Concepção Formal de Aplicações Multimídia Java

R. Willrich[1], P. de Saqui-Sannes[2,3]
[1]INE/CTC/UFSC, Caixa Postal 476, 88010-979 Florianópolis SC, Brasil
[2]ENSICA, 1 place Emile Blouin, 31056 Toulouse Cedex, França
[3]LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse Cedex, França
Email: willrich@inf.ufsc.br, pdss@ensica.fr

Resumo: A linguagem de programação Java tem sido projetada como uma linguagem universal para o desenvolvimento de aplicações multimídia distribuídas, particularmente sobre a Internet. Infelizmente, o projeto de aplicações multimídia Java é uma tarefa complexa, onde o autor deve ser um especialista desta linguagem. Este artigo propõe uma nova abordagem formal para a especificação, análise e geração automática de aplicações multimídia Java. Esta abordagem é baseada em um modelo formal, permitindo a especificação completa e não ambígua de documentos multimídia interativos. Este modelo permite também a aplicação de técnicas de análise do documento. Ele é uma extensão do modelo *Hierarchical Time Stream Petri Nets* [Sénac, 95] orientada para a produção automática de aplicações multimídia Java. A abordagem proposta é implementada por um ambiente de desenvolvimento oferecendo uma interface de alto nível facilitando a produção de aplicações multimídia Java estruturada/hierárquica e temporalmente coerentes mesmo por não especialistas desta linguagem.

Abstract. The Java programming language has been designed as a universal language for the development of distributed multimedia applications over heterogeneous systems and machines, and over the Internet in particular. However, designing a Java multimedia application is a complex task, for which the author must be a specialist of the Java language. The paper proposes a novel formal approach for the specification, analysis and automatic generation of Java multimedia applications. The proposed approach relies on a formal model, which permits complete and unambiguous description of interactive multimedia documents, making it possible to check these documents against undesirable properties. This models extends Hierarchical Time Stream Petri Nets [Sénac, 95] and is geared towards the generation of Java multimedia applications. The proposed approach is implemented in a development environment which offers non specialists a high level interface for producing Java multimedia applications that are structured, hierarchical and temporally coherent.

Palavras-chaves: Sistemas Multimídia Distribuídos, Modelos Formais, Redes de Petri, Linguagem Java, Especificação, Verificação, Concepção.

1. Introdução

Java, uma tecnologia criada pela Sun Microsystems [Sun, 95], é uma linguagem de programação orientada-à-objetos e *multithreaded* cuja característica principal é o seu caráter universal. Uma aplicação Java pode ser interpretada por todo sistema operacional ou máquina que possua uma máquina virtual Java. Esta portabilidade, adicionada a suas bibliotecas gráfica e de rede, tornam a linguagem Java uma boa candidata ao projeto de aplicações multimídia distribuídas. Mas ao mesmo tempo, estas bibliotecas, e mesmo a linguagem Java, tornam difícil a tarefa de um autor de um documento multimídia, que pode ser um artista ou outra pessoa que não seja um especialista na linguagem Java. Além disso, a concepção de documentos multimídia interativos com interfaces amigáveis, robustos, corretos e eficientes é uma tarefa não trivial. A complexidade da concepção destes documentos é grande principalmente pela necessidade da utilização explícita da noção de tempo [Hardman, 94]. A definição de ambientes de criação de aplicações multimídia interativas fornecendo interfaces gráficas de alto nível facilitando a concepção de aplicações multimídia Java pode reduzir grandemente este problema.

Portanto há um espaço vazio para linguagens de descrição de documentos multimídia que trabalhem em um alto nível de abstração. Essa linguagem deveria ser uma ferramenta de modelagem em que a expressão das restrições temporais e sincronizações seja natural. Esta necessidade é satisfeita pelo modelo *Hierarchical Time Stream Petri Net*, proposto por [Sénac, 95]. O modelo HTSPN oferece um formalismo gráfico e matemático para a expressão da sincronização lógica e temporal em sistemas multimídia e hipermídia. Seu fundamento matemático torna possível a realização de técnicas de verificação do comportamento lógico e temporal destes sistemas.

Este artigo propõe uma nova abordagem formal para a especificação, análise e geração automática de aplicações multimídia Java. Esta abordagem é baseada em uma extensão do modelo HTSPN orientada para a produção automática de aplicações multimídia Java, chamada de Java I-HTSPN.

O modelo HTSPN foi definido com documentos hipermidia em mente. A extensão Java I-HTSPN tem como domínio de atuação os documentos multimídia interativos. Os links hipertextos não serão considerados, pois eles não são suportados explicitamente pela linguagem Java (a linguagem alvo escolhida).

O modelo HTSPN foi proposto inicialmente para a especificação da estrutura conceitual de documentos hipermídia distribuídos, incluindo a definição dos componentes de um documento e o comportamento lógico e temporal deste documento. Em [Willrich, 96a], os autores propuseram uma extensão ao modelo HTSPN permitindo a especificação completa de documentos hipermídia e a produção automática de representações ISO MHEG [ISO 13522] dos documentos especificados. Essas representações MHEG podem então ser transferidas e apresentadas em um sistema aberto multimídia. Devido aos poucos trabalhos de implementação de interpretadores da representação MHEG, os documentos hipermídia MHEG só podem atualmente ser lidos por um número muito restrito de pessoas. A nova extensão proposta neste artigo é justificada pela globalização dos recursos necessários à produção e apresentação de aplicações multimídia Java possibilitando que um maior números de pessoas possam utilizar as aplicações desenvolvidas.

Este artigo apresenta também um ambiente de desenvolvimento implementando a abordagem proposta. Esse ambiente fornece uma interface gráfica de alto nível permitindo a concepção de documentos multimídia de uma maneira rápida e natural. Em seguida, ele permite a aplicação de técnicas de análise, possibilitando a obtenção de uma especificação precisa do documento em desenvolvimento. Como resultado final, este ambiente permite a criação de aplicações multimídia Java que podem ser interpretadas por qualquer máquina ou sistema operacional contendo uma máquina virtual Java.

Este artigo é organizado como segue. A seção 2 introduz o modelo HTSPN e sua versão interpretada (I-HTSPN) permitindo a especificação completa de documentos hipermídia e multimídia. A seção 3 apresenta a nova interpretação do modelo HTSPN orientada para a produção de aplicações multimídia Java. Essa extensão é a base de uma abordagem formal para a criação de aplicações multimídia Java apresentada na seção 4. O ambiente de concepção de aplicações multimídia Java implementado a abordagem proposta é apresentado na seção 5. A seção 6 apresenta um exemplo de desenvolvimento de uma aplicação multimídia interativa utilizando o ambiente desenvolvido. Enfim, as conclusões deste trabalho são apresentadas na seção 7.

