2
Geração Dinâmica de Conteúdo e Templates de Composição

Alguns dos aspectos mais importantes na arquitetura proposta nesta dissertação são: a *geração dinâmica de conteúdo* e a *utilização de templates de composição*. Devido à importância que esses aspectos possuem neste trabalho, este capítulo tem a finalidade de investigá-los, apresentando como são abordados pelos pontos de vista de outros trabalhos do estado da arte.

Ainda que alguns dos trabalhos apresentados neste capítulo realizem geração dinâmica de conteúdo com o uso de *templates*, o trabalho realizado nesta dissertação difere desses trabalhos por ser aplicado de uma forma peculiar no ambiente de TVD. Analisando o contexto atual em TVD para questões de geração dinâmica de conteúdo, é possível verificar que não há um trabalho que realize a especificação de aplicações que se recriam dinamicamente utilizando *templates* de composição.

A Seção 2.1 apresenta as questões relativas à geração de conteúdos dinâmicos em sistemas hipermídia em geral. Em seguida, a Seção 2.2 introduz o conceito de *templates* de composição e como um método orientado a *templates* foi aplicado no presente trabalho para o desenvolvimento de aplicações que se recriam dinamicamente.

2.1.
Conteúdos dinâmicos em sistemas hipermídia

Sistemas hipermídia que dão suporte à geração de conteúdo dinâmico tipicamente permitem que documentos possam embutir um componente interno chamado nesta dissertação de *gerador de conteúdo*. Esse componente realiza a geração e especificação de parte do conteúdo que não é passível de ser criada em tempo de autoria. A geração do conteúdo pelo gerador é feita em tempo de apresentação à medida que ocorrem determinados eventos na apresentação da

aplicação. O gerador captura esses eventos, que informam como deve ocorrer a modificação do documento. Interações do usuário e atualizações de dados da aplicação são alguns dos tipos de eventos que podem ser capturados pelo gerador de conteúdo. Por exemplo, a Figura 2.1 ilustra a visão estrutural de uma aplicação de TVD exemplo que contém um componente gerador de conteúdo. Nessa aplicação, um evento de interação do usuário é capturado pelo gerador de conteúdo, que interpreta esse evento como uma sinalização de que deve adicionar novos conteúdos ao documento.

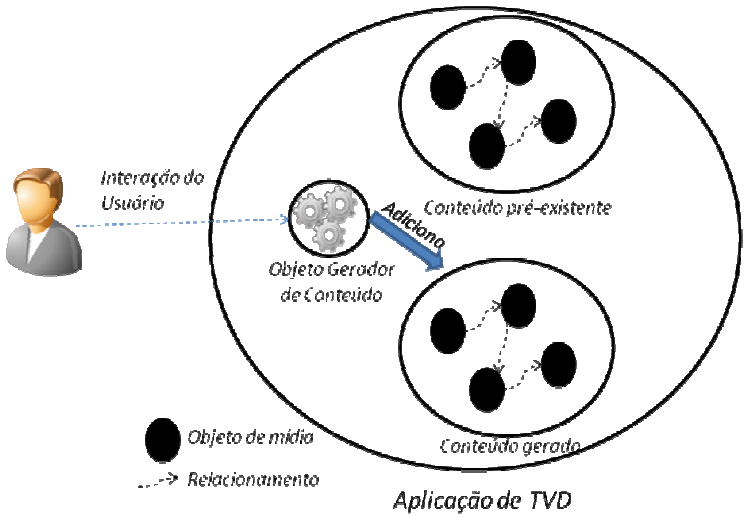


Figura 2.1 - Visão estrutural de uma aplicação de TVD

Geradores de conteúdo são objetos de mídia ou partes da aplicação hipermídia que podem ser executados tanto no espaço do cliente quanto do servidor (provedor) da aplicação. As principais vantagens do gerador ser executado no servidor são a redução da carga de processamento do cliente e a portabilidade. A primeira vantagem é importante nos casos em que o cliente não dispõe de muitos recursos de processamento. A segunda, quando a aplicação é para exibição em múltiplas plataformas.

