possuem uma semântica que, para apresentações simples, fica clara até para quem não está acostumado a desenvolver documentos hipermídia nessa linguagem;

- Manipulação de camadas: foi através desse recurso que conseguimos colocar mídias visuais umas sobre as outras. O funcionamento dessa funcionalidade foi bastante consistente e foi muito fácil conseguir o efeito desejado;
- A existência de um player associado à ferramenta de autoria é de extrema importância, pois em tempo de criação é possível ter noção de como está ficando a apresentação. Isso agiliza o processo de desenvolvimento e diminui consideravelmente o retrabalho;
- A existência de várias visões permite ao autor lançar mão lançar das suas habilidades da forma que mais lhe convém. Vale destacar que se usadas complementarmente, pode ser explorado o que há de melhor existe em cada uma das visões;
- Extensões da linguagem fornecem uma série de funcionalidades interessantes para dar um toque mais refinado às apresentações hipermídia. Um exemplo disso é o módulo de animação do RealPlayer, o qual fornece uma série de efeitos de visuais (transições, transparência, etc);
- Relativa facilidade na implementação de eventos interativos. Os eventos e semânticas de sincronização que precisávamos foram possíveis de serem implementados com as funcionalidades disponíveis na linguagem;
- Não demanda conhecimento avançado para se elaborar apresentações. Na verdade, já na elaboração do primeiro documento é possível captar a essência do modelo por trás dessa linguagem.

3.2. Pontos negativos

Quanto às dificuldades encontradas vale a pena ressaltar:

- O grande problema não fica nem por conta da falta de suporte na linguagem, mas sim pela implementação dos players disponíveis. Nenhuma das versões que encontramos foi capaz de atender totalmente as nossas necessidades durante o trabalho. Enquanto uns eram muito bons na sincronização temporal de mídias, não davam suporte a interatividade, e vice-versa. Isso consegue ferir qualquer ímpeto, pois a exibição de um documento após grande trabalho fica bem abaixo das expectativas;

- Garantia de Qualidade de Serviço: aparentemente os players utilizados não dão nenhuma garantia de QoS. Em alguns casos, só depois de alguns segundos do tempo de execução especificado que a mídia era iniciada. Por conseguinte, era perdido todo o sincronismo outrora desejado. Um exemplo disso foi a implementação da troca de canal, na qual era utilizado o atributo soundLevel para modificar o volume dos objetos de mídia em uma região. Esse mecanismo fazia com que até mesmo a sincronização entre o som e vídeo dentro de um mesmo arquivo MPEG fosse perdida. Entretanto vale destacar que uma análise mais conclusiva demanda mais testes em um ambiente "preparado";
- Sintaxe muito estrita: o "preciosismo" demandado na elaboração de um documento pode vir a atrapalhar o processo de criação. Uma alternativa para contornar isso é utilizar uma ferramenta que auxilia o usuário criador, e isso é possível, por exemplo, no GR*i*NS;
- Apesar de com as funcionalidades da linguagem ser possível desenvolver grande parte dos efeitos desejados, falta um recurso que forneça um maior poder de expressão a usuários avançados. Isso foi notório durante o trabalho quando foi tentado utilizar na linguagem algum mecanismo que desse suporte a contabilização de votos. Dessa forma, falta suporte a possibilidade de se utilizar alguns recursos disponíveis em uma linguagem procedural qualquer;
- Falta de portabilidade entre players: pelo que pudemos notar em alguns casos, a linguagem SMIL é estendida (por exemplo, no GR*i*NS), sendo essas novas linguagens específicas para alguns players. Isso faz com que recursos disponíveis em uma apresentação não possam ser exibidos em todos os players. Muitas vezes, a apresentação sequer pode começar a ser apresentada. Apesar disso, no GR*i*NS é possível se exportar o documento para um número limitado de players;
- A inexistência de composições com semânticas de sincronização temporal mais específicas demanda do projetista atenção na utilização de eventos para obter o efeito desejado. Um exemplo disso é quando se quer que a apresentação de dois vídeos dentro de uma composição paralela termine junto, mesmo as duas mídias tendo durações diferentes. Para obter o efeito desejado o SMIL lança mão de parâmetros que possuem uma semântica

muito parecida com a desempenhada por elos do paradigma de causalidade/restrição.