2. Especificação Formal de Documentos Multimídia e Hipermídia

Três estruturas são importantes na criação de um documento multimídia interativo ou hipermídia [Willrich, 96c]: (i) a **estrutura conceitual** especifica a organização semântica do documento, seus componentes e as relações lógicas e temporais entre estes componentes; (ii) a **estrutura de apresentação** descreve como e onde os diferentes componentes serão apresentados; (iii) a **estrutura do conteúdo** descreve as informações que constituem os componentes. Note que para fins de apresentação, estas estruturas devem ser representadas em um código interpretável que possa ser processado por um sistema multimídia.

A representação gráfica, juntamente com uma fácil modelagem de esquemas de sincronização e a possibilidade de analisar propriedades importantes [Murata, 89], fazem das redes de Petri uma boa candidata para a modelagem de documentos multimídia e hipermídia. Existem vários modelos multimídia e hipermídia baseados em redes de Petri ([Little, 90], [Diaz, 93], [Stotts, 90], [Sénac, 95]). Em particular, o modelo *Hierarchical Time Stream Petri Nets* (HTSPN) [Sénac, 95] é uma extensão do modelo TSPN (*Time Stream Petri Nets*) [Diaz, 93] usado para a especificação e análise da estrutura conceitual de documento hipermídia. Em [Sénac, 95] e [Willrich, 96c] os autores apresentam as vantagens do modelo HTSPN com relação aos outros modelos baseados em redes de Petri.

Em [Willrich, 96a] os autores propuseram uma versão interpretada do modelo HTSPN, chamada de *Interpreted HTSPN* (I-HTSPN), permitindo a especificação completa de documentos hipermídia distribuídos. Interpretar uma rede de Petri é associar uma semântica particular aos lugares e transições da rede. No caso do modelo I-HTSPN, a semântica associada aos lugares HTSPN foi estendida afim de especificar, além da estrutura conceitual, das estruturas do conteúdo e de apresentação do documento. Esta extensão foi orientada para a produção automática de documentos hipermídia MHEG a partir da tradução de uma especificação I-HTSPN validada.

Esta seção introduz ao modelo HTSPN e a sua versão interpretada.

2.1 Modelo HTSPN

O Modelo HTSPN [Sénac, 95] repousa sobre os conceitos estabelecidos pelo modelo de referência Dexter [Halasz, 94] e sobre o modelo TSPN [Diaz, 93] para a especificação da estrutura conceitual de documentos hipermídia. Ele permite a modelagem formal, com extensões temporais, dos três principais conceitos do modelo Dexter:

- Os componentes atômicos do modelo Dexter representam os dados codificados (as informações textuais, seqüências de áudio e vídeo). No modelo HTSPN, um componente atômico é modelado por um arco com um intervalo de validade temporal (IVT) e um lugar do tipo atômico associado a um recurso. A marcação de um lugar atômico representa o início do tratamento do componente atômico modelado, por exemplo, o início do tratamento de um vídeo. O IVT é uma tríplice [x, n, y], onde x, n e y especificam

respectivamente a duração mínima, nominal e máxima admissível do tratamento associado ao componente atômico. Esta modelagem permite a definição de uma tolerância de sincronização em sistemas hipermídia distribuídos [Sénac, 95]. Na **Figura** 1, o lugar *Video*1 modela uma apresentação de um componente atômico, um vídeo, tendo um IVT de [9,10,11].

- No modelo Dexter, as ligações permitem a definição das relações entre e no interior dos componentes. No modelo HTSPN, uma ligação é modelada por um arco temporizado (L, t), onde L é um lugar do tipo ligação. Este tipo de lugar é representado graficamente por um círculo em negrito. A marcação de um lugar ligação representa o início de apresentação da ligação, por exemplo a apresentação de um botão ou de uma ligação hipertexto. O IVT associado a ligação introduz o conceito de ligação temporizada [Sénac, 95]. Por exemplo, o lugar L da Figura 1 especifica uma ligação.
- Um componente composto do modelo Dexter fornece um mecanismo de estruturação hierárquico baseado na composição recursiva de componentes atômicos, ligações e compostos. No modelo HTSPN, um componente composto é modelado por um lugar abstrato, chamado lugar composto, que representa uma sub-rede. Esse tipo de lugar é representado graficamente por um círculo pontilhado. Esta modelagem é ilustrada na Figura 1, onde o lugar C é um lugar do tipo composto.

  A marcação de um lugar composto representa o início do tratamento do cenário multimídia modelado pela sub-rede associada a este lugar. Quando um lugar composto é marcado, o lugar de entrada da sub-rede associada também é marcado (P.e., o lugar *Video11* da Figura 1). A saída da ficha de um lugar composto implica na saída de todos as fichas da sub-rede associada, modelando a interrupção do cenário multimídia modelado pela sub-rede. O fim do tratamento modelado por um lugar composto ocorre quando o lugar de saída da sub-rede é marcado (P.e. quando o lugar *End* é marcado).

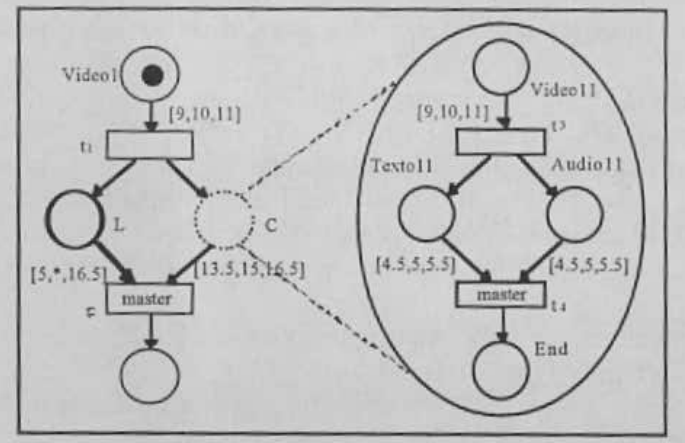

Figura1. Modelagem dos componentes do modelo Dexter

As relações lógicas e temporais entre e no interior dos componentes são modeladas por transições tipadas. Por exemplo, a transição t_2 da Figura 1 define uma relação lógica e temporal entre a ligação L e o cenário multimídia modelado pelo lugar composto C. A fim de descrever os esquemas de sincronização tomando em consideração o não determinismo introduzido pela ginga temporal da duração de tratamento dos componentes do documento, o modelo HTSPN permite que o autor associe às transições uma estratégia de sincronização entre: *and*, *weak-and*, *or*, *strong-or*, *master*, *or-master*, *and-master*, *strong-master*, *weak-master*. A Figura 2 apresenta os intervalos de tiro da transição t_5 para todas as estratégias de sincronização que podem ser associadas a esta transição. Nesta figura, se τ_i é a data de marcação do lugar P_i então $[\tau_i+x_i, \tau_i+n_i, \tau_i+y_i]$, anotados também por $[t_i^{min}, \tau_i+n_i, t_i^{max}]$, especifica o intervalo de validade temporal absoluto (IVTA) do arco $a_i=(p_i,t)$.

