6
Composer II

Diversos são os perfis de autores que estão direta ou indiretamente ligados com o processo de criação, edição e transmissão de aplicações interativas na TVDi. Cada um desses perfis possui especialidades, necessidades e expectativas diferentes. Além disso, tais autores também podem estar em ambientes com capacidades diversas (radiodifusor, ambiente de autoria, ambiente doméstico).

A fim de dar suporte a esses diferentes perfis de autores de uma ferramenta de autoria, a nova versão do Composer é baseada no conceito de múltiplas visões. O Composer II foi construído sobre a arquitetura apresentada no Capítulo 4, servindo como prova de conceito para os requisitos que está arquitetura suporta.

No Composer II, cada visão realça uma particularidade do documento e cria uma perspectiva que abstrai características da linguagem, NCL no caso. Sendo assim, cada visão pode ser mais apropriada para um determinado perfil de autor, permitindo a manipulação do documento sem a necessidade de conhecimentos técnicos profundos sobre NCL.

No caso do Composer II, uma visão é sempre vista como um plug-in, mas um plug-in não necessariamente é uma visão sobre o documento. Plug-ins relacionados a transmissão e edição em tempo real não criam uma visão sobre o documento e não dão a opção de manipulação através de suas interfaces. Um plug-in pode ser ainda um conjunto de visões que se comunicam entre si.

PUC-Rio - Certificação Digital Nº 0821366/CA

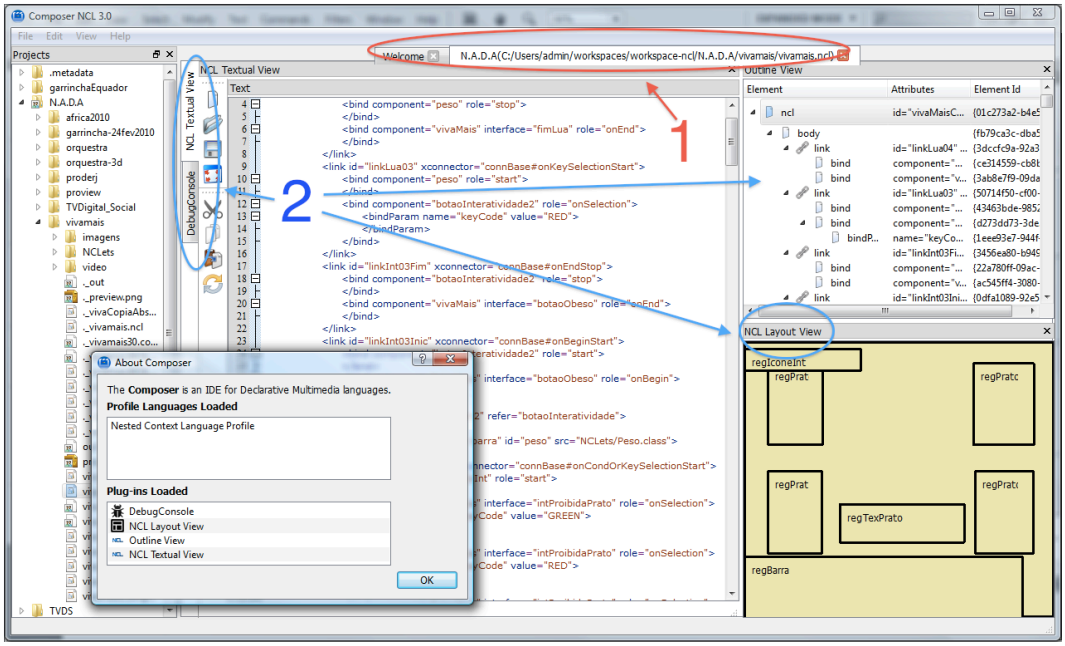

Figura 19 - Composer II rodando no Windows

A Figura 19 apresenta a interface gráfica do Composer II sendo excutada no Windows. A área marcada em vermelho (**1**) traz as abas com os documentos NCL que estão sendo editados no momento. Para cada documento presente na aba, os plug-ins carregados no micro-núcleo são lançados e apresentados na tela. A área marcada em azul (**2**) apresenta os plug-ins carregados no micro-núcleo, nesse caso: plug-in textual, plug-in de leiaute, plug-in de debug e um plug-in de *outline*.

