6
Conclusões e Trabalhos Futuros

Neste capítulo descrevemos as conclusões obtidas a partir desta dissertação e relatamos todos os possíveis trabalhos futuros que podem ser realizados.

6.1
Conclusões

A escassez de ferramentas de autoria para o Sistema Brasileiro de TV Digital, mais especificamente para a NCL, tem limitado a criação de aplicações interativas por profissionais de mídia. Atualmente, é necessário um aprendizado inicial em NCL para poder participar do processo de autoria.

Devido a essa escassez, este trabalho procurou na literatura ferramentas de autoria de documentos hipermídia e/ou multimídia para outros sistemas, como o MHP (modelo europeu de TV Digital). Essas ferramentas foram analisadas e passaram a ser a base da ferramenta proposta por este trabalho.

Sendo assim, propomos uma ferramenta de autoria, a NCLite, que abstrai a NCL para agilizar e facilitar o início da autoria. Essa abstração é obtida através de componentes gráficos que suportam a manipulação direta por parte do autor.

Com o intuito de validar os conceitos inseridos na NCLite antes mesmo da implementação, um estudo qualitativo com possíveis usuários da ferramenta foi realizado. Esse estudo nos mostrou que o uso de cenas interativas e da sincronização espaço-temporal foi adequado, mas que precisava de algumas adaptações, as quais foram realizadas ou indicadas caso estivessem fora do escopo deste trabalho.

Partimos então para a implementação da NCLite. Como o projeto contempla uma ferramenta de autoria completa, não foi possível implementar todas as funcionalidades idealizadas. Dessa forma, capturamos somente as funcionalidades que possibilitariam a avaliação dos conceitos principais, evitando despender esforço e tempo de programação em conceitos que não foram adequadamente validados. A Tabela 6.1 sumariza os elementos do projeto da NCLite indicando quais foram implementados.

		Escopo	
		Implementação	Projeto
Funcionamento	NCL usual	+	+
	Esqueleto NCL	−	+
	Ideia	−	+
Componentes	Interatividade visual	+	+
	Visão espacial	+	+
	Linha do tempo	+	+
	Templates	−	+
	Propriedades	+	+
	Widgets Interativos	−	+
	Biblioteca de mídias	+	+

Tabela 6.1: "−" não contempla e "+" contempla completamente a funcionalidade.

Com a implementação atual da NCLite, já conseguimos carregar uma aplicação NCL convertendo-a para o modelo de autoria da ferramenta. Além disso, realizamos alterações no conteúdo e nas características espaço-temporais da aplicação NCL carregada, como apresentado na Seção 5.3.

Vale destacar que a NCLite foi desenvolvida em módulos e utilizou o *plugin* EMF para criar uma arquitetura fácil de ser estendida por trabalhos futuros. O maior obstáculo para continuar o desenvolvimento é a curva de aprendizado elevada para implementação de plugins no *Eclipse-RCP*.

6.2 Trabalhos Futuros

É possível listar as seguintes atividades que podem ser realizadas para continuar o trabalho aqui apresentado:

1. realização de uma avaliação somativa;
2. criação de ajuda contextual, tutoriais e vídeos explicativos;
3. utilização do perfil estendido da NCL;
4. integração com Composer 3;
5. integração com NCL Eclipse;
6. adaptação do conceito de cenas;
7. implementação das funcionalidades relativas às cenas mestres, ao esqueleto NCL e à inserção de novos elementos;
8. implementação da edição ao vivo;
9. implementação de melhorias na interface gráfica;

10. implementação de melhorias na conversão do NCL para o modelo de autoria da NCLite.

Para que seja possível entender melhor o que cada item desse representa, detalhamos todos eles a seguir.

6.2.1 Avaliação Somativa

A avaliação somativa (Barbosa & Silva, 2010) é realizada nas etapas finais do desenvolvimento quando o *software* está praticamente concluído. Útil para identificar problemas de interação e interface que podem ser corrigidos de forma a aumentar a qualidade do sistema.

Para validar os conceitos incorporados na implementação da ferramenta, é preciso realizar um novo estudo qualitativo com usuários. A intenção desse estudo é revalidar os conceitos encontrados no projeto da NCLite. Caso o estudo descubra inconsistências na implementação, será necessário corrigir também o projeto para que ambos caminhem em conjunto.

Seria interessante que esse novo estudo fosse realizado antes de continuar a implementação da NCLite. Isso porque consideramos que a melhor abordagem é: (1) implementar uma gama de funcionalidades suficientes para termos uma ferramenta usável; (2) realizar um estudo com usuários para validar a implementação. Essa abordagem provoca um desenvolvimento com ênfase nos usuários e que ajuda a encontrar e corrigir os problemas o mais cedo possível.