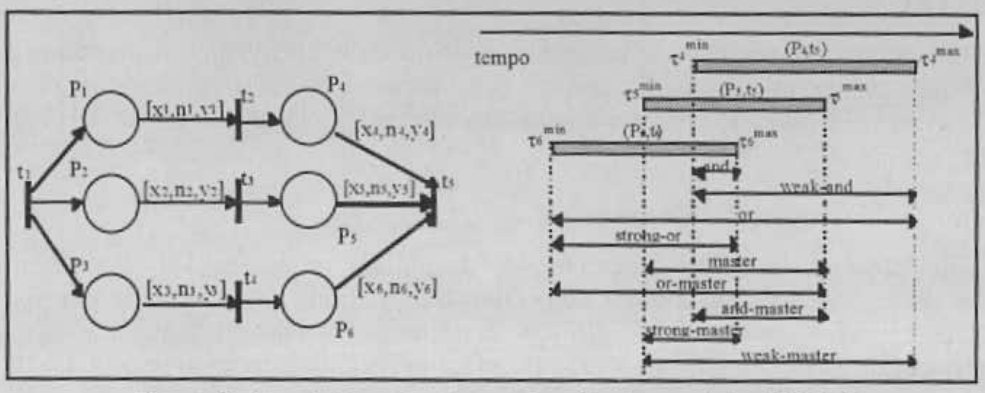

Figura2. Semântica de sincronização do modelo HTSPN

O modelo HTSPN permite a aplicação das seguintes técnicas de análise [Sénac, 96]:

- a verificação da consistência temporal das partes mais críticas dos documentos hipermídia, que são os cenários multimídia;

- a simulação das atividades dinâmicas do documento hipermídia modelado, isto graças ao conceito de ficha de redes de Petri;
- como uma HTSPN pode ser traduzida numa rede de Petri temporal (com intervalos temporais nas transições) todas as técnicas de análise deste último modelo podem ser utilizadas afim de validar a especificação.

Definição formal do modelo HTSPN

A definição formal de um HTSPN se apoia na definição de um TSPN.

Definição 1. Uma TSPN é uma tripla TSPN = (P, T, α, β, M_o, IM, SYN, MA) tal que:

- (P, T, α, β, M_o) define uma rede de Petri, onde P é o conjunto de lugares, T é o conjunto de transições, β e α são respectivamente as funções de incidência para a frente e para traz (que definem respectivamente os arcos entre os lugares e transições, e entre transições e lugares), e M_o é a marcação inicial da rede de Petri.
- IM é uma aplicação que associa um IVT [x, n, y] a um arco de saída de um lugar, onde:
  - $A = \{ a = (p, t) \in PxT \mid \beta(p, t) \neq 0\}$
  - $IM:A \rightarrow (Q^+ \cup \infty) \times (Q^+ \cup \infty) \times (Q^+ \cup \infty)$, $IM(a) [x, n, y]$, $0 \leq x \leq n \leq y$
- SYN é uma aplicação que associa um tipo de sincronização a uma transição:
  - $SYN:T \rightarrow$ {*and, weak-and, or, strong-or, master, or-master, and-master, strong-master, weak-master*}
- $MA:T_m \rightarrow A$ é uma aplicação que associa uma transição de um dos tipos mestre com um arco mestre (representado graficamente por uma flecha em negrito).
  - $T_m = \{t \in T \mid SYN(t) \in \{master, or\text{-}master, and\text{-}master, strong\text{-}master, weak\text{-}master\}\}$

Definição 2. Uma rede HTSPN é uma tripla HTSPN = (R, S, Pin, FS, Fin) tal que:

- $R= (P_r, T_r, \alpha_r, \beta_r, M_{or}, IM_r, SYN_r, MA_r, PT_r, LA_r)$ é uma rede TSPN, chamada de raiz da especificação HTSPN, estendida para modelar os lugares do tipo atômico, ligação e composto.
  - $(P_r, T_r, \alpha_r, \beta_r, M_{or}, IM_r, SYN_r, MA_r)$ define uma TSPN (ver definição 1).
  - $PT_r:P_r \rightarrow$ {*atomic, link, composite*} é uma aplicação de atribuição do tipo de lugar.
  - $LA_r:Al \rightarrow$ *Name* é uma aplicação que associa um nome com um arco de saída de um lugar do tipo ligação, onde:
    - *Name* é um conjunto de nomes associados as ligações;
    - $Al = \{a = (p, t) \in PxT \mid \beta(p,t) \neq 0, PT(p) = link\}$
- $S = \{S_i \mid i \in I\}$ é um conjunto finito de TSPN estendido para a modelagem dos lugares do tipo atômico, ligação e composto, onde:
  - I é o conjunto de índices dos elementos de S.
  - $S_i = (P_i, T_i, \alpha_i, \beta_i, M_{oi}, IM_i, SYN_i, MA_i, PT_i)$.
  - $(P_i, T_i, \alpha_i, \beta_i, M_{oi}, IM_i, SYN_i, MA_i)$ define uma TSPN.
  - $PT_i:P_i \rightarrow$ { *atomic, link, composite* }
  - O conjunto de elementos da rede são pares disjuntos: $\forall (i,k) \in I, i \neq k, ((P_i \cup T_i) \cap (P_k \cup T_k) = \varnothing)$.
- $Pin \subset P$ é o conjunto de lugares iniciais das sub-redes de S, onde $P = \{\cup_{i \in I} P_i\} \cup P_r$
- FS: $C \rightarrow S$ é uma aplicação que associa um lugar composto com um elemento de S e onde $C = \{p \in P_r \mid (\exists\ PT_r(p) = composite)\} \cup \{p \in P \mid (\exists\ i \in I, PT_i(p) = composite)\}$
- $Fin:S \rightarrow Pin$ é uma aplicação que associa um elemento de S com um lugar de entrada.

2.2. Modelo I-HTSPN

Em [Willrich, 96a], os autores propõem uma extensão ao modelo HTSPN afim de permitir a especificação completa de documentos hipermídia a partir da especificação das estruturas conceptuais, de apresentação e do conteúdo. Esta extensão é uma interpretação do modelo HTSPN, chamada de I-HTSPN (*Interpreted-HTSPN*).