3.3. Outras considerações

O desenvolvimento de apresentações hipermídia em SMIL é relativamente fácil e rápido. Isso se deve ao fato de que suas composições com semântica temporal são muito intuitivas. A dificuldade aparece quando se deseja um comportamento um pouco diferenciado, como por exemplo, a mudança do funcionamento de uma composição ou interatividade. O que é feito para tratar essas questões é muito parecido com os elos utilizados em NCL (próximo capítulo), mas como não são apresentados com a mesma clareza do paradigma de causalidade/restrição, nem sempre é fácil se obter o comportamento desejado.

4 NCL

A linguagem NCL (*Nested Context Language*) [MuSS03] é uma linguagem declarativa para autoria de documentos hipermídia baseados no modelo conceitual NCM (*Nested Context Model*) [SoRM03].

Para compreender as características da linguagem NCL, é fundamental conhecer o modelo NCM, uma vez que as entidades principais desse modelo correspondem aos elementos da linguagem NCL. Nesse modelo um documento hipermídia é representado por um nó de composição, podendo conter um conjunto de nós, que podem ser objetos de mídia ou outros nós de composição, recursivamente, e ainda elos relacionando esses nós.

Nós de composição no modelo NCM não têm nenhuma semântica embutida, diferente das composições SMIL, por exemplo, que têm semântica temporal. Contudo são uma poderosa ferramenta no sentido de possibilitar o reuso. A semântica dos relacionamentos entre os componentes de um documento é feita através dos elos NCM, que podem especificar relações de referência, relações de sincronização, relações de derivação, relações entre tarefas de um trabalho cooperativo etc. Elos NCM são definidos fazendo referência a um conector hipermídia, que pode representar qualquer tipo de relação. Além da referência a um conector, um elo define um conjunto de *binds* associando papéis do conector a nós do documento. Assim sendo, elos podem representar relacionamentos multiponto entre vários nós de um documento.

Elos são agrupados em bases de elos, logo, o conjunto de elos de uma composição é dado pela união de suas bases de elos. Além do reuso de nós, o NCM também permite o reuso de elos e de bases de elos, tornando mais flexível a definição de elos no modelo.

O NCL é uma linguagem bastante atraente para o desenvolvimento de apresentações hipermídia interativas, pois nela é possível se definir de forma simples e intuitiva como os objetos de mídia estarão dispostos no tempo. Isso tudo é possível através da utilização de elos causais/restrição que dão toda a semântica de semântica de sincronização necessária.

4.1. Pontos positivos

Algumas facilidades do NCL notadas durante o desenvolvimento do trabalho são:

- Modularização: separação da definição das regiões de exibição da parte responsável pela sincronização temporal. Essa distinção facilita muito a elaboração e edição de um documento hipermídia interativo;
- Reuso: é dado suporte ao reuso de entidades, como por exemplo, descritores;
- Relativa facilidade na implementação de eventos interativos. Pensar em interatividade lançando mão de elos é fácil. Além disso, a sincronização utilizando o paradigma da causalidade/restrição é bastante intuitiva no desenvolvimento da seqüência lógica da apresentação. Os eventos e semânticas de sincronização que precisamos foram na maioria das vezes possíveis de serem implementados com os recursos da linguagem;
- Não demanda conhecimento avançado para se elaborar apresentações simples. Escrever o primeiro programa em NCL pode até levar um grande tempo, contudo os demais são feitos em muito menos tempo. A essência dessa linguagem é logo captada;
- Garantia de Qualidade de Serviço: o formatador Maestro se comportou bem na orquestração dos objetos de mídia, sendo estressado ao ponto de tocar 5 mídias de vídeo ao mesmo tempo. Porém, um estudo mais detalhado sobre a provisão de QoS deve ser realizado com um ambiente melhor "preparado" para que possam ser alcançados resultados mais conclusivos;
- Utilização de scripts: apesar de com as funcionalidades da linguagem ser possível desenvolver grande parte dos efeitos desejados, faltava um recurso que fornecesse um maior poder de expressão a usuários avançados. Isso foi notório durante o trabalho quando tentamos utilizar na linguagem algum mecanismo que desse suporte a contabilização de votos. Com a incorporação do NCLlet (script Lua [Lua]) à linguagem, uma imensa gama de novos serviços poderão ser oferecidos.

