Edição e Execução de Apresentações de Documentos Hipermídia no HyperProp*

Guido Lemos de Souza Filho
DIMAp, UFRN
Campus Universitário, Lagoa Nova
59072-970- Natal-RN, Brasil
guido@dimap.ufrn.br

Luiz Fernando Gomes Soares
Depto. de Informática, PUC-Rio
R. Marquês de São Vicente 225
22453-900 - Rio de Janeiro, Brasil
lfgs@inf.puc-rio.br

Resumo

Para editar a apresentação de um documento hipermídia é necessário utilizar linguagens para especificação do sincronismo dos vários componentes do documento, bem como um modelo que capture esta especificação. Para executar uma apresentação, um sistema deve ser capaz de, em tempo de execução, avaliar o relacionamento temporal e espacial entre os diversos componentes, programar temporalmente a recuperação destes componentes, e programar temporal e espacialmente sua exibição. Este artigo descreve o ambiente HySEE (Hypermedia Show Editor and Executor), que agrupa os módulos responsáveis pela edição e pelo controle da execução das apresentações de documentos hipermídia no sistema HyperProp.

Abstract

In order to edit the presentation of a hypermedia document one needs to use a language to specify the synchronism of the various components of the document, as well as a model that can capture this specification. To run a presentation, a system must be able to evaluate, in real time, the temporal and spatial relationships among the components, to program the retrieval of these components, and to program temporally and spatially its exhibition. This paper describes the HySEE (Hypermedia Show Editor and Executor) environment, which groups the modules that take care of the edition and execution control of the hypermedia documents presentation in the HyperProp system.

1. Introdução

Este artigo descreve o ambiente de edição e execução de apresentações de documentos hipermídia do sistema HyperProp [Soar95]. O termo *apresentação* é usado neste texto para denotar as atividades de entrega das várias mídias ao usuário. Para imagens e texto, apresentação implica em exibição na tela. Para os componentes de áudio e vídeo, apresentação indica reprodução dinâmica audível e visual do dado. As apresentações multimídia, que serão aqui discutidas, são *interativas*, isto é, seus autores podem definir pontos onde poderão ser capturadas ações dos espectadores, determinando como as apresentações devem prosseguir.

* Este trabalho foi desenvolvido com o apoio do CNPq, através do projeto PROTEM II - HyperProp, e com o apoio da Embratel, através do projeto RAVel.

Uma apresentação, normalmente, é formada por um conjunto de componentes: imagens, vídeos, textos, trechos de áudio e inclusive outras apresentações já editadas. Os conceitos de nó terminal (que armazena um fragmento de informação) e nó de composição (que armazena uma coleção de nós terminais e de composição, recursivamente) do NCM (Nested Context Model), modelo conceitual adotado no ambiente em questão, ajustam-se com perfeição a situação [SoCR95]. Para definir apresentações, os componentes dos documentos devem ser posicionados no espaço (sincronismo espacial) e no tempo (sincronismo temporal). Em muitas situações, o posicionamento dos componentes é relativo, ou seja, depende de outro(s) componente(s). Isto ocorre porque o espaço e o tempo ocupado por alguns dos componentes podem não ser conhecidos com precisão no momento da edição da apresentação (interações com usuários, reflexos de mudanças na velocidade da exibição quando uma apresentação é executada em máquinas diferentes etc.). Uma representação adequada para as apresentações precisa fornecer um mecanismo que capture esses relacionamentos de sincronismo entre os componentes. No NCM, os elos fornecem esse mecanismo. Extensões ao modelo conceitual NCM, descritas em [SoSC95], o dotaram de recursos para capturar a especificação do sincronismo temporal e espacial, que determina como um documento hipermídia deve ser exibido para os usuários através dos dispositivos de entrada/saída de um computador.

Para editar uma apresentação de um documento hipermídia é necessário utilizar uma ou mais linguagens para especificação do sincronismo dos vários componentes do documento, bem como um modelo que capture esta especificação. Para executar a apresentação, um sistema deve ser capaz de, em tempo de execução, avaliar o relacionamento temporal ou espacial entre os diversos componentes, programar temporalmente a recuperação destes componentes e programar temporal e espacialmente sua exibição. Na seção seguinte são descritos os módulos do sistema HyperProp responsáveis por essas funções, o conjunto destes módulos foi denominado ambiente HySEE — HyperProp Show Editor and Executor.