2.2.1 Modelagem da Estrutura do Conteúdo

Os dados multimídia são modelados por um conjunto de objetos, chamado DS (*Data Specification*). Estes objetos são instâncias de várias classes *Data*. Cada classe *Data* permite a descrição de um tipo de informação, por exemplo as classes *Video*, *Audio*, *Text*, *StillPicture*, permitem a especificação de videos, áudios, textos e imagens respectivamente. Por exemplo, um vídeo pode ser especificado pelo objeto instância da classe *Video* seguinte:

```
Video DataVideo ::= {
    content          "home\dt\video11.mpg";
    code             ISO_11172_MPEG_VIDEO;
    Original-Characteristics:
```

```
        original-size              128(H) x 256 (W);
        original-duration 10 s;
}
```

Este objeto *Video* especifica que a informação, chamada de *DataVideo*, é um vídeo-clip comprimido MPEG, com um tamanho original de 128 (H) x 256 (V) pixels e tem uma duração original de 10 s, armazenado em /home/dt/video11.mpg.

2.2.2 Modelagem da Estrutura de Apresentação

A estrutura de apresentação de um documento é descrita por um conjunto de objetos chamado PS (*Presentation Specification*). Esses objetos são instâncias das várias classes *Presentation*. Cada classe *Presentation* permite a especificação das características de apresentação de um certo tipo de informação. As classes *PSAudio* e *PSVideo* são exemplos de classes *Presentation* utilizadas para a especificação das características de apresentação de áudios e vídeos respectivamente.

Um conjunto de objetos *Channel*, chamado de CS (*Channel Specification*), especifica todos os canais utilizados pelo documento. Um objeto *Channel* especifica um espaço lógico em que uma ou várias informações serão apresentadas para o usuário (por exemplo, uma janela ou um canal de áudio). Um objeto *Channel* descreve o identificador e os requisitos (espaciais/sonoros) de um canal.

Por exemplo, uma apresentação do objeto *DataVideo*, apresentado anteriormente, pode ser especificado pelo objeto instância da classe *PSVideo* abaixo:

```
PSVideo PVideo ::= {
        data                    DataVideo
        Channel                 VideoC;
        presentation-position   100 (H) x 100 (W);
}
```

Este objeto especifica uma apresentação de *DataVideo* a ser posicionada em 100 (H) x 100 (V) no canal especificado por *VideoC*. Este último é um objeto da classe *ChannelVisual* (uma classe *Channel*) que descreve os requisitos do canal (P.e., os valores mínimos e máximos dos eixos x e y que devem ser satisfeitos pelo dispositivo físico que vai implementar o canal).

2.2.3 Modelagem da Estrutura Conceitual

Como apresentado na seção 3.1, a rede HTSPN permite a especificação da estrutura conceitual de documentos hipermídia. Afim de permitir a especificação completa destes documentos, foi realizada uma interpretação do modelo HTSPN pela associação das características de apresentação aos componentes modelados ao lugares da rede HTSPN. Esta associação é realizada pela função F_{PS}, que associa um lugar do tipo atômico ou ligação a um objeto definido em PS. Além disso, duas outras funções, chamadas F_{DS} e F_{CS}, foram adicionadas permitindo a associação de objetos *Presentation* com objetos *Data* e *Channel*, respectivamente.

Definição formal do modelo I-HTSPN

Definição 3. Uma HTSPN interpretada é uma tripla I-HTSPN = (HTSPN, PS, CS, DS, F_{PS}, F_{CS}, F_{DS}) tal que:

- HTSPN = (R, S, Pin, FS, Fin) define uma HTSPN (ver definição 2).
- PS é um conjunto de objetos das classes *Presentation*.
- CS é um conjunto de objetos das classes *Channel*.
- DS é um conjunto de objetos das classes *Data*.
- F_{PS}:P_{al}→PS, é uma função que associa um objeto *Presentation* com um objeto atômico ou ligação definido em R ou S, onde $P_{al} = \{p \in P_r \mid (\exists\, PT_r(p) \in \{atomic, link\})\} \cup \{p \in P \mid (\exists\, i \in I,\ PT_i(p) \in \{atomic, link\})\}\}$
- F_{CS}:PS→CS é uma função que associa um objeto *Presentation* com um objeto *Channel*.
- F_{DS}:PS→DS é uma função que associa um objeto *Presentation* com um objeto *Data*.

Um aspecto importante desta extensão é que todas as características do modelo HTSPN são preservadas: os esquemas de sincronização lógica e temporal e as regras de tiro são mantidas como definidas em [Sénac, 95]; e todas as técnicas de análise desenvolvidas para o modelo HTSPN são diretamente aplicáveis à versão interpretada deste modelo.

2.2.4 Geração automática de representações MHEG

Em [Willrich, 96a], os autores propuseram uma versão interpretada do modelo HTSPN orientada para a geração de representações MHEG. Os documentos hipermídia assim representados podem ser transferidos em um sistema aberto e apresentados a partir de um interpretador MHEG. Este último é um software que interpreta (apresenta) representações MHEG.

A interpretação do modelo HTSPN orientada para MHEG é realizada no sentido em que foi associada uma semântica, derivada dos conceitos propostos pela norma MHEG, aos lugares do modelo HTSPN. Mais especificamente, os atributos dos objetos *Data*, *Presentation* e *Channel* foram definidos afim de permitir a representação de todas as informações necessárias à geração automática de representações MHEG.

Para a apresentação destes documentos MHEG, foi utilizado o interpretador MHEG desenvolvido pela Euroclid (uma soft-house francesa), sob comando do CCETT (centro de pesquisa da France Télécom - França). Devido a limitações deste interpretador, somente uma pequena parte do poder de expressão do modelo I-HTSPN pôde ser testada. Infelizmente, não existe atualmente um interpretador MHEG suportando todas as funcionalidades necessárias à interpretação da maioria das representações MHEG geradas a partir de especificações I-HTSPN.

Em paralelo com a norma ISO MHEG, a linguagem Java vem alcançando o estado de padrão de fato para a produção de aplicações portáveis. Contudo, o desenvolvimento de aplicações multimídia Java diretamente a partir de sua programação não pode ser abordado eficazmente: numa tal abordagem orientada-linguagem o poder de expressão é muito grande, mas a composição lógica e temporal de uma aplicação multimídia em uma forma textual é difícil de produzir e modificar. A definição de modelos de alto nível permitindo a concepção de aplicações multimídia Java reduziria grandemente estas dificuldades. Neste sentido, a próxima seção apresenta uma nova interpretação do modelo HTSPN orientada para a produção de aplicações multimídia Java.