Quanto ao gerador ser executado no cliente, uma vantagem é a possibilidade de alterar o conteúdo da aplicação sem que seja necessário enviar informações ao servidor, reduzindo o tempo de resposta aos eventos de geração de conteúdo quando o retardo da comunicação pela rede é maior do que o tempo de processamento no cliente. Essa vantagem, no entanto, se perde, se para a geração do novo conteúdo é necessário acessar uma base de dados no servidor. No

ambiente de TVD, talvez a principal vantagem seja a personalização da aplicação, ou seja, a possibilidade da geração ser realizada ou não conforme o desejo e o perfil do usuário cliente.

A *Web* (*World-Wide Web* ou WWW) (Berners-Lee, 1994) é uma referência quando trata-se de sistema hipermídia com suporte à geração dinâmica de conteúdos, tanto no cliente quanto no servidor. Para ambas as abordagens existem várias tecnologias que permitem às páginas HTML (*HyperText Markup Language*) (W3C, 1999) - os documentos de especificação de conteúdos na Web - serem geradas dinamicamente.

Essa geração pode ser no cliente através de geradores de conteúdo utilizando tecnologias como *plug-ins*, JavaScript (Flanagan, 2011), *applets* Java (Sun, 1994) ou diversas outras. *Scripts* em JavaScript e *applets* Java são embutidos no próprio código da página HTML ou por referência e executados pelo navegador do usuário (cliente) que faz a requisição da página. Os eventos que ocorrem enquanto a página está sendo exibida são capturados pelos *scripts* em JavaScript ou pelos *applets* Java, que por sua vez geram conteúdo e editam a página sem que esta precise ser recarregada.

Ainda em relação à *Web*, as páginas HTML podem ser geradas dinamicamente no servidor por meio de geradores de conteúdo baseados em diversas tecnologias como PHP[1], ASP[2] (*Active Server Pages*) ou JSP[3] *(Java Server Pages).* Nessa abordagem, os *scripts* são executados no servidor e geram código HTML pronto para ser exibido ao cliente. Em outros termos, o servidor recebe requisições de cliente por páginas, sendo que essas páginas embutem código de alguma das tecnologias mencionadas, e respondem ao cliente um documento HTML processado com informações dinâmicas (oriundas, por exemplo, da requisição ou de acesso a bancos de dados).

Assim como na Web, alguns sistemas de TVD dão suporte à geração de conteúdos dinâmicos tanto no cliente quanto no servidor. Conteúdos podem ser gerados no cliente por meio de programas que executam no espaço do agente de usuário e que alteram o documento de especificação sendo apresentado. Outra possibilidade é gerar conteúdos no espaço do provedor (servidor) de conteúdos e

[1] http: //www.php.net
[2] http: //www.asp.net
[3] http://www.oracle.com/technetwork/java/javaee/jsp

transmiti-los sob demanda (como nos serviços de vídeo sob demanda) ou como dados não solicitados (como, por exemplo, transmissões em broadcast de TV ao vivo). O Ginga, por exemplo, prevê a presença de um canal de retorno, módulo do sistema que fornece um meio de comunicação entre o usuário (cliente) e o provedor de conteúdo (servidor), permitindo ao usuário demandar conteúdos em tempo de apresentação. Para dados não solicitados, o Ginga utiliza o protocolo de carrossel de objetos DSM-CC que permite a transmissão cíclica de objetos de eventos e de sistemas de arquivos do difusor para os usuários (os arquivos de documentos NCL e os conteúdos de objetos de mídia são organizados em estruturas de sistemas de arquivos).

O Ginga oferece suporte a comandos de edição ao vivo que podem ser transmitidos pelo carrossel de objetos ou gerados a partir de *scripts* Lua. Essa característica permite a alteração de qualquer parte do documento NCL em tempo de apresentação.

Outro exemplo de suporte à edição dinâmica de documentos é BML, a linguagem utilizada no padrão ARIB (ARIB, 2002). BML possui dois elementos para o controle de eventos denominados *bevent* e *bitem*. O elemento *bevent* possui todos os elementos *bitem* que serão processados no documento. Um elemento *bitem*, por sua vez, possui a capacidade de associar, através de seus atributos, código procedural à ocorrência de um evento específico. Cada elemento *bitem* possui um atributo denominado *onoccur*, que possui a identificação da chamada ao código procedural que será executado quando o evento específico ocorrer. Esses códigos procedurais, tradicionalmente especificados em ECMAScript, podem atuar editando o documento do *bitem* associado.