2. O Ambiente HySEE

O ambiente de edição e execução de apresentações multimídia HySEE é composto pelos módulos de edição, execução e intercâmbio, como mostra a Figura 1.

No módulo de edição, o autor tem a seu dispor duas formas para descrever suas apresentações: textualmente, através das linguagens SOD (Show Objects Description) e SDD (Show Dynamic Description), ou graficamente, através da linguagem SGD (Show Graphical Description). A descrição da apresentação nas linguagens SOD, SDD ou SGD é traduzida para entidades NCM, que se constituem na representação interna da descrição no ambiente HySEE.

O modulo de execução gera, com base na representação NCM, planos para a exibição dos documentos e ajusta estes planos em resposta a eventos (comunicados pelos exibidores) que ocorrem em tempo real, como, por exemplo, as interações com os usuários. Além de computar os planos de exibição, o módulo de exibição controla os exibidores, despachando as informações a serem apresentadas e gerenciando sua apresentação.

No módulo de intercâmbio, a representação NCM dos documentos hipermídia é traduzida para uma representação MHEG. Note que todas as facilidades do ambiente se aplicam a qualquer modelo em conformidade com a proposta de padrão para intercâmbio de documentos multimídia e hipermídia MHEG.

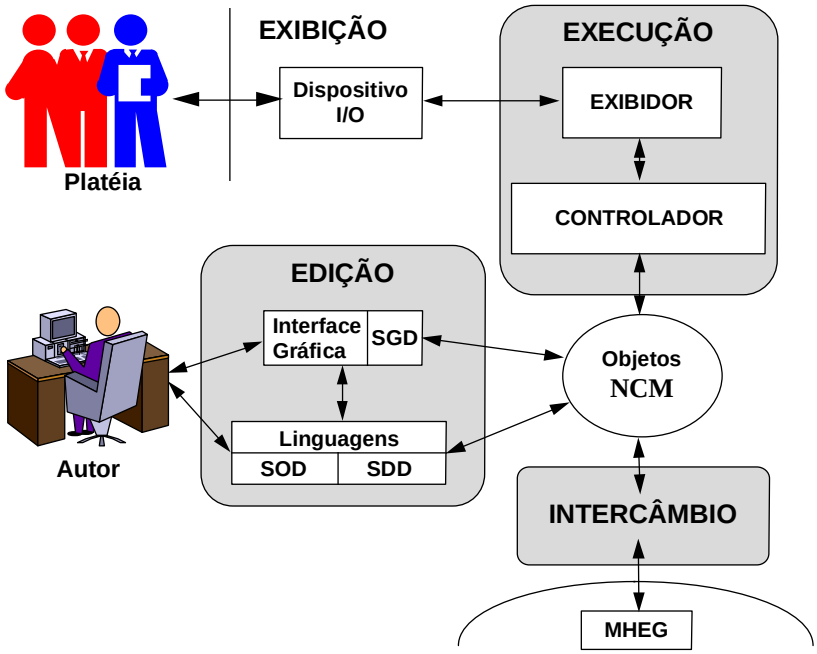


Figura 1: Arquitetura do ambiente HySEE**.**

2.1. O Modelo de Documentos NCM

O modelo de documentos adotado no ambiente HySee é o Modelo de Contextos Aninhados (NCM — Nested Context Model) [SoCR95]. A hierarquia de classes desse modelo é ilustrada na Figura 2.

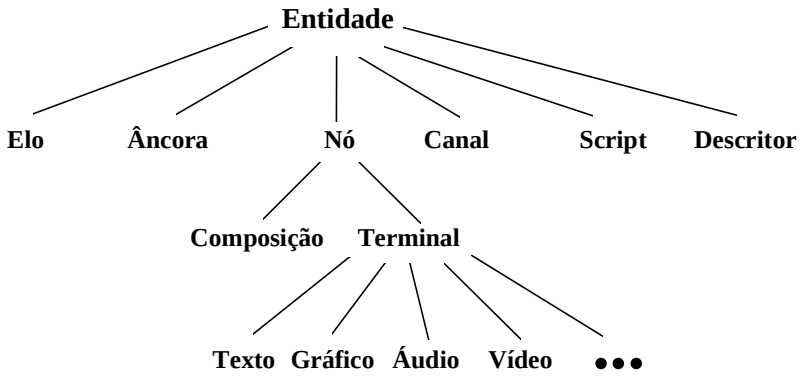


Figura 2: Hierarquia de classes do NCM.