Um dos objetivos dessa arquitetura é dar suporte a multiplataforma, as Figura 19, Figura 20 e Figura 21, mostram o Composer II rodando nos principais sistemas atuais, Windows, Mac OS e Linux. Em todos esses sistemas, o código-fonte tanto do micro-núcleo, quanto do Composer II e dos plug-ins foram obtidos e compilados da mesma fonte, ou seja, é o mesmo.

PUC-Rio - Certificação Digital Nº 0821366/CA

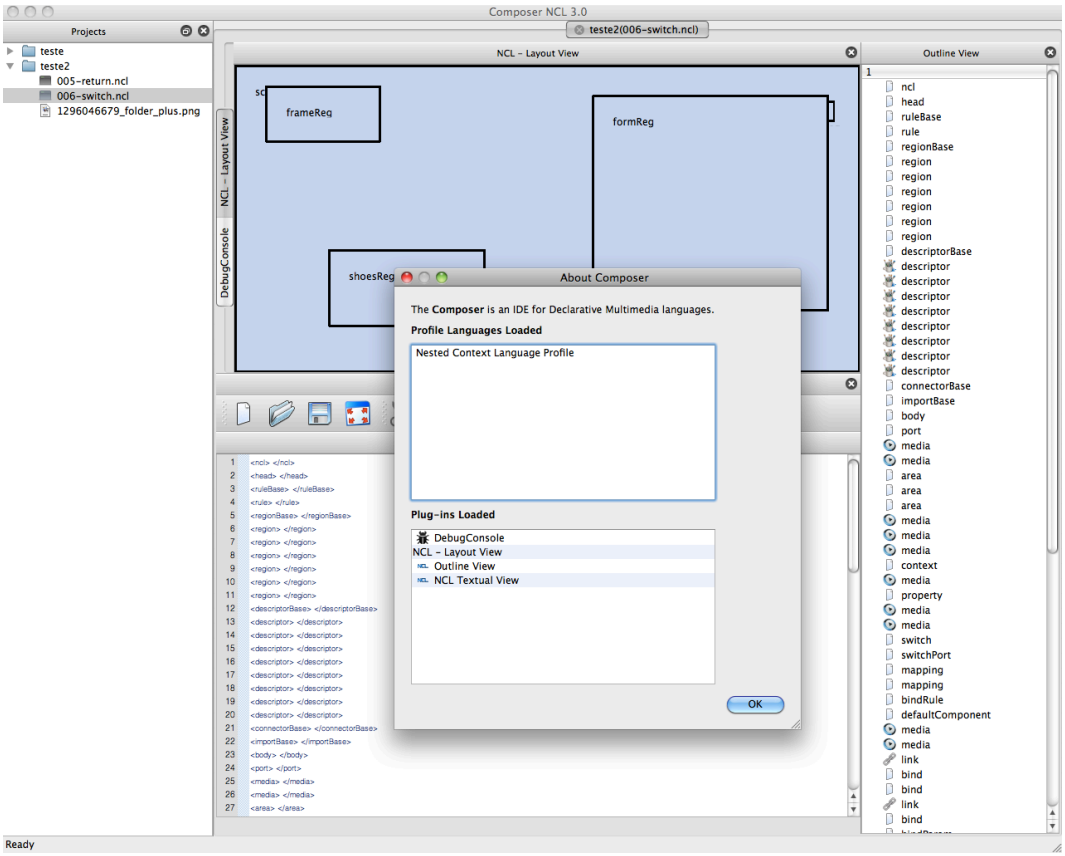

Figura 20 - Composer II rodando no Mac OS

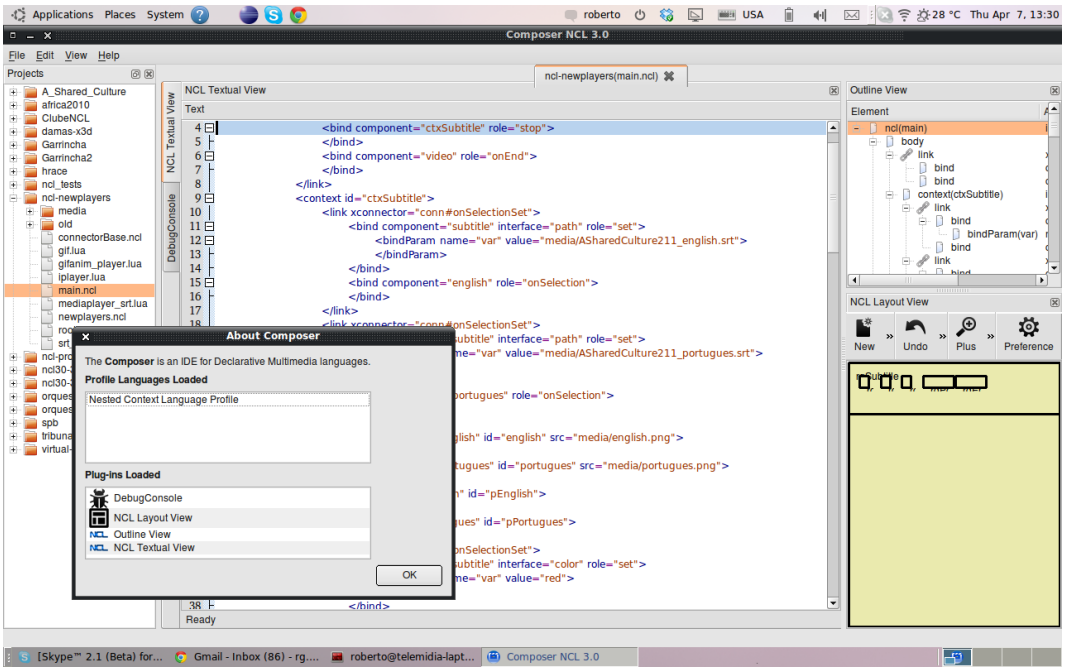

Figura 21 - Composer II rodando no Linux

PUC-Rio - Certificação Digital Nº 0821366/CA

6.1. Integração com o micro-núcleo

O Capítulo 5 apresentou os aspectos de implementação do micro-núcleo e como suas classes colaboravam para realizar as tarefas requisitadas pelo autor. Todos os detalhes discutidos nesse capítulo abstraíam qualquer tipo de interface com o autor, referenciado na Figura 12 como *ToolGUI*. Este capítulo discute como ocorre a integração entre a interface gráfica com o autor e as funcionalidades do micro-núcleo.

É importante ressaltar que as funcionalidades presentes no micro-núcleo são independentes da existência de uma interface gráfica. Para isso, foi desenvolvida uma API para qual a *ToolGUI* pode fazer chamadas requisitando funcionalidades presentes no micro-núcleo. Essa interface é chamada de *CoreManager*.

A **Listagem 3**, apresenta a API do *CoreManager*. Essa API contêm chamadas para funcionalidades relacionadas aos plug-ins (**1-6**), assim como funções para manipulação de documentos (**7-8**) e a manutenção de sessões de edições (**9-12**).