3. Modelo Java I-HTSPN

Esta seção introduz uma interpretação do modelo HTSPN orientada para a especificação de documentos multimídia interativos e a produção automática de aplicações multimídia Java correspondentes. Esta extensão é chamada de Java I-HTSPN.

O modelo Java I-HTSPN não se trata de uma reformulação do modelo I-HTSPN apresentado na seção anterior. Nesta nova extensão, somente os atributos dos objetos *Data* e *Presentation* foram modificados afim de que as informações de acesso aos dados e as características de apresentação definidas por estes objetos tenham elementos correspondentes na linguagem Java.

3.1 Especificação da Estrutura do Conteúdo

Como apresentado na seção 3.2.1, os dados multimídia utilizados por um documento multimídia são especificados por uma lista de objetos das classes *Data* e cada classe *Data* permite a especificação de um tipo de mídia. No modelo Java I-HTSPN são considerados quatro tipos de informações: imagens, áudios, textos e animações (uma seqüência de imagens apresentadas em intervalos fixos), que são as informações tratadas explicitamente pela biblioteca padrão Java. No modelo Java I-HTSPN, estes dados podem ser descritos por objetos instâncias das classes *Data StillPicture*, *Audio*, *Text* e *Animation*.

As classes *Data* têm como atributo comum o URL (*Uniform Resources Locator*) dos dados multimídia, sendo-a informação de acesso a estes dados. A especificação de diferentes tipos de dados por diferentes classes permite a verificação de que o URL especifica realmente o tipo de médium descrito (identificada pela extensão do arquivo). A classe *Animation* especifica além do URL das imagens que compõem a animação, a duração original da animação e sua velocidade original de apresentação.

Por exemplo, o objeto *StillPicture* abaixo especifica uma imagem.

```
StillPicture MinhaImagem ::= {
        URL    http: www.inf.ufsc.br ~willrich minhaImg.jpeg;
}
```

3.2 Especificação da Estrutura de Apresentação

Como introduzido na seção 2.2.2, as apresentações que compõem uma aplicação multimídia são especificadas por um conjunto de objetos instâncias de uma das classes *Presentation*. No modelo Java I-HTSPN, são definidas as seguintes classes *Presentation*:

- *PSStillPicture*, permite a especificação das características de uma apresentação de uma imagem. Um objeto desta classe especifica o objeto *StillPicture* que descreve os dados a serem apresentados, a posição e o tamanho da apresentação, e uma apresentação alternativa. Este último atributo especifica uma duração máxima de preparação da imagem e uma apresentação alternativa especificada por um objeto *Presentation*. Esta apresentação alternativa pode repor a apresentação principal se houver problemas de acesso, ou se a duração máxima de preparação é ultrapassada.
- *PSAudio*, permite a especificação das características de uma apresentação de um áudio. Um objeto *PSAudio* especifica o objeto *Audio* que descreve os dados a serem apresentados, o tipo de término de apresentação e uma apresentação alternativa. O atributo terminação especifica o comportamento de uma apresentação quando da espera de um ponto de sincronização. Ele especifica uma escolha entre congelar a

apresentação na sua posição final, interromper a apresentação ou exibir uma apresentação alternativa para preencher espaços vazios quando da espera de um ponto de sincronização. Mecanismos similares são propostos por [Steinmetz, 90] e [Hudson, 93]. Para áudios, o atributo terminação pode especificar uma apresentação alternativa (p.e. o mesmo áudio ou outro).

- *PSText*, permite a especificação das características de uma apresentação de um texto. Um objeto instância desta classe especifica o objeto *Text* que descreve os dados a serem apresentados, o tipo de texto (*area* ou *field*), a fonte (*dialog*, *helvetica*, *timesRoman*, *courier* e *symbol*), o estilo do fonte (*bold*, *italic*, *plain*) e o tamanho dos caracteres, além da apresentação alternativa.
- *PSAnimation*, permite a especificação das características de apresentação de uma animação. Um objeto desta classe especifica o objeto *Animation* que descreve os dados a serem apresentados, a posição e o tamanho da apresentação, a posição de partida e fim de apresentação, a velocidade, o tipo de término de apresentação e a apresentação alternativa.
- *PSLink*, permite a especificação das características de apresentação de um botão da aplicação multimídia Java. Um objeto desta classe especifica a posição de apresentação do botão, as cores de fundo e de primeiro plano, e a fonte do texto do botão.

Por exemplo, uma apresentação da imagem *MinhaImagem*, apresentada anteriormente, pode ser especificada pelo objeto *PSStillPicture* seguinte:

```
PSStillPicture PresMinhaImagem ::= {
        data                    MinhaImagem
        presentation_size [ 134 181 ]
        presentation_position   [ 100 420 ]
}
```

3.3 Especificação da Estrutura Conceitual

No modelo Java I-HTSPN, além da rede HTSPN, a estrutura conceitual do documento é especificado por um objeto da classe *Description*. Este objeto permite a especificação de informações gerais sobre o documento, como o título do documento, o nome do autor, versão, as dimensões da janela principal e as cores de primeiro plano e de fundo.

4. Produção Automática de Aplicações Multimídia Java

O modelo Java I-HTSPN fornece os meios para a geração automática de aplicações multimídia Java (um código Java padrão) a partir da especificação dos documentos multimídia. Estas aplicações multimídia Java são a implementação dos documentos multimídia especificados e analisados.

As aplicações Java geradas a partir de uma especificação Java I-HTSPN contém dois conjuntos de classes: a *classe documento*, que contém a especificação Java I-HTSPN do documento e é o ponto de partida da aplicação Java; e *as classes de Interpretação*, que permitem a interpretação de qualquer documento especificado em Java I-HTSPN;

4.1 Classe Documento

A *classe documento* é gerada automaticamente a partir da especificação Java I-HTSPN. Esta classe é o ponto de partida da aplicação multimídia Java. Ela contém como atributo a especificação Java I-HTSPN do documento. A partir desta especificação, a *classe documento* executa as seguintes operações:

- Criação de uma janela principal onde o documento será apresentado. As dimensões, título e cores de primeiro plano e de fundo são definidos pelo objeto *Description* (seção 3.3). Esta janela contém um menu permitindo que o utilizador inicie ou interrompa a apresentação, ou permitindo que ele encerre a aplicação.
- Instanciação de um objeto ativo (implementado em Java por uma *thread*) da classe *Interpretador* (apresentada a seguir).