Uma *entidade* é um objeto que possui como atributos no mínimo um *identificador único*, uma *lista de controle de acesso* e um *descritor da entidade*. O identificador único possui o significado usual. Para cada atributo da entidade, a lista de controle de acesso possui uma entrada que associa um usuário ou grupo de usuários aos seus respectivos direitos de acesso ao atributo. O descritor da entidade contém informações que determinam como a entidade deve ser apresentada ao usuário, a semelhança das especificações de apresentação do modelo Dexter [HaSc90].

Um *nó* é uma entidade que possui como atributos adicionais uma *lista de âncoras* e um *conteúdo*. O primeiro conceito será endereçado mais adiante e a definição exata do conteúdo

do nó depende da classe do nó. Um *nó terminal* é um nó cujo conteúdo e a lista de âncoras são ambos dependentes da aplicação. O modelo permite que a classe nó terminal seja especializada em outras classes (*texto, áudio, vídeo* etc.). Um *nó de composição C* é um nó cujo conteúdo é uma coleção *L* de nós e elos. Uma entidade E que pertence a *L* é um *componente de C*, diz-se que *E está contida em C*. Adicionalmente, uma entidade *A* está *recursivamente contida em B* se e somente se *A* está contida em *B* ou *A* está contidada em uma entidade recursivamente contida em *B*. A classe nó de composição pode ser especializada em outras classes, que não têm interesse direto para a discussão em questão.

Os elementos da lista de âncoras, atributos de um nó terminal ou de composição, são chamados *âncoras* do nó. Uma âncora é uma entidade que possui como atributo adicional uma *região*. Uma região de um nó terminal pode ser o símbolo especial λ, representando todo o conteúdo do nó, ou um conjunto de unidades de informação marcadas. A noção exata do que constitue uma unidade de informação (UI) é parte da definição do nó terminal. Por exemplo, uma unidade de informação de um nó vídeo pode ser um quadro, enquanto uma UI de um nó texto pode ser um caracter. Qualquer subconjunto das unidades de informação de um nó terminal pode ser marcado. Em um nó de composição *C*, a região associada a uma âncora deve ser o símbolo especial λ (novamente representa todo o conteúdo do nó) ou um subconjunto de *L* contendo apenas nós.

Um elo é uma entidade que possui três atributos adicionais: o *conjunto de extremidades de origem*, o *conjunto de extremidades de destino* e o *ponto de encontro*. Os valores dos dois primeiros atributos são conjuntos cujos elementos, chamados *extremidades* do elo, são pares da forma $<(N_k,...,N_1),A>$ onde N_1 é um nó, N_{i+1} é um nó de composição e N_i está contido em N_{i+1}, para todo $i \in [1,k)$, com k>0. *A* é uma âncora de N_1. O nó N_k é denominado nó base do elo. O atributo *ponto de encontro* define condições que devem ser satisfeitas nas extremidades de origem (por exemplo, extremidade selecionada ou apresentada) para que sejam executadas ações (por exemplo, exibir, suspender exibição, retomar exibição, etc.) nas extremidades de destino do elo, conforme discutido nas referências [SoSo95, SoSC95].

Um *descritor* possui como atributos uma coleção de descrições de eventos e uma especificação de iniciação. Cada *descrição de evento* define uma região, uma lista de operações e uma condição de ocorrência. As *regiões* associadas aos eventos correspondem a regiões definidas nas âncoras (extremidades de elos) ou a unidades de informação marcadas onde o comportamento da apresentação de um nó pode ser modificado. A *condição de ocorrência* de um evento baseia-se em uma modificação no estado da região associada, por exemplo, a região começou a ser apresentada ou a a região foi selecionada. As listas de operações dos eventos são usadas para controlar o comportamento da exibição do documento. O atributo *lista de operações* consiste, como diz o nome, de uma lista ordenada de operações, onde cada operação é um par condição/ação (a condição deve ser satisfeita para que a ação seja executada). São exemplos de operações: run, stop, pause, resume, prepare, select, unselect, start-modify, end-modify, speed-up, speed-down etc. Cabe ressaltar que os termos condição e ação são usados com o mesmo significado no NCM e na proposta de padrão MHEG [MHEG95]. Opcionalmente, o atributo lista de operações pode conter um objeto *script*.