4.2. Pontos negativos

Quanto às dificuldades encontradas vale a pena ressaltar:

- Apresentações um pouco mais complexas não são triviais de se elaborar. Elas levam à criação de conectores, o que requer do autor um conhecimento mais profundo da linguagem. Na ferramenta de autoria não é dado qualquer suporte a esse tipo de problema. Uma forma de contornar este problema seria, ainda, a criação de uma base padrão com conectores mais complexos que os de Allen;
- Sintaxe muito estrita: o "preciosismo" demandado na elaboração de um documento pode vir a atrapalhar o processo de criação. A utilização de uma ferramenta de autoria como o Maestro, pode auxiliar o usuário criador a se abster de tal preocupação.
- Regiões como camadas: a nossa idéia era sobrepor regiões para dar um efeito visual mais atraente a apresentação. Não obstante, nem sempre obtivemos o efeito esperado;
- O ambiente de autoria Maestro ainda está muito aquém de atender a demanda dos mais variados tipos de usuários. Essa ferramenta só serve pra quem não só conhece bem a linguagem, como também sabe seus macetes. Isso porque o ambiente ainda não prioriza o que o usuário quer fazer, mas sim o que a linguagem tem a oferecer;
- As mídias sempre se ajustam ao tamanho da região; Não é possível ter uma região com uma mídia com seu tamanho original (e menor que a região);
- Não dá suporte a transparência de figuras: isso é uma deficiência grave visto que limita quem está criando. Grande parte do impacto de um programa se deve aos efeitos audiovisuais existentes, e recursos como transparência sempre dão um toque a mais as apresentações;
- Na presença de uma âncora em um nó de mídia (por exemplo, uma imagem) que conten há um evento de seleção, quando o mouse é passado sobre ela não há mudança do cursor do mouse. Dessa forma o usuário precisa desvendar aonde quem fez o programa colocou a interação;
- Não é nada agradável ter que transformar uma imagem estática em um vídeo para que este possa ser exibido durante um determinado período de tempo. Essa limitação é bastante significativa e demanda um maior poder de armazenamento, processamento e transmissão;

- As mensagens de erros de compilação são pouco descritivas a nível de usuário leigo. Além disso não há uma separação de erros de sintaxe (na estrutura do documento) ou de execução (*binds* inválidos, arquivos não encontrados);
- A ferramenta de autoria não valida a sintaxe do documento;
- Os conectores constituem uma ferramenta poderosa para criação de interatividade, porém durante o projeto alguns conectores, fiéis à sintaxe, não produziram o efeito desejado. Foi necessário, por exemplo, durante o projeto, desestruturar todo o documento por conta de um conector que não executava sua ação quando um atributo de um contexto era modificado. E ainda, atributos como "*sound level*" e "*visibility*", são apenas levados em conta quando alterados durante a execução da mídia, alterações prévias não provocaram nenhum efeito. E, por fim, o mesmo atributo "*visibility*" funciona apenas para a mídia vídeo.

4.3. Outras considerações

Quando pensamos nas potencialidades do NCL, logo viajamos no pensamento, pois parece óbvio que grandes coisas irão surgir daí. Não obstante, a impressão que fica é que o NCL não foi criado em nenhum momento pensando em quem realmente vai utilizar essa linguagem, ou seja, os autores. Ferramentas de autoria que abstenham o autor da complexidade da linguagem serão determinantes para seu sucesso. Nesse contexto, cabe ressaltar que o ambiente Maestro ainda está muito aquém do que pode oferecer a seus usuários. Não só pelos bugs, como também pela falta de diálogo do ambiente com seu parceiro mais importante, o autor. É certo que se o ambiente deixar o autor se expressar com toda a sua criatividade, coisas antes inimagináveis poderão surgir.