4.2 Classes de Interpretação

As classes de interpretação são responsáveis pela apresentação de qualquer documento especificado segundo o modelo Java I-HTSPN. Dentre estas classes, a classe *Interpretador* é responsável pela criação e controle de uma *estrutura de interpretação*. Esta estrutura é composta de um conjunto de objetos ativos e comunicantes. Estes objetos são responsáveis pela execução das funcionalidades modeladas pelos lugares, arcos e transições da especificação I-HTSPN. O *Interpretador* instancia e interliga os objetos ativos afim de criar uma estrutura de interpretação correspondente a especificação Java I-HTSPN a ser apresentada.

O processo de criação da estrutura de interpretação é recursivo, a estrutura de interpretação de uma sub-rede I-HTSPN S genérica (incluindo a raiz ou uma das sub-redes de S) é representada por um módulo *Sub-rede*.

Este módulo é uma representação abstrata que representa os objetos ativos (instâncias das classes de interpretação) responsáveis pela interpretação da sub-rede S.

O módulo *Sub-rede* é o bloco elementar de criação da estrutura de interpretação da especificação Java I-HTSPN. A Figura 3 apresenta o módulo *Sub-rede* em uma notação baseada em SA/RT. Nesta notação, os círculos representam processos funcionais, os arcos orientados em traços plenos representam fluxos de dados e em traço pontilhado representam fluxos de controle.

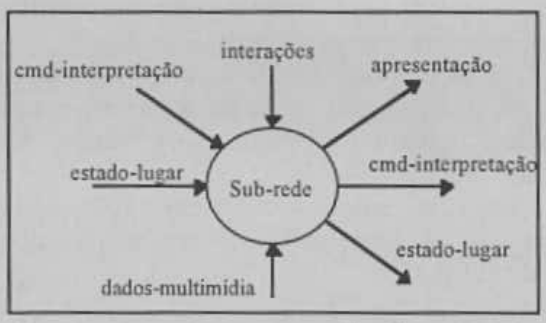

Figura 3. Bloco elementar para a construção da estrutura de interpretação

Um módulo *Sub-rede* recebe os comandos de interpretação (*cmd-interpretação*) de sua sub-rede mãe e recebe também os comandos de interação do utilizador (*interações*). Se a sub-rede é a raiz da especificação, então ela recebe comandos de interpretação pedidos pelo utilizador (via o menu principal). Os dados de entrada são as informações sobre o estado de suas sub-redes filhas (*estado-lugar*) e os dados multimídia. O dado de saída é o estado do lugar composto representando a sub-rede (*estado-lugar*). Estes dados são enviados quando da mudança de estado. O módulo *Sub-rede* envia comandos de interpretação à suas sub-redes filhas (*cmd-interpretação*).

A Figura 4 apresenta um exemplo de estrutura necessária à interpretação de uma especificação Java I-HTSPN cuja raiz contém um lugar composto A e cuja sub-rede associada contém um outro lugar do tipo composto B. Esta estrutura de interpretação contém 3 módulos Sub-rede interligados contendo todas as funcionalidades especificadas pela rede Java I-HTSPN.

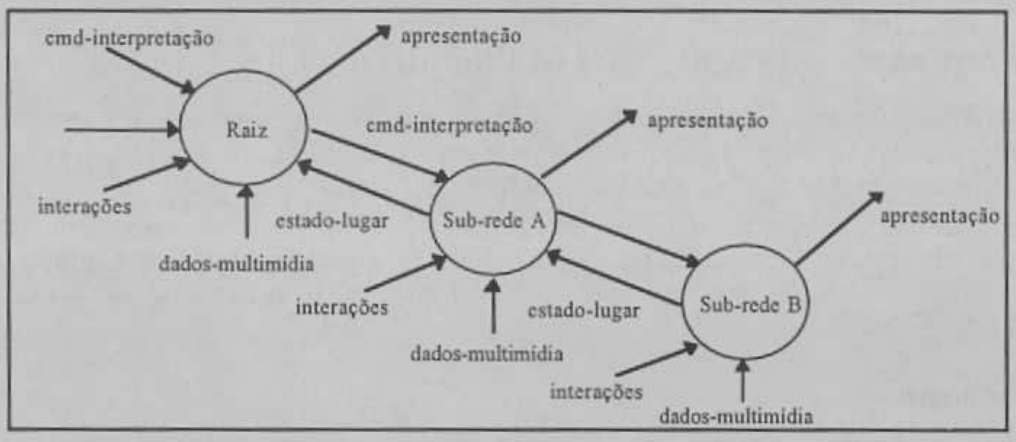

Figura 4. Exemplo de estrutura de interpretação

Um módulo *Sub-rede* S é responsável pela execução das atividades de apresentações dos componentes (modelados pelos lugares atômicos, ligações e compostos) de uma sub-rede S e pelo escalonamento destas apresentações (definido pela marcação, arcos e transições de S). Um controle da validade temporal é realizado sobre estas atividades. O escalonamento da sub-rede S é realizado por uma instância do módulo *Escalonador*. As apresentações modeladas pelos lugares de S são suportadas por instâncias dos módulos *Apresentação-Texto*, *Apresentação-Imagem*, *Apresentação-Áudio*, *Apresentação-Botão*, e *Apresentação-Animação*:

- Cada apresentação de um texto modelada por um lugar atômico é suportado por uma instância do módulo *Apresentação-Texto*.
- Cada apresentação de uma imagem modelada por um lugar atômico é suportado por uma instância do módulo *Apresentação-Imagem*.
- Cada apresentação de um áudio modelada por um lugar atômico é suportado por uma instância do módulo *Apresentação-Áudio*.
- Cada apresentação de uma animação modelada por um lugar atômico é suportado por uma instância do módulo *Apresentação-Animação*.
- Cada apresentação de um botão modelada por um lugar ligação é suportado por uma instância do módulo *Apresentação-Botão*.
- Cada apresentação de um cenário multimídia modelada por um lugar composto S (a raiz da especificação é sempre associada a um lugar composto abstrato) é suportado por uma instância do módulo *Sub-rede* (uma definição recursiva).

Quando da criação da estrutura de interpretação associada a uma sub-rede S, o *Interpretador* é responsável pela interligação do objeto *Escalonador* (representando o lugar composto associado a S) com os objetos ativos que representam os lugares da sub-rede S (podendo ser inclusive outros objetos *Escalonador*). A partir destas interligações, o módulo *Escalonador* recebe mudanças de estados dos lugares e envia comandos de parada e partida de apresentação. Por exemplo, a Figura 5 apresenta o refinamento de um módulo *Sub-rede* necessário à interpretação de uma sub-rede especificando um cenário contendo um texto, uma imagem, um áudio, um botão e uma animação.