O objeto *script* permite, no NCM, uma outra forma de definição do sincronismo temporal e espacial, em alternativa a especificação por elos e a definição de mudanças no comportamento dos objetos em listas de operações. A função do objeto script é encapsular um programa escrito em uma linguagem cujas instruções, quando executadas, geram mensagens invocando a ativação de métodos válidos nos objetos NCM. Um exemplo de uma tal linguagem é a SDD

descrita na Seção 2.2.2. O modelo requer que o objeto script possua todos os atributos e métodos necessários para interpretar e controlar sua própria execução.

A *especificação de iniciação* de um descritor contém todas as informações necessárias para iniciar a apresentação de uma entidade, em especial, define os métodos que serão usados para exibir e editar a entidade e os *canais* (abstrações de dispositivos de entrada ou saída) que serão usados na apresentação.

2.2. Edição de Apresentações de Documentos Hipermídia

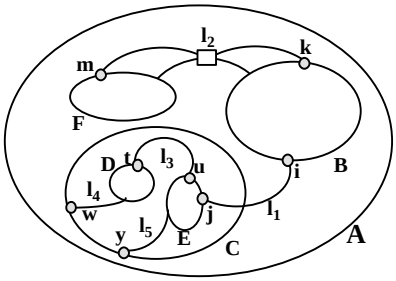
No ambiente HySEE o autor edita a apresentação de um documento definindo sua estrutura, isto é, a hierarquia de composições e nós terminais que compõem o documento e como esses nós se relacionam entre si através de elos. No processo de edição o autor pode também programar a execução de modificações no comportamento dos objetos, que ocorrerão ao longo de suas exibições, em resposta a ocorrência de eventos ou navegações através de elos. Note que o produto do processo de edição é um *plano de exibição* para uma apresentação multimídia interativa. O termo *plano* caracteriza o fato que a apresentação, propriamente dita, só é definida *em tempo de execução*, pois alguns aspectos que influenciam diretamente o seu sincronismo temporal, como o tempo gasto em uma interação com um espectador, não podem ser determinados *a priori*. Para especificar a apresentação de um documento no HySEE, o autor tem a sua disposição as linguagem SOD (Show Object Description) e SDD (Show Dynamic Description). Que serão apresentadas nas Seções 2.2.1 e 2.2.2, respectivamente.

O ambiente HySEE permite, adicionalmente, que o autor programe a apresentação de documentos hipermídia através de uma linguagem gráfica denominada SGD. Na *linguagem SGD* são definidas operações, baseadas na manipulação direta dos componentes da interface gráfica do ambiente HySEE, através das quais o autor pode criar, gravar, ler e posicionar objetos, NCM, representando componentes do documento, no tempo e no espaço, bem como estabelecer relações entre esses objetos. A referência [Souz94] traz uma especificação das operações baseadas em manipulação direta definidas pela linguagem SGD. Esta especificação utilizou uma adaptação do paradigma GOMS (Goal, Operators, Methods & Selection Rules) [CaMN83], proposta em [Souz93] para a modelagem das tarefas que o usuário pode executar usando a interface do ambiente HySEE. Foram especificadas também, as regras da TAG (Task-Action Grammar) [PaGr86] que descrevem a estrutura das ações da interface. Cabe ressaltar que a linguagem SGD é uma forma alternativa às linguagens SOD e SDD na especificação de uma apresentação. Uma implementação de uma interface para a linguagem SGD é descrita em outro artigo desses Anais [CoSS96].

Resumindo, no ambiente HySEE, o modelo NCM de uma apresentação é compilado a partir de descrições das características dos objetos que a compõem, elaboradas nas linguagens textuais SOD e SDD ou na linguagem gráfica SGD. O modelo NCM atua como referencial semântico na descrição das apresentações.