6.2.2 Ajuda

Conforme verificado no Capítulo 4, a NCLite precisa incorporar componentes de ajuda contextual, de tutoriais e de vídeos explicativos. Entretanto, nenhum desses componentes foi incorporado na versão atual da NCLite porque o foco desse trabalho foi definir os conceitos da interface e implementar um exemplo para avaliá-los.

Para criar todo o material de ajuda, é preciso buscar profissionais de documentação técnica. Além disso, para os vídeos, é necessário envolver também profissionais de produção de mídias. E, para a ajuda contextual, profissionais de IHC e engenharia de software devem colaborar para oferecer acesso adequado ao conteúdo de ajuda.

Também não podemos esquecer da necessidade de criação da ajuda de conteúdo (*help*) que é comumente encontrada em ferramentas de autoria.

6.2.3 Perfil estendido da NCL

Inicialmente, a intenção era suportar ambos os perfis da NCL na NCLite. Como apresentado no Capítulo 5, o parser utilizado pela NCLite foi gerado automaticamente pelo *XMLBeans*, através do *XML Schema* da linguagem. Entretanto, alguns problemas ocorreram durante esse processo:

1. o *XMLBeans* não conseguiu gerar o parser para o *XML Schema* no NCL. O gerador de código do *XMLBeans* parecia se perder em meio a grande utilização do `<xsd:substitutionGroup>`, por parte do *XML Schema* da NCL. Isso aconteceu também com outros geradores de parsers XML para Java;

2. o *XML Schema* possuía alguns erros e discrepâncias com o texto da *Norma ABNT* da NCL;

3. o *XML Schema* possuía erros na referência ao SMIL (SMIL 3.0, 2008).

Para progredir com o trabalho, pegamos o *XML Schema* do perfil básico da NCL e o reescrevemos de forma a retirar os `<xsd:substitutionGroup>`. Com essa modificação, o *XMLBeans* conseguiu fazer a geração de código corretamente. Toda a implementação da NCLite se baseou no *parser* gerado a partir do perfil básico da NCL.

Antes de finalizarmos a escrita desta dissertação, verificamos que os problemas descritos acima foram solucionados. Entretanto, não geramos o novo *parser* a partir do perfil estendido corrigido. Sendo assim, é preciso gerar esse parser antes de continuar a implementação da NCLite.

6.2.4 Integração com Composer 3

Atualmente, encontra-se em fase de desenvolvimento a versão 3 do Composer (Soares & Lima, 2010). Essa nova versão visa atender aos requisitos não funcionais, tais como confiabilidade, manutenibilidade e extensibilidade, que foram deixados de lado nas versões anteriores.

Dessa forma, o Composer 3 consiste de um micro-núcleo que recebe notificações de modificações na aplicação NCL, altera o modelo único que representa essa aplicação e emite notificações da mudança para as demais visões. Além disso, pensando na extensibilidade, o Composer 3 permitirá que as visões presentes no seu antecessor sejam vistas como *plugins* e com isso não sejam fortemente acopladas ao micro-núcleo. A intenção também é que outros

desenvolvedores possam criar novas visões mais facilmente de acordo com as suas necessidades.

Pensando nisso, uma possível maneira de continuar a NCLite é adaptá-la para utilizar o Composer 3 em substituição à camada de conversão (Seção 5.2.1). O módulo do parser (Seção 5.2.1) seria totalmente substituído e o de persistência (Seção 5.2.1) teria que sofrer alterações na arquitetura.

Ao utilizar o Composer 3, a NCLite passaria a ser um **plugin** com um modelo próprio e com diferentes visões que se relacionariam a partir do seu modelo de autoria. O Composer funcionaria exatamente como a camada de conversão atual.

Entretanto, há uma grande dificuldade nessa adaptação. O Composer 3 está sendo desenvolvido em QT/C++ e a NCLite foi implementada em Java. Dessa forma, é preciso portar todo o código fonte para QT/C++.

Vale destacar que, se a integração com o Composer 3 for realizada, não é preciso incorporar o suporte ao perfil estendido no *parser* da NCLite, como sugerido na Seção 6.2.3. Isso porque o Composer já será desenvolvido com suporte ao perfil estendido.

6.2.5 Integração com NCL Eclipse

Como o NCL Eclipse (NCLEclipse, 2008) também foi desenvolvido como um *plugin* para o *Eclipse*, é possível integrá-lo à NCLite com um reduzido esforço de implementação. A inclusão do NCL Eclipse faz com que a NCLite também contemple o usuário programador, que atualmente é deixado em segundo plano.

Entretanto, para que isso seja possível, é preciso adaptar o plugin do NCL Eclipse para ser executado diretamente na NCLite, como parte de uma aplicação autônoma ao *Eclipse*. Além disso, temos também que alterar os conversores do modelo de autoria da NCLite de maneira que a estrutura do código NCL gerado seja idêntica à inicialmente carregada. Isso é importante porque os usuários do NCL Eclipse manipulam diretamente o código fonte da aplicação NCL e mudanças drásticas podem provocar estranheza e/ou empecilho para utilização da ferramenta.