A existência de suporte à geração de conteúdos dinâmicos nesses sistemas de TVD possibilita a especificação de aplicações de TVD dinâmicas. Essas aplicações permitem a personalização de conteúdos para usuários. É viável personalizar aplicações para um número limitado de classes de usuários com conteúdos alternativos pré-gerados. No entanto, é inviável para uma grande quantidade de usuários, quando seria necessário criar e especificar manualmente o conteúdo da aplicação de cada usuário. Para esses casos, um gerador de conteúdo que seja mais complexo do que a simples escolha de alternativas se faz necessário. Se os objetos de mídia geradores de conteúdos têm acesso a informações desses usuários por meio de repositórios (por exemplo, arquivos, bases de dados etc.), os

conteúdos gerados dinamicamente podem ser personalizados de forma mais granular com base nessas informações.

As aplicações dinâmicas, no entanto, também são úteis quando o conteúdo da aplicação precisa ser modificado pela ocorrência de eventos em tempo de apresentação. Em geral, os sistemas que dão suporte à interação notificam os geradores de conteúdo das ocorrências de eventos, e posteriormente, esses geradores tratam as notificações e produzem conteúdos de acordo com análise das informações extraídas dos eventos. Por exemplo, se um usuário interage com uma aplicação pressionando uma tecla de controle remoto, essa interação é um evento que pode ser notificado a um gerador de conteúdo, que analisa qual tecla foi pressionada e gera conteúdos conforme essa análise.

2.2. *Templates* de Composição Hipermídia

Uma composição hipermídia é formada por componentes, representando objetos de mídia ou objetos de composição recursivamente aninhados, e relacionamentos entre eles. Um *template* de composição (Saade, 2003), por sua vez, especifica tipos de componentes, tipos de relacionamentos, componentes e relacionamentos que uma composição possui ou pode possuir, sem precisar especificar quais são todos os componentes e relacionamentos.

A Figura 2.2 ilustra um exemplo de composição hipermídia que possui alguns componentes e relacionamentos entre eles. Os componentes são os objetos de mídia *mainVideo*, uma lista de objetos *bSponsor*, uma lista de objetos *vSponsor* e um objeto de composição *menu*. A quantidade de objetos da lista *bSponsor* é igual à quantidade da lista *vSponsor* (|*bSponsor*| = |*vSponsor*|). É possível extrair uma semântica temporal de apresentação de componentes por meio dessa composição. O objeto de composição *menu* se inicia no momento em que o objeto *mainVideo* começa; no objeto menu, o objeto *vSponsor[i]* se inicia quando o objeto *bSponsor[i]* é selecionado, onde *i* é um valor inteiro que varia entre 1 e |*bSponsor*|. Essa composição poderia ser a especificação de uma aplicação que contém um vídeo principal (objeto *mainVideo*) e um menu de botões (objetos *bSponsor[i]* são botões) que, ao serem selecionados, iniciam a exibição de um

vídeo propaganda (respectivo objeto *vSponsor[i]*) de um patrocinador associado ao botão, como ilustra a Figura 2.2.