Seguem algumas sugestões de melhorias para o ambiente Maestro:

- Disponibilização na visão textual de uma funcionalidade que permita colapsar o conteúdo de entidades (blocos XML). Tal funcionalidade ajuda bastante no processo de desenvolvimento de um documento;
- Disponibilização de uma opção para se fazer ou não um *wrap* (quebra) das linhas de código na visão textual. Isso permite que as linhas sejam ou não

quebradas de acordo com a preferência do usuário que está usando a ferramenta;

- Na árvore hierárquica, ícones mais intuitivos poderiam ser colocados para cada uma das entidades. Além disso, seria bom se fosse possível manipular (editar, excluir, etc.) entidades a partir dessa árvore;
- Uma sugestão para a visão textual é pesquisar na internet um editor de código aberto que apresente as características desejadas. Acreditamos que dessa forma muito trabalho pode ser poupado, e grande parte das funcionalidades encontradas em editores convencionais estaria presente no ambiente Maestro.
- Modificar o cursor do mouse sobre uma âncora de seleção.

5 Conclusões

Durante esta monografia procuramos apontar as facilidades e dificuldades encontradas no processo de desenvolvimento de um programa interativo em XHTML, SMIL e NCL, analisando também seus recursos e funcionalidades. Feita essa análise, é possível destacar um conjunto de diferenças entre essas linguagens.

Dado que todas são baseadas em XML, a necessidade de sintaxe bem escrita existe em todas elas. Como mencionado anteriormente, a utilização de um editor (ou um ambiente de autoria) pode facilitar o processo de elaboração de um documento.

Quanto à utilização de camadas para se alcançar efeitos visuais mais interessantes, a linguagem que mais se destacou foi o SMIL. O pior resultado ficou por conta do NCL, onde além de nem sempre obtermos o comportamento desejado, é preciso criar um vídeo de uma figura para que esta possa ser exibida durante um determinado período de tempo. Isso é extremamente limitante.

Nenhuma análise conclusiva pôde ser feita quanto à qualidade de serviço. Isso pelo fato de não termos um ambiente preparado especialmente para esses testes. Contudo, dos players que utilizamos o que apresentou melhor comportamento foi o browser do XHTML. O pior resultado ocorreu nas implementações dos players do SMIL, que uma hora davam suporte a sincronização e não a interatividade, e em outros casos o contrário.

Analisando a facilidade de uso, a linguagem que aparentemente mais se destaca é o SMIL com suas composições com semântica. Contudo para documentos complexos a melhor experiência é com o NCL. O SMIL ainda precisa melhorar na questão de composições com semânticas não convencionais, enquanto o NCL precisa aperfeiçoar o processo de criação de conectores. O XHTML, por sua vez, é o mais difícil para usuários inexperientes, dado que toda sincronização tem que ser feita utilizando-se scripts.

Dentre as linguagens analisadas, o SMIL é a que dá maior suporte a efeitos visuais. E não só dá suporte, como o uso desses é extremamente fácil. O XHTML dá menos suporte, mas praticamente tudo que é feito no SMIL pode ser obtido via

script. Já o NCL precisa começar dar mais foco a esse tipo de recurso, como transparência e sobreposição, visto que grande parte dos autores gosta de lançar mão desses efeitos.

O SMIL e o NCL dão um bom suporte ao autor. Isso em seus ambientes de autoria, como o GR*i*NS e o Maestro, respectivamente. Já para o XHTML poderia ser pensado um ambiente de autoria que possibilitasse a edição em múltiplas visões como nos ambientes anteriores. Um problema que ainda persiste fica por conta da criação de scripts, mas um framework como o XiTV pode abstrair tal necessidade do autor.

A maneira mais fácil de se implementar eventos interativos ocorreu no NCL, através de seus elos/conectores. O SMIL também não fica para trás. Já no XHTML existe a necessidade de se conhecer um pouco de programação de scripts, mas também não é complicado.

Quando o assunto é suporte a usuários iniciantes e usuários experientes, o NCL desponta na frente, principalmente agora com a incorporação de script Lua na sua arquitetura. Podemos considerar que o SMIL atende melhor aos usuários menos experientes, enquanto o XHTML está no outro extremo, tendendo para o lado dos usuários mais avançados.