2.2.1. A Linguagem SOD

Na *linguagem SOD*, cuja sintaxe é descrita na referência [Souz94], o usuário dispõe de um conjunto de classes pré-definido pelo modelo NCM. Com base nessas classes, ele compõe suas apresentações, criando objetos e definindo, através dos seus atributos, a forma e o momento que devem ser exibidos. A Figura 3 mostra um exemplo simplificado de um documento NCM descrito na linguagem SOD. Por limitação de espaço, apenas parte da especificação em SOD é listada.



COMPOSITE NODE A: Nodes: {C, B, F}
Links: {l_1, l_2}
Descriptor: desc1

COMPOSITE NODE C: Nodes: {D, E}
Links: {l_3,l_4,l_5}
Descriptor: desc2

TERMINAL NODE B: Content: Graphic-animation
Anchors: {i, k}
Descriptor: desc3

TERMINAL NODE F: Content: Audio
Anchors: {m}
Descriptor: desc10

TERMINAL NODE E: Content: Text1
Anchors: {j, u}
Descriptor: desc11

TERMINAL NODE D: Content: Text2
Anchors: {t}
Descriptor: desc12

LINK l_1: Source[1]: <(B), i>
Target[1]: <(C, E), j>
Meeting Point: Selection(Source[1]) → Run(Target[1])

LINK l_2: Source[1]: <(B), k>
Source[2]: <(F), m>
Target[1]: <(B), λ>
Target[2]: <(F), λ>
Meeting Point: Selection(Source[1]) and not(presentation(Source[2])) → Run(Target[1]), Stop(Traget[2])

LINK l_3: Source[1]: <(C,D), t>
Target[1]: <(C, E), u>
Meeting Point: Selection(Source[1]) → Run(Target[1]) with desc12

LINK l_4: Source[1]: <(C), w>
Target[1]: <(D), λ>
Meeting Point: Selection(Source[1]) → Run(Target[1]) with desc4

LINK l_5: Source[1]: <(C), y>
Target[1]: <(E), λ>
Meeting Point: Selection(Source[1]) → Run(Target[1]) with desc5

DESCRIPTOR desc1: Start-up: Play-content
Channels: { }
Events: {(E_1 | λ | presentation | run SDD_SCRIPT_1)}

DESCRIPTOR desc2: Start-up: Browse-content
Channels: chan1

DESCRIPTOR desc3: Start-up: Play-content with Animator
Channels: chan2

DESCRIPTOR desc4: Start-up: Play-content with Word-for-Windows
Channels: chan3

DESCRIPTOR desc10: Start-up: Play-content with AudioTool
Channels: chan5
Events: {(E_1 | λ | presentation | set volume 0.7),
(E_2 | m | presentation| set volume 0.5)}

DESCRIPTOR desc11: Start-up: Play-content with WordPerfect
Channels: chan4

DESCRIPTOR desc12: Start-up: Play-content with Synthesizer
Channels: chan5

CHANNEL chan1: type: display position: <50,100,300,400>

CHANNEL chan2: type: display position: <100,0,500,700>

CHANNEL chan3: type: display position: <0,0,200,300>

CHANNEL chan4: type: display position: <400,0,600,300>

CHANNEL chan5: type: audio volume: 0.6

Figura 3: Exemplo simplificado de parte da especificação SOD de um documento NCM.

2.2.2. A Linguagem SDD (Show Dynamic Description)

A linguagem SDD (Show Dynamic Description) é uma linguagem de programação de objetos script. Ela complementa a SOD permitindo que o autor de uma apresentação especifique o sincronismo espacial e temporal, utilizando instruções de mais alto nível que as da linguagem SOD. As instruções da linguagem SDD são executadas quando os scripts que as contêm são ativados em consequência da ocorrência de eventos ao longo da apresentação dos documentos. O termo dynamic no título da linguagem se deve ao fato de suas instruções só serem executadas quando o documento está sendo apresentado. Cabe ressaltar que as instruções da linguagem SDD englobam as da linguagem SOD, usadas nos atributos ponto de encontro e lista de operações, acrescidas de construções mais complexas. A linguagem SDD é descrita em mais detalhes na referência [Souz94].