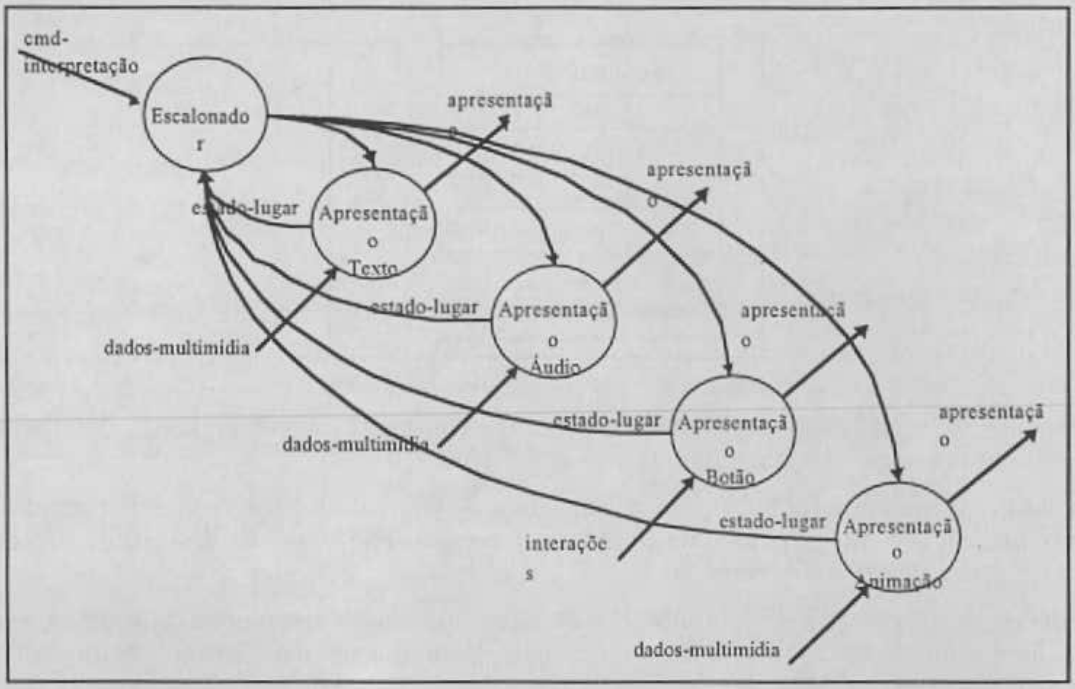

Figura 5. Exemplo de estrutura de interpretação de uma sub-rede Java I-HTSPN

4.2.1. Módulo Escalonador

O módulo *Escalonador* é responsável pelo escalonamento das apresentações modeladas pelos lugares de uma sub-rede S. Ele é composto de dois objetos ativos:

- *Controle*-IVT, responsável pelo controle da validade temporal de S. Este objeto comporta-se como um alarme indicando à *Máquina-RdP* associada a sub-rede mãe de S os instantes em que as durações mínimas e máximas de tratamento de S são atingidos.
- *Máquina-RdP*, responsável pela evolução da sub-rede S. A partir do estado da rede, este objeto identifica as transições tiráveis e as dispara. O tiro de uma transição implica na interrupção das apresentações de entrada desta transição, na partida das apresentações de saída, e na mudança da marcação da rede de Petri. A verificação das transições tiráveis é realizada quando da mudança do estado da sub-rede. Quando o lugar de saída da sub-rede torna-se marcado, o objeto *Máquina-RdP* envia informações sobre o estado do lugar composto à sub-rede mãe.

4.2.2 Módulos de Apresentação

Os módulos *Apresentação-Texto*, *Apresentação-Imagem*, *Apresentação-Áudio* e *Apresentação-Animação* são responsáveis pela apresentação modeladas pelos lugares atômicos de S. Eles são compostos pelos seguintes objetos ativos:

- *Apresentação*, responsável pela apresentação modelada por um lugar do tipo atômico. Existe uma classe Apresentação especializada para a apresentação de cada um dos tipos de informação suportadas pelo modelo (textos, imagens, áudio e animações).
- *Controle-IVT*, responsável pelo controle da validade temporal da apresentação. Este objeto comporta-se como um alarme indicando à *Máquina-RdP* associada a sub-rede S os instantes em que as durações mínimas e máximas de tratamento modelado pelo lugar atômico são atingidos.
- *Apresentação-Alternativa*, responsável pela mudança de apresentação se a apresentação principal não pode ser totalmente preparada em um certo intervalo de tempo.

4.2.3 Módulos Apresentação-Botão

O módulo *Apresentação-Botão* é responsável pela apresentação de um botão modelado por um lugar do tipo ligação. Ele é composto pelos seguintes objetos ativos:

- *Botão*, responsável pela apresentação do botão e dá o suporte necessário às interações com o utilizador.
- *Controle-IVT*, responsável pelo controle da validade temporal da apresentação.

5. Ambiente de Produção de Aplicações Multimídia Java

Um ambiente Java, chamado *ambiente Java I-HTSPN*, está sendo implementado afim de auxiliar o processo de criação, de análise e de apresentação de documentos multimídia interativos. Os módulos funcionais deste ambiente são apresentados na Figura 6.

Figura 6. Ambiente Java I-HTSPN

Baseado no modelo Java I-HTSPN, este ambiente implementa uma metodologia de desenvolvimento de documentos multimídia comportando as seguintes etapas Figura 7:

A) Utilização do modelo Java I-HTSPN para a especificação formal do documento multimídia interativo. Este modelo fornece os meios para uma descrição precisa e não ambígua das estruturas conceitual, de apresentação e do conteúdo do documento.

B) Simulação da especificação HTSPN, afim de verificar a correção do comportamento lógico e temporal do documento. Se o comportamento não é aquele desejado, o autor deve retornar à primeira etapa;

C) Aplicação de técnicas de verificação suportadas pelo modelo I-HTSPN. Estas técnicas são: a verificação da consistência temporal dos cenários multimídia modeladas e a aplicação das técnicas desenvolvidas para o modelo redes de Petri temporais (após a tradução da HTSPN em uma rede de Petri temporal). Se existem erros, o autor deve retornar à primeira etapa;

D) Teste do documento multimídia pela interpretação de sua especificação Java I-HTSPN analisada. Nesta fase, o autor pode visualizar e interagir com documento multimídia em desenvolvimento. Se o comportamento desta apresentação não é aquele desejado, o autor deve retornar à primeira etapa da metodologia.

E) Produção automática de uma aplicação multimídia Java implementando o documento multimídia interativo especificado (cujo procedimento foi apresentado na seção 4).