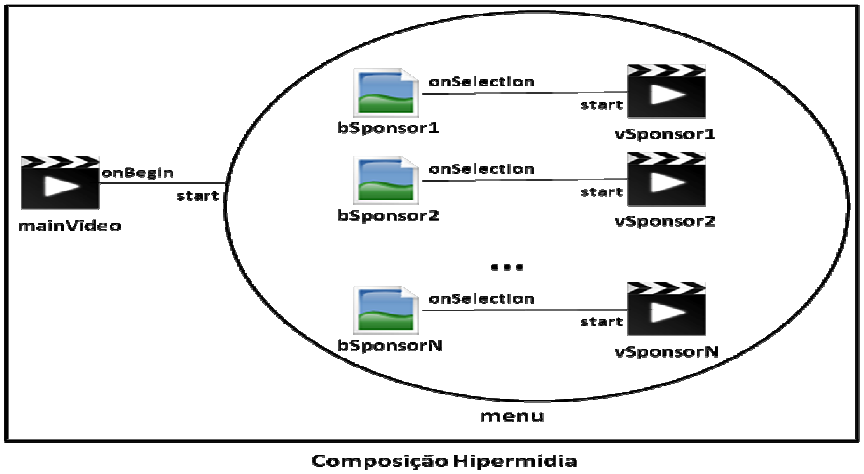


Figura 2.2 - Visão estrutural de uma composição hipermídia

A mesma semântica temporal de apresentação da Figura 2.2 é especificada pelo *template* da Figura 2.3. Esse *template*, em vez de especificar componentes e relacionamentos propriamente ditos como acontece na Figura 2.2, especifica tipos de componentes (identificados como *mainVideo_t, menuButton_t* e *menuVideo_t*) e tipos de relacionamentos. Componentes e relacionamentos concretos também pode estar presentes em *templates*.

De posse de um *template*, o autor de um documento só precisa especificar quais os componentes que fazem parte da composição hipermídia da aplicação. Esse novo documento de especificação é chamado de *documento de preenchimento*. Assim, a composição da Figura 2.2 poderia ser construída a partir da especificação feita pelo *template*, e da identificação do tipo ao qual pertence cada componente (por exemplo, *mainVideo* deve ser identificado como do tipo *mainVideo_t, bSponsor1* como do tipo *menuButton_t*, e assim por diante). O *template* da Figura 2.3 poderia ser reusado tanto pela composição da Figura 2.2, quanto por outras composições da mesma família de documentos. Uma família de documentos é um conjunto de documentos cuja especificação reutiliza a especificação de relacionamentos feita por um *template* em comum.

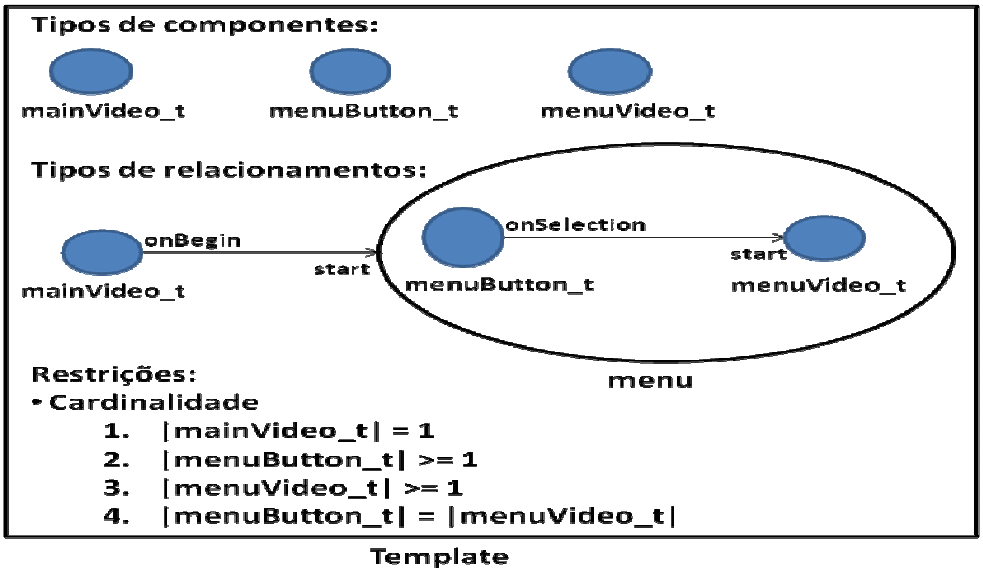


Figura 2.3 - Exemplo de *Template* de Composição

A especificação de um *template* de composição é dividida em duas partes: *vocabulário* e *restrições*. O vocabulário permite o reuso das especificações feitas pelos *templates*, e as restrições definem regras que estabelecem um limite para o vocabulário. O vocabulário consiste nos tipos de componentes, tipos de relacionamentos e pontos de interface, podendo definir número mínimo e máximo de elementos para cada tipo. As restrições podem ser de três tipos. O primeiro tipo consiste em restrições sobre componentes, relacionamentos e pontos de interface. Por exemplo, as restrições de cardinalidade sobre os tipos *mainVideo_t*, *menuButton_t* e *menuVideo_t* da Figura 2.3 determinam que nas composições conformes esse *template*: 1 - há somente um vídeo principal; 2 – há no mínimo um botão de menu de patrocinadores; 3 – há no mínimo um vídeo no menu de patrocinadores; 4 – há a mesma quantidade de botões e vídeos no menu. O segundo tipo consiste em definição de instâncias de componentes através de recursos específicos. Finalmente, o terceiro tipo define relacionamentos concretos entre componentes.

