5
Conclusões

O objetivo desta dissertação foi apresentar o ambiente de autoria *Composer*, o qual é voltado para a criação de programas NCL, versão 3.0, para TV digital interativa. Da mesma forma que no editor HyperProp, no qual foi baseado, no *Composer* as abstrações são definidas nos diversos tipos de visões que permitem simular um tipo específico de edição. Conforme foi apresentado neste documento, essas visões funcionam de maneira sincronizada, a fim de oferecer um ambiente integrado de autoria.

A metodologia empregada para a execução deste trabalho envolveu uma ampla análise da literatura em ambientes de autoria (Capítulo 2), a proposta de melhorias (Capítulo 3), principalmente para a arquitetura da visão gráfica temporal, bem como sua implementação (Capítulo 4).

Como foco principal, a criação e manipulação de objetos de mídia dispostos na visão temporal foram contempladas. A criação de objetos de mídia no tempo e seus relacionamentos com os demais objetos foram tratados de forma a tornar a especificação da apresentação mais intuitiva para o autor. Problemas de representação e edição de objetos de mídia, relacionamentos entre objetos, relacionamentos interativos e edição ao vivo também foram tratados. De forma geral, o *Composer* integra técnicas a fim de facilitar o processo de edição de documentos NCL para TV digital interativa.

Na próxima subseção é feita uma análise comparativa dos recursos, principalmente os de sincronização temporal e edição ao vivo, entre as ferramentas apresentadas no Capítulo 2 e o *Composer*, levando em conta as características de cada uma, além de serem sumarizadas as principais melhorias do *Composer* com relação ao editor HyperProp. Por fim, são discutidos pontos que ficam em aberto para serem tratados em trabalhos futuros, visando tornar o ambiente de autoria *Composer* mais atrativo para profissionais de TV digital.

5.1. Análise Comparativa

Os ambientes JAME Author, Cardinal Studio, AltiComposer e Macromedia Flash dão um maior suporte ao autor do ponto de vista de abstração com relação ao modelo e linguagem utilizados. Através de seus modelos de autoria e interfaces gráficas, é mais simples pensar nas abstrações de botão, menu etc. Nesses ambientes, o processo de autoria se caracteriza principalmente pela especificação da disposição espacial das mídias do documento, dando ao autor uma noção de como os objetos serão apresentados na tela.

Contudo, esses ambientes deixam a desejar quanto à definição do sincronismo entre mídias. O JAME Author, por exemplo, não possui uma visão temporal e não oferece nenhum suporte à definição do sincronismo temporal. O Cardinal Studio e o AltiComposer demandam conhecimentos avançados para configuração do sincronismo através da personalização do comportamento de componentes. Ademais, O Cardinal Studio possui uma visão temporal muito simples, enquanto o AltiComposer nem possui uma. Nesse contexto, o Macromedia Flash, apesar de só permitir a definição de relacionamentos através da utilização de scripts, lança mão de sua visão temporal (*timeline*) em conjunto com a visão espacial para fornecer ao autor uma poderosa ferramenta de edição.

As abstrações fornecidas pelas ferramentas de autoria GR*i*NS, LimSee2 e *Composer* são mais ligadas à linguagem hipermídia na qual esses ambientes se apóiam (GR*i*NS e LimSee2 em SMIL, e o *Composer* em NCL). No GR*i*NS e no LimSee2, o processo de autoria foca na configuração temporal dos objetos de mídia da apresentação. O LimSee2 tenta ainda utilizar uma visão espacial, que em conjunto com a visão temporal, permite a edição de relacionamentos espaço/temporais. Contudo, na versão analisada, a implementação da visão espacial não funcionava efetivamente.

O *Composer*, por sua vez, além de fornecer todos os recursos para organização lógica de documentos (através da visão estrutural), permite agora que o sincronismo temporal seja especificado e editado através da visão temporal. Além disso, O *Composer* é o único desses ambientes que permite a simulação de eventos interativos em tempo de autoria, sem a necessidade de apresentação do documento.

Quanto ao suporte a usuários com diferentes níveis de conhecimento, a ferramenta JAME Author tem como recurso mais avançado a especificação da navegação entre páginas e componentes. Os ambientes Cardinal Studio, AltiComposer e Macromedia Flash dão suporte a usuários iniciantes através de suas abstrações, e a usuários avançados através da possibilidade de personalização de componentes e seus comportamentos. Já o GR*i*NS, o LimSee2 e o *Composer* podem ser considerados ferramentas que dão suporte a usuários que conhecem bem as linguagens nas quais esses ambientes se baseiam. Ao longo deste trabalho, a interface gráfica do *Composer* foi remodelada, contudo, ainda é preciso se trabalhar em abstrações que visem tirar do usuário a necessidade de conhecer a linguagem NCL.

Ainda é importante ressaltar que nenhum dos ambientes estudados dá suporte ao monitoramento e envio de comandos de edição ao vivo, como faz o *Composer*. O Cardinal Studio e o AltiComposer permitem a utilização de *stream events*, que podem ser utilizados para tal finalidade. Não obstante, esses ambientes não fornecem um mecanismo simples e direto que possibilite ao autor enviar suas modificações em tempo de apresentação de um documento. Nesse quesito, o *Composer* é único, devido a todas as características discutidas (Seção 3.3 e 4.4).