Como discutido no capítulo anterior, o NCL possui potencialidades enormes. Entretanto, para que essa linguagem vingue, é necessário que toda sua complexidade não fique transparente para os autores, os quais na maioria das vezes, não querem conhecer a linguagem, mas sim executar suas idéias. O editor Maestro tem um papel de suma importância nesse processo. Um ponto que deve ser amplamente trabalhado é a criação de conectores, que ainda hoje é uma tarefa árdua e nessa ferramenta não é dado nenhum suporte a isso. Outro ponto é a questão da estruturação da apresentação da apresentação através do paradigma de *timeline*. Antes de se mudar abruptamente a cabeça de quem produz programas para que eles utilizem um novo paradigma, deve-se trabalhar para que esse processo seja gradativo. Para grande parte dessas pessoas é muito mais intuitivo dispor as mídias na escala do tempo. Claro que quando se trata de interatividade, isso não é possível, mas deve-se dar suporte a esse tipo de estruturação até onde for possível. Nesse contexto, a visão temporal pode desempenhar papel determinante. Por fim, o ambiente de autoria Maestro deve se preocupar um pouco mais com o autor e suas aspirações, além de começar a disponibilizar

recursos presentes na maioria das ferramentas comerciais existentes, para que o processo de migração se dê da forma mais natural possível.

6 Referências Bibliográficas

[BuRu04] BULTERMAN, D.; RUTLEDGE, L. **SMIL 2.0: Interactive Multimedia for Web and Mobile Devices**. Springer, Abril de 2004.

[GR*i*NS/Pro] **GR*i*NS Pro Editor for SMIL 2.0** Disponível em http://www.oratrix.com/GRiNS/

[Lua] **Lua - The Programming Language** - Disponível em http://www.lua.org/

[MuSS03] MUCHALUAT-SAADE, D.C., SILVA, H.V.O, SOARES, L.F.G. **Linguagem NCL versão 2.0 para Autoria Declarativa de Documentos Hipermídia**, IX Simpósio Brasileiro de Sistemas Multimídia e WEB - WebMídia 2003, Salvador, Brasil, Novembro de 2003.

[PAAÇ02] PEMBERTON, S.; AUSTIN, D.; AXELSSON, J.; et al. **XHTML 1.0 The Extensible HyperText Markup Language (Second Edition)**, 2002. Disponível em http://www.microsoft.com/windows/ie/. Acesso em 16 nov. 04. Disponível em http://www.w3.org/TR/xhtml1/. Acesso em 16 nov. 04.

[RoGui05] Rodrigues, Rafael Ferreira; Guimarães, Rodrigo Laiola **XiTV - XHTML iTV Framework** – Trabalho desenvolvido durante a disciplina de Fundamentos de Sistemas Multimídia. Disponível em http://sourceforge.net/projects/xitv

[SoCR04] Soares, L. F. G., Colcher, S., Rodrigues, R., **"Relatório TV Digital: Análise das Alternativas Tecnológicas"**, Laboratório Telemídia, PUC-Rio, 2004.

[SoRM03] SOARES, L.F.G., RODRIGUES, R.F., MUCHALUAT-SAADE, D.C. **Modelo de Contextos Aninhados – versão 3.0**, Relatório Técnico, Laboratório TeleMídia, Departamento de Informática, PUC-Rio, 2003.

[W3C01b] W3C - World-Wide Web Consortium. **Synchronized Multimedia Integration Language (SMIL 2.0) Specification**, W3C Recommendation, Agosto de 2001. Disponível em http://www.w3.org/TR/smil20/. Acesso em 28 out.. 04.

[W3C04a] W3C - World-Wide Web Consortium. **Extensible Markup Language (XML) 1.1**, fevereiro de 2004. Disponível em

http://www.w3.org/TR/2004/REC-xml11-20040204/. Acesso em 05 nov. 04. [W3C04b] W3C - World-Wide Web Consortium. Cascading Style Sheets Home Page, 2004. Disponível em http://www.w3.org/Style/CSS/. Acesso em 08 nov. 04.

[W3C99a] W3C - World-Wide Web Consortium. **HTML 4.01 Specification**. W3C Recommendation, dezembro de 1999. Disponível em http://www.w3.org/TR/html401/. Acesso em 07 nov. 2004.