A sintaxe da linguagem SDD é semelhante a da linguagem *Pascal*, sendo as instruções de controle de seqüência e repetição (*if, case, repeat, while, for*) idênticas nas duas linguagens. Um programa SDD, a semelhança de um programa Pascal, é composto por uma lista de instruções que deve ser executada sequencialmente.

A principal instrução da linguagem SDD é a instrução *show*. A execução desta instrução implica na exibição de uma região ou de um grupo de regiões que, neste caso, são exibidas em paralelo. Portanto, a composição sequencial das regiões de uma apresentação é definida na linguagem SDD através da sequência segundo a qual são dispostas as instruções *show* em um programa SDD. A composição paralela das regiões é especificada diretamente nas instruções *show* que comandam a exibição simultânea de um grupo de regiões. Neste caso a instrução exige como parâmetros:

1. *Região base*: aquela que determina a duração da exibição do grupo, a região base pode ser uma região específica, a maior ou a menor região do grupo;
2. *Tipo de alinhamento*: define como as regiões devem ser posicionadas no tempo em relação à região base, as regiões podem ser alinhadas no início, no final, centralizadas ou ajustadas de forma a começar e terminar junto com a região base.

Note que uma instrução show na linguagem SDD corresponde a uma ou mais instruções run na linguagem SOD. No exemplo da Figura 3, especificado na linguagem SOD, o evento E_1, definido no descritor *desc1* associa o início da apresentação do nó de composição A ao scritp SDD_SCRIPT_1. Esse script pode, por exemplo, conter uma instrução *show C,B,F start* indicando que os três nós contidos na composição A devem começar a ser exibidos em paralelo (uma ativação do método run em cada um deles) quando for invocado o método run na composição.

Para permitir a definição explícita da sincronização, em alernativa a sincronização por elos da SOD, foram definidas as instruções: *notify* e *wait*. A execução da instrução *notify* provoca o envio de uma mensagem avisando que um evento ocorreu. A instrução *wait* é complementar à instrução *notify*. Sua execução provoca uma pausa na exibição do conteúdo do nó, que fica parado aguardando a chegada de mensagens geradas pela execução de instruções notify. A combinação de instruções wait e notify produz, assim, efeitos semelhantes aos dos elos NCM. A instrução *wait* permite ainda que sejam definidas instruções que são executadas enquanto a exibição do nó estiver suspensa (cláusula *while-waiting*), ou quando o tempo da pausa for menor, ou maior, que determinados valores de tempo (cláusulas *wait-min* e *wait-max*, respectivamente).

Além das instruções mencionadas, a linguagem SDD possui instruções para ler e gravar valores de atributos dos objetos do modelo (*get* e *put*), encerrar a exibição de um nó (*stop*),

suspender e retomar a exibição de um nó (*pause* e *continue*), modificar a seqüência de exibição de um nó (*jump*), e definir um intervalo de tempo dentro do qual o valor de um atributo deve variar de um valor inicial para um valor final linearmente (*make*).

2.3. Intercâmbio de Documentos

A representação NCM obtida via tradução das linguagens de descrição *SOD*, *SDD* ou *SGD* é usada pelo módulo executor para controlar a exibição das apresentações. Esta representação é traduzida para uma representação MHEG correspondente, para fins de armazenamento e intercâmbio, pelo módulo de intercâmbio do ambiente HySEE.

Objetos MHEG não possuem métodos, sendo compostos apenas por atributos (dados). No ambiente proposto, as apresentações, além de editadas e armazenadas, devem ser executadas. Para tornar possível sua execução são necessárias, além dos dados, informações adicionais (métodos) que controlem sua exibição. Isso fez com que fosse adotado um modelo de nível mais alto que o MHEG para representar internamente os componentes da apresentação. A representação interna escolhida foi a definida pelo NCM.

A conformidade com a proposta de padrão MHEG, contudo, permitirá ao sistema intercambiar seus objetos com as mais diversas aplicações multimídia e hipermídia que obedeçam ao padrão, além de permitir a reutilização de objetos gerados por estas aplicações.