2.2.1. Método de autoria orientado a *templates*

O método de autoria orientado a *templates* proposto em (Soares Neto, 2010) consiste em um conjunto de regras para um processo de autoria de documentos

que possibilita que autores com pouca ou nenhuma experiência em programação criem documentos hipermídia baseados em especificações de *templates* criados por autores mais especializados. A tarefa do autor menos especializado é preencher as lacunas que tornam seu documento único no conjunto da família de documentos caracterizada pelo *template* usado por esse autor.

O autor mais especializado é chamado nesse método de *autor de templates*, e o autor menos especializado é chamado de *autor de documentos*. O autor de *templates* é responsável por identificar e conceber *templates*, e o autor de documentos precisa entender superficialmente o *template* e como preencher suas lacunas.

No âmbito deste trabalho, o gerador de conteúdos faz o papel do autor de documentos, semi-automatizando o processo de geração do documento final. Conceitualmente, o autor de documentos instancia *templates* de documentos, e o resultado final de sua tarefa é a criação de documentos hipermídia em uma linguagem de autoria alvo.

A Figura 2.4 ilustra o método em questão. Nessa figura é possível observar que ambos os autores utilizam um ambiente de autoria próprio para suas tarefas. No caso do ambiente para autoria de documentos, poderia ser provida uma abstração gráfica que permitisse ao autor determinar o preenchimento do *template* sem que seja necessário conhecimento de uma linguagem textual para especificação de documentos de preenchimento. A ferramenta apresentada no Capítulo 4 desta dissertação faz uso de uma abstração gráfica como essa para a aplicação descrita no Capítulo 3. Essa ferramenta permite ao autor de documentos especificar graficamente os documentos conformes ao *template* da aplicação em questão.

O fluxo de trabalho do método se encerra na geração de um *documento hipermídia* em uma linguagem de autoria alvo, como NCL, SMIL (Bulterman, 2004) ou HTML. Esse documento é obtido como saída de um *processador de templates*. Esse processador recebe como entrada um *template* e um documento de preenchimento, e produz uma instância desse *template* como um documento completo pronto para ser exibido.