6.2.6 Adaptação das cenas

Verificamos na Seção 5.3 que a NCLite agiliza e facilita a autoria de aplicações NCL que, quando representadas no grafo temporal hipermídia, são apresentadas como uma árvore e não um grafo.

A NCLite ainda não foi avaliada para as demais famílias de aplicações NCL. Sendo assim, é preciso avaliá-la para validar o conceito de cena aqui apresentado e, se preciso, adaptá-lo para englobar todos os demais tipos.

6.2.7 Cenas mestres, Esqueleto NCL e Inserção de elementos

Como apresentado no Capítulo 3, o projeto da NCLite prevê a inclusão de conceitos como as cenas mestres e o esqueleto NCL na ferramenta. Além disso, são previstos também todos os componentes gráficos para a inserção de novos elementos a partir de uma ideia. Entretanto, a implementação atual da NCLite não conseguiu contemplar essas funcionalidades. Por isso, sugerimos a inclusão delas em uma nova versão da NCLite.

Consideramos também que é necessária uma análise antes de continuar o desenvolvimento para saber qual é a melhor política para implementação das funcionalidades acima.

6.2.8 Edição ao vivo

Atualmente, a NCLite não contempla a edição ao vivo de aplicações NCL. A intenção é utilizar os conceitos apresentados e adaptá-la para que seja possivel realizar esse tipo de edição no futuro.

É preciso rever todo o módulo de conversão da aplicação NCL para o modelo de autoria e inserir novos módulos exclusivos da edição ao vivo.

6.2.9 Melhorias na interface gráfica

A implementação atual da NCLite precisa de algumas melhorias nos componentes gráficos. Algumas dessas melhorias já foram identificadas e são descritas a seguir separadas por componente:

- **Módulo Componentes auxiliares - Propriedades**

  1. separação em abas de acordo com o tipo do objeto selecionado: Espaço, Tempo, Audio e Navegacao (Menu). Precisa criar classes `Section` e utilizar a classe `TabbedPropertySheetPage`;
  2. exibição correta dos valores temporais da classe `Time`. Precisa criar um property descriptor para representar essa classe;
  3. exibição do valor correto e não `ValueWrapper` quando é realizada a edição das propriedades.

– **Módulo Componentes auxiliares - Biblioteca de mídias**

1. inclusão das formas de exibição em árvore e com prévia da mídia.

– **Módulo Espacial**

1. tratamento da exibição de um componente SWT em um canvas do GEF. Isso faz com que o html dos elementos textuais não seja exibido corretamente;
2. tratamento do caso que o tamanho e a posição são dados em porcentagem. Atualmente lemos as porcentagens e as convertemos para pixels, mas ainda é preciso oferecer a possibilidade do autor escolher a forma de edição: pixel ou porcentagem;
3. tratamento dos diferentes valores de *ZIndex* que os recursos podem assumir. Pesquisar no GEF / Draw2d como é o tratamento da ordem que os objetos `Figures` sao pintadas na tela.

– **Módulo Linha do Tempo**

1. tratamento de recursos com loops: repetir o intervalo do recurso até a maior mídia e usar um ícone de seta retornando ao mesmo intervalo;
2. tratamento de recursos infinitos: repetir o intervalo até a maior mídia e depois colocar um ícone de infinito ou reticências;
3. inserção de algumas limitações na edição para se adequar ao que é possível ser feito na NCL;
4. criação de uma comunicação temporal para o relacionamento causal entre os recursos.

– **Módulo Interatividade - Mídias interativas**

1. inclusão das formas de exibição em árvore e com prévia da mídia.

– **Módulo Interatividade - Cenas interativas**

1. fixar a seleção da cena no grafo de cenas como a cena corrente;
2. posicionar o nome do recurso interativo no meio da aresta do grafo de cenas;
3. deixar fixo o tamanho dos nós que representam as cenas no grafo de cenas.

6.2.10 Melhorias na conversão

A implementação atual da NCLite precisa de algumas melhorias na conversão entre o modelo de autoria da ferramenta e a aplicação NCL. Algumas dessas melhorias foram identificadas e são descritas a seguir:

1. tratar nós de mídias dos tipos: `Audio`, `Settings`, `Lua`, `Java`, `NCL` e `Time`;
2. tratar contextos tanto no arquivo de entrada NCL quanto no de saída de forma a utilizar o reúso oferecido pela NCL;
3. tratar mais de um arquivo NCL como entrada e `import` em um arquivo NCL;
4. tratar elementos `CausalConnector`. Atualmente, utilizamos somente os elementos `Link` para contemplar o sincronismo espaço/temporal e a interatividade. Talvez seja melhor verificar o `CausalConnector` que o `Link` utiliza;
5. tratar várias portas na aplicação NCL;
6. tratar `CompoundCondition` no elemento `Link`;
7. tratar o elemento `Switch` da NCL;