2.4. Execução de Apresentações de Documentos Hipermídia

O módulo executor de apresentações é responsável pelo planejamento e acompanhamento da execução das apresentações definidas por objetos NCM, sendo composto por módulos controladores e exibidores.

O módulo controlador interpreta a representação NCM do documento a ser apresentado, e gera um plano para sua exibição. Para definir o plano de exibição, são calculadas estimativas para a duração da apresentação dos componentes do documento e o instante no qual ocorrerão os eventos relevantes do ponto de vista do sincronismo.

Como nos documentos existem objetos cuja duração não pode ser determinada a priori (por exemplo, componentes que modelam interações com o usuário), os planos de exibição devem ser refeitos à medida que as durações da exibição desses objetos são definidas, em tempo de execução. Na realidade, o executor não gera o plano de exibição do documento inteiro, mas sim dos trechos da apresentação do documento que não dependem de componentes cuja duração da exibição é indeterminada. À medida que a duração destes componentes vai sendo determinada, novos trechos *determinísticos* são formados e, consequentemente, novos planos de exibição podem ser calculados.

Ao gerar o plano de exibição, o controlador verifica se ele é factível, isto é, se é possível executar a apresentação do documento, segundo os requisitos da especificação do sincronismo temporal e espacial definidos pela representação NCM do documento, com os recursos disponíveis na máquina onde ele será exibido. Caso não seja possível cumprir os requisitos definidos pelo autor, o controlador modifica a velocidade de exibição dos componentes da apresentação, usando técnicas de otimização para minimizar as modificações. Se, mesmo depois de ajustar a velocidade de exibição dos componentes, não for possível apresentar o documento, o controlador adota um procedimento mais radical e passa a eleger partes dos componentes que serão suprimidas quando da exibição. Para eleger as partes dispensáveis, o controlador utiliza informações fornecidas pelo autor da apresentação. Ao especificar a

exibição dos componentes de um documento, o autor pode definir que partes podem ser omitidas, sem que seja prejudicada a integridade da informação armazenada no componente. Por exemplo, em trechos de áudio que armazenam discursos falados, até 50% da duração dos intervalos de silêncio pode ser omitida sem prejuízo da integridade da informação. Se, depois de aplicadas estas duas formas de ajuste ainda não for possível apresentar o documento segundo as restrições definidas na especificação de seu sincronismo temporal e espacial, o modelo NCM da apresentação do documento é considerado inconsistente. Ou seja, no ambiente em questão é impossível apresentar o documento respeitando as restrições de sincronismo definidas no modelo. As inconsistências do modelo são então informadas ao autor da apresentação, que recebe também uma lista com os pontos críticos identificados. Cabe ressaltar que, antes do início da exibição do documento, o controlador só pode detectar as inconsistências nos trechos do documento compostos por objetos cuja duração de exibição pode ser calculada *a priori*.

Note que os planos são computados com base em estimativas para o tempo esperado de ocorrência dos eventos, nem sempre estas estimativas se concretizam, isto é, um determinado evento pode ocorrer antes ou depois do momento esperado, devido à variação estatística nos tempos de processamento e comunicação dos dados que estão sendo apresentados. Nesse caso o plano que está sendo executado pode tornar-se inconsistente. Por exemplo, suponha que sejam definidos dois evento: E_1 marcando o fim da exibição de um nó de áudio A_1 no canal C_1 e E_2 marcando o início da exibição do nó de áudio A_2 também no canal C_1. Suponha que, no plano de exibição, o evento E_1 foi escalonado para ocorrer 2 segundos antes que o evento E_2. Nesse caso, se E_1 atrasar ou E_2 adiantar por um tempo maior que 2 segundos, o plano que inicialmente era consistente torna-se inconsistente, devido a um conflito no uso do canal C_1. A detecção desse tipo de inconsistência também dispara o procedimento de ajuste explicado no parágrafo anterior.