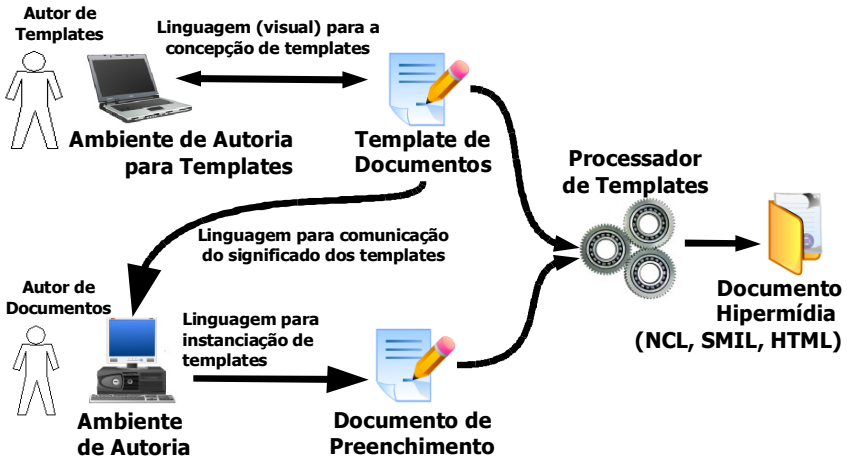
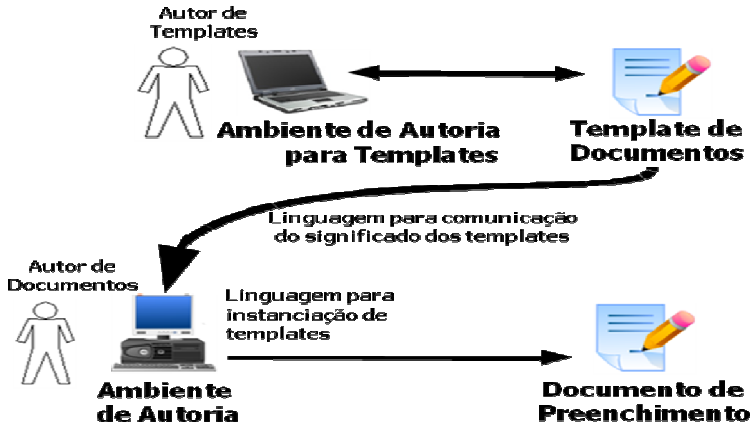
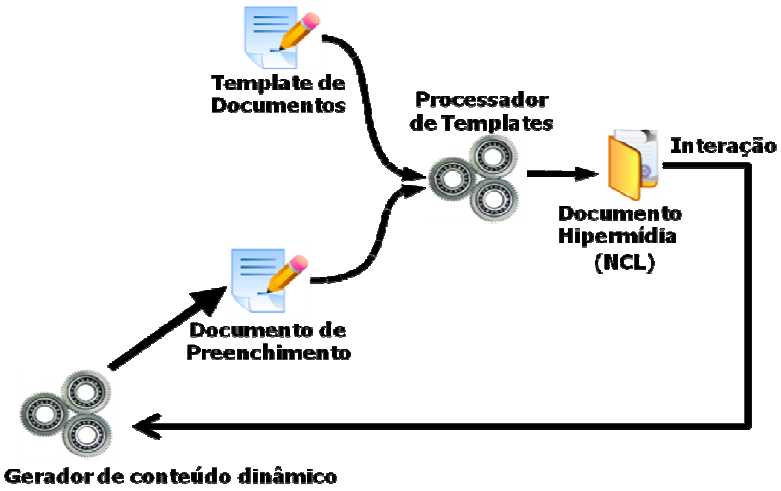


Figura 2.4 - Autoria de Documentos Hipermídia Orientada a *Templates* (Soares Neto, 2010)

O documento hipermídia é produzido pelo processador de *templates* em tempo de autoria. Entretanto, devido à separação entre os tempos de autoria e de apresentação em aplicações dinâmicas não ser muito clara, o fluxo de trabalho representado na Figura 2.4 precisa ser modificado quando o método é direcionado a esse tipo de aplicações. A Figura 2.5 ilustra o método orientado a *templates* adaptado para aplicações dinâmicas. Nesse novo fluxo, a autoria de documentos pode ser dividida em dois momentos. O primeiro momento, representado pela Figura 2.5 (a), é semelhante ao processo da Figura 2.4: o ambiente de autoria de documentos permite ao autor instanciar um documento de preenchimento e o ambiente de autoria para *templates* permite ao autor de *templates* instanciar *templates* de documentos. O segundo momento, representado pela Figura 2.5 (b), consiste na parte dinâmica da autoria de documentos: o *template* e o documento de preenchimento são submetidos ao processador de *templates* para gerar documentos hipermídia à medida que ocorrem eventos de interação no documento. As interações são comunicadas ao gerador de conteúdo dinâmico, responsável por gerar documentos de preenchimento em tempo de execução.



(a)



(b)

Figura 2.5 - Método de autoria orientado a *templates* com gerador de conteúdo dinâmico

A existência de uma interface entre documento e gerador de conteúdo, como ilustra a Figura 2.5, permite a especificação de aplicações que se recriam dinamicamente com o uso de *templates*. A interação do usuário pode ser comunicada ao gerador de conteúdo, que por sua vez produz um documento de preenchimento em tempo de apresentação e envia esse documento ao processador de *templates*, recriando a aplicação por completo.

A adaptação do método orientado a *templates* para aplicações que se recriam dinamicamente foi o meio encontrado no presente trabalho para especificar a arquitetura para aplicações NCL proposta nesta dissertação. Essa adaptação pode ser igualmente aplicada a quaisquer aplicações que se recriem dinamicamente com o uso de *templates*.