Ao iniciar o Composer II, a função *restorePreviousSession()* é chamada com o objetivo de restaurar uma sessão anterior, caso exista. Dentre os parâmetros que serão restaurados está o caminho, no sistema de arquivos, em que os plug-ins estão localizados. Esse caminho é dado como parâmetro para função *initPlugins()* que irá carregar os plug-ins presentes no diretório. Essa função pode ser chamada pelo autor em qualquer momento da sessão. Caso um novo plug-in tenha sido instalado no diretório, essa chamada irá carregar esse novo plug-in e sincronizá-lo com os demais.

PUC-Rio - Certificação Digital Nº 0821366/CA

```
1: QWidget* getPluginInstanceWidget(IPlugin *);
2: QList<IPlugin*> getLoadedPlugins();
3: QList<IPlugin*> initPlugins(QString path);
4: void releaseLoadedPlugins();
5: void releasePlugin(IPlugin *);
6: IPlugin* reloadPlugin(IPlugin *);
7: Document* openDocument(QString path);
8: void closeDocument(Document *);
9: void saveDocument(Document *);
10: void restorePreviousSession (QString wsPath);
11: void saveSession(QString wsPath);
12: void cleanUp();
```

Listagem 3 - API CoreManager

A API ainda prevê chamadas para manipulação individual de plug-ins, como *releasePlugin()* e *reloadPlugin()*, que descarrega ou recarrega um plug-in dentro do micro-núcleo. Caso o plug-in seja descarregado, os documentos sendo editados permanecem abertos, mas os *widgets* associados a ele em cada documento serão fechados. Além disso, estão presentes funções para manipulação dos plug-ins como um todo, como *getLoadedPlugins()* e *releaseLoadedPlugins()*. Tais chamadas requisitam os plug-ins carregados e o descarregamento de todos os plug-ins presentes no micro-núcleo.