O acompanhamento da execução dos componentes é realizado com base em um protocolo que define a interface entre o módulo controlador do sincronismo das apresentações e os módulos exibidores dos componentes dos documentos. Este protocolo define mensagens padronizadas usadas para comunicação entre objetos NCM, comunicando a ocorrência de eventos e carregando ativações de métodos a serem executados pelos objetos de destino. A divisão da função de execução de apresentações em dois módulos: o controlador do sincronismo e o exibidor de componente, permite implementar, no módulo exibidor, um tradutor de mensagens NCM padronizadas para comandos específicos de editores/exibidores quaisquer, possibilitando assim a utilização de exibidores comerciais, como o Microsoft Word, o ToolBook, etc. na exibição dos componentes dos documentos hipermídia sob o controle do HySEE.

3. Conclusão

O ambiente de edição e execução de apresentações de documentos hipermídia descrito neste artigo está sendo implementado no âmbito do projeto PROTEM/CNPq - HyperProp. Das características do ambiente HySEE destaca-se o fato de permitir uma visão e um tratamento integrado dos aspectos estruturais e dinâmicos da apresentação dos documentos. Outra característica importante do ambiente, é a existência de uma interface entre os módulos controlador e exibidor de apresentações, que permite a utilização de exibidores quaisquer na apresentação dos componentes dos documentos, conferindo ao HySEE um comportamento de sistema aberto. Por fim, a utilização do padrão MHEG para intercâmbio de documentos reforça ainda mais o comportamento de sistema aberto do ambiente. Embora essas duas

últimas características tenham sido abordadas apenas superficialmente nesse artigo, elas se constituem em aspectos essenciais ao módulo cliente de um sistema hipermídia, particulamente no cliente definido na arquitetura do sistema HyperProp.

Referências

[BuZe92] Buchanan, M.C.; Zellweger, P.T. "Specifying Temporal Behavior in Hypermedia Documents", *Proceedings of European Conference on Hypertext, ECHT'92.* Milano. December 1992.

[CaMN83] Card, S. K.; Moran, T. P.; Newell, A. “The Psychology of Human-Computer Interaction”. Hillsdale. Lawrence Erlbaum \& Associates, 1983.

[CoSS96] Costa, F.; Soares, L.F.G. e Souza, G.L. “Editor e Browser Gráfico para Sincronização Temporal e Espacial de Objetos Multimídia/Hipermídia”. *Relatório Técnico - Laboratório TeleMídia - Depto. de Informática - PUC - Rio de Janeiro*. Março de 1996. (Aceito para publicação no II Workshop em Sistemas Hipermídia Distribuídos, Fortaleza - CE, Maio de 1996).

[MHEG95] MHEG. "Information Technology - Coded Representation of Multimedia and Hypermedia Information Objects - Part 1: Base Notation. *ISO/IEC DIS 13522-1*. 1995.

[PaGr86] Payne, S. J. e Green, T. R. G. “Task-Action Grammars: a Model of the Mental Representation of Task Languages”. *Human-Computer Interaction, vol 2, pp 93-133*, 1986.

[Soar95] Soares, L.F.G.; et al “HyperProp: uma Visão Geral”. *I Workshop em Sistemas Hipermídia Distribuídos*, São Carlos - SP, Julho de 1995.

[SoCR95] Soares, L.F.G.; Casanova, M.A.; Rodriguez, N.L.R. "Nested Composite Nodes and Version Control in an Open Hypermedia System". *International Journal on Information Systems; Special issue on Multimedia Information Systems.* September 1995.

[SoSC92] Souza; G. L.; Soares, L.F.G.; Casanova, M.A. “Synchronization Aspects of an Hypermedia Presentation Model with Composite Nodes”. *ACM Workshop on Effective Abstractions in Multimedia*, in connection with the ACM Multimedia’95, San Francisco, EUA. November 1995.

[SoSo95] Souza, G.L.; Soares, L.F.G. “Synchronization Aspects of a Presentation Model for Hypermedia Documents with Composite Nodes”. *Research Report PUC-RioInf.MCC31/95*. Rio de Janeiro, Brasil. Outubro 1995.

[Souz93] Souza, C. S. “Regularidade e Generalização em Interfaces Gráficas”. *Relatório Técnico, Depto de Informática*, 1993.

[Souz94] Souza, G.L. “Um Ambiente para Edição e Execução de Apresentações Multimídia”. Exame de Qualificação de Doutorado. Departamento de Informática, PUC, Rio de Janeiro. 1994.