Um ponto positivo é que todos os ambientes analisados possuem um emulador associado à ferramenta de autoria, e isso permite que o autor teste sua aplicação sem a necessidade de um *set-top box*, evitando que problemas sejam percebidos tardiamente.

Com relação ao editor HyperProp, primeiramente cabe destacar que essa ferramenta é voltada para a autoria de aplicações hipermídia em NCL *main profile* (versão 2.3), enquanto o *Composer* tem por objetivo dar suporte à criação de programas NCL versão 3.0 (perfis de TV digital).

O *Composer* teve grande parte da sua interface gráfica remodelada a fim de tornar essa ferramenta mais atrativa para o autor. Nesse processo, foram desenvolvidos os *Enhanced Items* e o módulo de localização, os quais facilitam o reuso e a manutenção dos itens de interface. Visando suprir ainda outras carências do editor HyperProp, foram desenvolvidos o módulo de ajuda, de mensagens, de registro, e o módulo de preferências.

O *Composer* é baseado praticamente na mesma arquitetura de visões do HyperProp. Contudo, todas as visões foram remodeladas, a fim de melhorar o

processo de edição de hiperdocumentos. A visão espacial que no HyperProp só permite a especificação de leiautes, no *Composer* passou a se chamar visão de leiaute. Já a nova visão temporal permite a criação e manipulação de objetos de mídia no tempo, bem como a especificação de relacionamentos entre eles, diferente do que acontece no HyperProp. A visão temporal do *Composer* ainda dá suporte à simulação de eventos interativos, permitindo que o autor visualize como se dará a apresentação em tempo de criação. No editor HyperProp, a visão temporal só permite a visualização de como os objetos de mídia estarão dispostos no tempo. Por fim, a edição ao vivo, requisito importante no contexto de TV, é mais uma novidade presente somente no *Composer*.

5.2. Trabalhos Futuros

Alguns pontos do trabalho realizado nesta dissertação podem ser explorados como trabalhos futuros.

Em determinadas situações, a exibição de composições com semânticas temporais pode ajudar o autor na especificação dos relacionamentos entre os objetos de mídia. Tal funcionalidade enriqueceria a visão temporal, tornando-a mais atraente no desenvolvimento de documentos NCL através do *Composer*. Para isso, *templates* podem ser utilizados, por exemplo, para obter a semântica das composições SMIL, que possuem semântica paralela, seqüencial ou exclusiva.

Apresentações um pouco mais complexas, como por exemplo, com muitos elos de sincronismo, não são triviais de se elaborar. Elas, geralmente, demandam a criação de conectores, o que requer do autor um conhecimento mais profundo da linguagem. No *Composer* ainda não é dado suporte a esse tipo de problema.

Os mecanismos de filtragem baseados na técnica olho-de-peixe, conforme descritos em (Coelho & Soares, 2004), precisam ser reimplementados em todas as visões do *Composer*. Uma alternativa interessante seria implementar as visões se baseando na abordagem ZUI (*Zoomable User Interfaces*). No caso da visão temporal, os mecanismos de filtragem nunca existiram, e são muitos os problemas que precisam ser tratados. Dentre eles, pode-se destacar a utilização de *templates* e a exibição de composições como múltiplas cadeias.

Ademais, seria importante a especificação de uma API que permitisse que novas visões fossem adicionadas ao *Composer*, tendo como base o modelo Java NCM ou a árvore DOM que representa um documento, mantendo assim as características de edição e sincronismo. Isso permitiria que o *Composer* fosse personalizado de acordo com as necessidades do autor.

Atualmente, projetos do *Composer* são salvos como objetos serializados. Seria interessante que esses projetos fossem salvos como arquivos XML para tornar mais fácil a compatibilidade de projetos entre versões do ambiente de autoria.

Um ponto que ainda precisa ser tratado é o suporte à criação de outras entidades da NCL através da interface gráfica, como por exemplo, os nós de *switch*. Além disso, devem ser implementados mecanismos de suporte a ações de desfazer (*undo*) e refazer (*redo*), e melhorado o verificador de documentos NCL, o qual, na atual implementação, não tem detectado todos os erros de compilação. Para isso, uma alternativa interessante é o uso de Schematron (ISO, 2006). Sugere-se ainda o estudo de alternativas para contemplar a especificação da navegação, de transições e de animação através da interface do *Composer*.

Na visão temporal, no caso de um objeto de mídia contínua, seria interessante exibir sua duração natural mesmo que o objeto termine antes ou depois desse tempo. Além disso, na atual implementação do modelo de cadeias temporais hipermídia não é dado suporte a documentos com ciclos (*loops*). Alternativas para se contornar esse problema devem ser estudadas.