As chamadas para manipulação de documentos preveem a sua abertura (*openDocument()*) e o seu fechamento (*closeDocument()*). Caso o documento tenha sido modificado, a função de salvar é chamada (*saveDocument()*). O autor também pode salvar o documento a qualquer momento da sessão. Finalmente, a API apresenta funções para a manutenção das sessões de edição. Os parâmetros dessas sessões podem ser de caráter geométrico, como no caso da disposição dos elementos da interface gráfica na tela, ou até mesmo parâmetros específicos de cada plug-in, como cor de texto, tamanho de fonte (plug-in textual), cor de determinadas regiões (plug-in de leiaute). As sessões são salvas dentro da área de trabalho (*workspace)* definida pelo autor no início do uso do Composer II.

A Figura 22 apresenta o diagrama das classes utilizadas na integração do Composer II com o micro-núcleo. A classe *ComposerWindow* é a janela principal de interface gráfica com o autor. Essa classe possui uma referência para o

PUC-Rio - Certificação Digital Nº 0821366/CA

CoreManager, possibilitando a comunicação com o micro-núcleo através dessa interface.

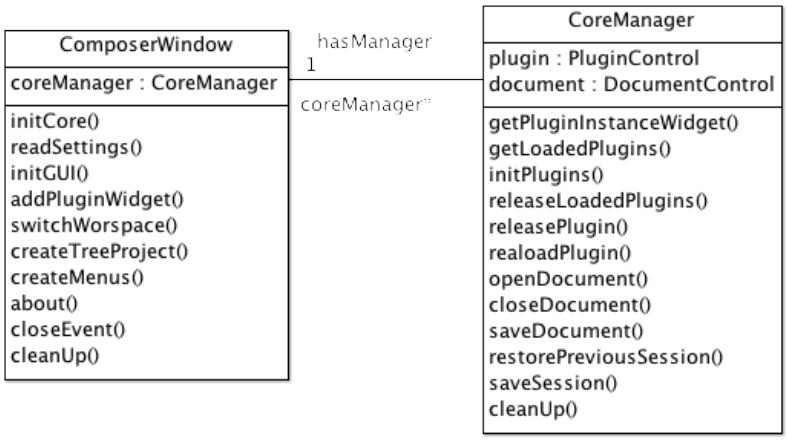

Figura 22 - Diagrama de classe da integração do Composer II com micro-núcleo

6.2.Plug-ins

Diferentes plug-ins foram e estão sendo implementados por diferentes desenvolvedores para essa nova versão do Composer. Dentre os quais pode-se destacar plug-ins voltados para autoria: textual, leiaute, estrutural, outline. O plug-in textual incorpora as principais funcionalidades encontradas no NCL Eclipse, presente no Capítulo 2. O plug-in de leiaute permite a disposição espacial inicial dos objetos de mídia presentes na aplicação. O plug-in estrutural permite a estruturação dos objetos de mídias em contextos. O plug-in de *outline* dispõe os elementos do documento NCL em forma de árvore, criando uma maneira rápida de navegação sobre o documento.

Ainda existem plug-ins em desenvolvimento voltados para autoria de família de documentos em NCL, que são especificados utilizando TAL (*Template Authoring Language*), uma linguagem de domínio especifico que provê um grande nível de reuso (Neto, 2010). Ainda sobre família de documentos, um conjunto de plug-ins está sendo desenvolvido, sob o nome de SAGGA (*Support for Automatic Generation of Ginga Applications*). O SAGGA utiliza TAL para geração rápida de documentos NCL baseados em arquétipos previamente conhecidos.

Além dos plug-ins para autoria, estão sendo desenvolvidos plug-ins voltados para transmissão do documento NCL em um sistema de transporte especifico. Um

PUC-Rio - Certificação Digital Nº 0821366/CA

desses plug-ins permite a geração do carrossel de objetos DSM-CC e seu encapsulamento em um fluxo de transporte MPEG-2. Um outro plug-in dá suporte à geração de fluxo de eventos NCL para transportar comandos de edição em tempo real.

Adicionalmente, o Composer II terá um plug-in para integração com o equipamento de transmissão (*play-out*) da EITV. Esse plug-in será capaz de conversar com o *play-out* para o envio do documento NCL e seu agendamento para transmissão um serviço de TVD. Essa integração fará do Composer II uma ferramenta completa para TVD, englobando as fases de design, autoria, armazenamento e transmissão da aplicação NCL.

PUC-Rio - Certificação Digital Nº 0821366/CA