Para testar sua autoria, o autor pode querer iniciar a apresentação a partir de um determinado instante, especificado pela barra de tempo da visão temporal. É esperado que além de iniciar o documento a partir de qualquer instante, seja possível o disparo de eventos interativos anteriores ou posteriores a esse ponto, para verificação de como se dará o restante da apresentação. Para isso, um exibidor simulado (formatador) que permita tal funcionamento deve ser implementado e integrado à ferramenta de autoria. A utilização do modelo de cadeias hipermídia será de grande valia nesse processo, uma vez que para apresentar um documento a partir de um determinado instante será necessário caminhar no grafo definido pelo modelo, simulando a apresentação de entidades e a ocorrência de eventos até tempo especificado.

A visão espacial, anteriormente definida em (Costa & Soares, 1996), (Moura & Soares, 2001) e (Coelho & Soares, 2004), precisa ser implementada no *Composer* para dar suporte à especificação e edição do sincronismo espacial em documentos NCL. Essa visão permitirá ao autor visualizar graficamente como os objetos de mídia estarão dispostos no espaço em um dado instante da apresentação do documento. Trabalhando de forma coordenada, a visão espacial e a visão temporal podem ser úteis para criação e edição de relacionamentos espaço/temporais. A Figura 32 ilustra o funcionamento da visão espacial.

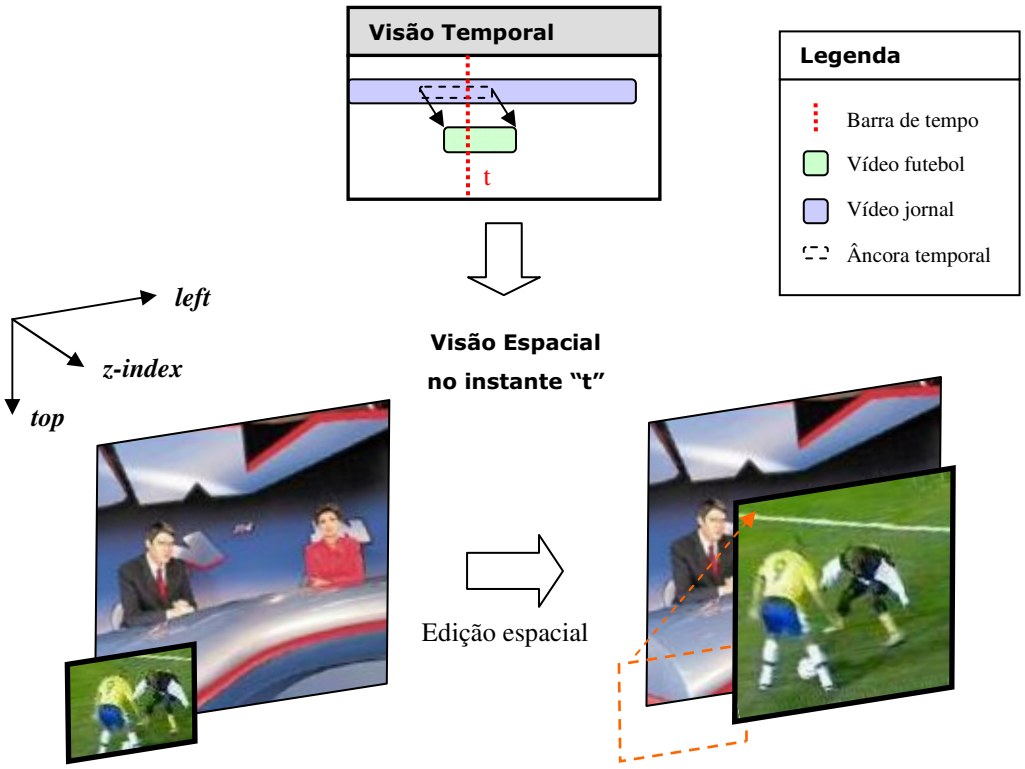

Figura 32 - Edição de relacionamentos espaciais no *Composer*

Na figura acima, a edição do posicionamento e das dimensões (da região) de um objeto de mídia, como por exemplo, o vídeo do futebol, deve ser refletida no documento NCL. Para que isso ocorra, o *Composer* deve ser capaz de transformar as ações do autor em relacionamentos espaciais.

A visão espacial deve ainda permitir a visualização e edição do volume do áudio de objetos de mídia, conforme proposto em (Costa & Soares, 1996). Nesse caso, barras indicariam os percentuais do volume usado na reprodução dos objetos que estão sendo apresentados em um determinado instante de tempo. Da mesma forma que no caso da edição espacial, a alteração do volume do áudio também deve ser refletida no documento.

Para concluir, a realização de testes de usabilidade permitirão verificar como o sistema produzido contribui para apoiar as necessidades inerentes ao processo de autoria de documentos hipermídia. Isso será muito importante para rastrear *bugs* e para que novas funcionalidades possam ser desenvolvidas para que o ambiente de autoria *Composer* se adapte às necessidades dos profissionais de TV digital.

O código-fonte do *Composer* foi publicado no portal do software público brasileiro (http://www.softwarepublico.gov.br) e, portanto, pode receber contribuições da comunidade de software livre para a sua evolução.