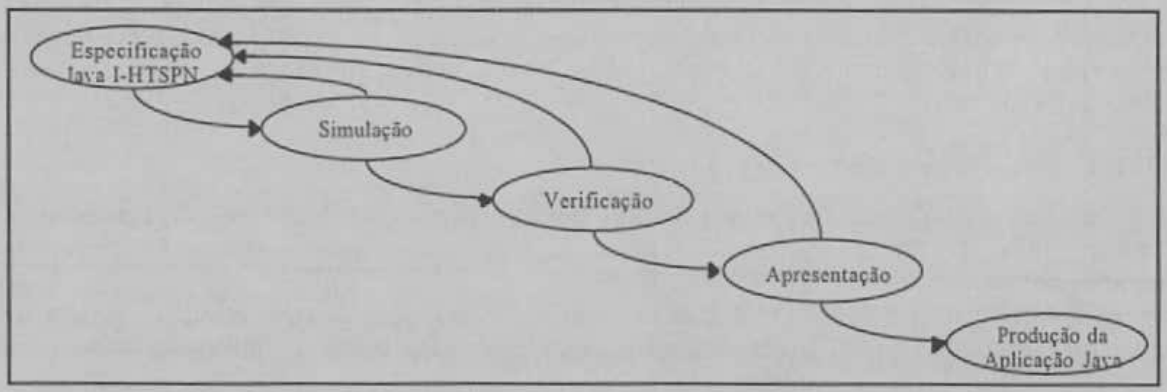

Figura 7. **Ciclo de vida de concepção de aplicações multimídia Java**

5.1 Editor I-HTSPN

O Editor fornece ao autor os meios de construir, de maneira gráfica, o documento multimídia interativo. Este editor gráfico facilita a especificação dos objetos *Data* e *Presentation*, além da construção da rede HTSPN.

Os objetos *Data* são especificados a partir de uma janela *Data Specification Editor*. Esta janela contém uma lista de objetos *Data* e fornece os meios para a adição e exclusão de objetos nesta lista, e a atribuição de valores aos atributos dos objetos *Data*. Por exemplo, a janela *Data Specification Editor* da Figura 8 apresenta a lista de objetos *Data* de um documento multimídia. Esta figura ilustra a especificação de um áudio localizado no URL http://www.inf.ufsc.br/~willrich/audio1.au.

A estrutura conceitual e de apresentação do documento podem ser especificadas a partir de um editor de redes Java I-HTSPN. Neste editor, o autor pode criar os lugares, transições e arcos da rede, e editar os atributos associados a estes objetos. Para esta edição dos atributos dos objetos são utilizados janelas de diálogo. Por exemplo, para os lugares do tipo ligação e atômico esta janela de diálogo permite a especificação dos atributos do objeto *Presentation* associado ao lugar.

A Figura 8 apresenta um exemplo de especificação da estrutura de apresentação e conceitual de um documento multimídia interativo. Esta figura ilustra a especificação de uma apresentação da informação *Audio1Data*, modelada pelo lugar *audio1*.

Após a realização de todas as etapas de desenvolvimento, o autor pode criar uma aplicação multimídia Java correspondente ao documento multimídia especificado. Esta operação deve ser requisitada pelo utilizador do ambiente via o menu principal do Editor Java I-HTSPN.

5.2 Simulador

Este módulo fornece ao autor os meios de realizar uma simulação formal (interativa ou automática) da especificação do documento, permitindo verificar (de maneira não exaustiva) se a especificação apresenta o comportamento desejado.

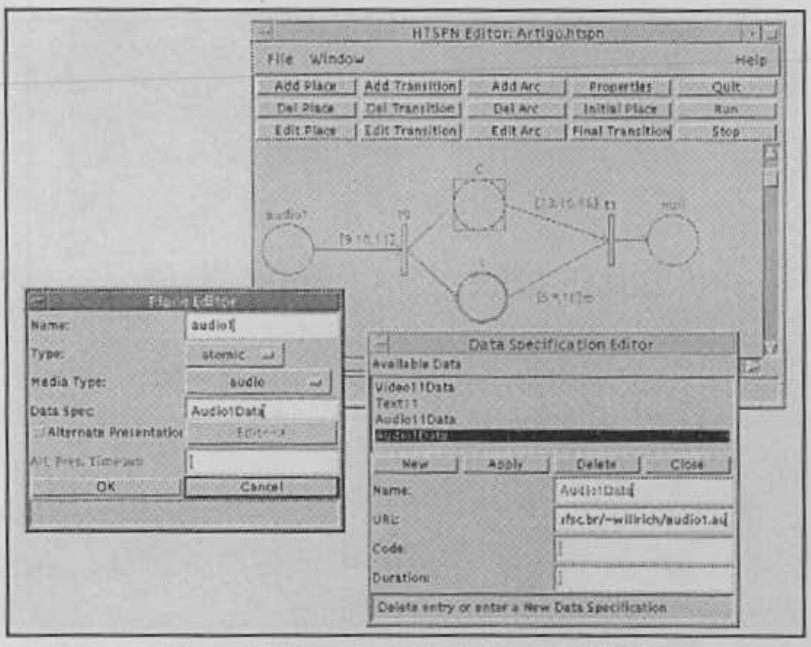

Figura 8. Interface do Editor I-HTSPN

5.3 Verificador

Este módulo é responsável pela análise temporal da rede HTSPN. Como saída, o verificador apresenta os erros. Se riscos de incoerência temporal não são aceitáveis, o autor deve editar novamente a especificação I-HTSPN afim de modificar o esquemas de sincronização.

5.4 Interpretador Java I-HTSPN

O módulo *Interpretador* Java I-HTSPN é uma aplicação Java responsável pela apresentação do documento multimídia interativo a partir de sua especificação Java I-HTSPN. Esta aplicação Java contém um menu a partir do qual um utilizador pode comandar a apresentação. Os seguintes comandos de interpretação são disponíveis: leitura da especificação I-HTSPN do documento gerada pelo módulo Editor; partida e interrupção da interpretação (apresentação) do documento.

Esta aplicação multimídia Java contém a mesmas classes de interpretação apresentadas na seção 4.2. Estas classes geram uma estrutura de interpretação associada a especificação Java I-HTSPN lida. Esta estrutura de interpretação é comandada pelo utilizador via o menu principal do módulo *Interpretador*.

6. Exemplo

Afim de testar o protótipo do ambiente Java I-HTSPN desenvolvido, nós utilizamos este ambiente para o desenvolvimento de um módulo EOC (Ensino Orientado por Computador) VACBI (*Video and Audio Computer Based Instruction*) [VACBI]. Este sistema é utilizado pela *Airbus Training* para a formação de pilotos e agentes de manutenção. O material de ensino VACBI é dividido em capítulos e módulos. Neste

estudo, nós utilizamos unicamente uma versão simplificada do módulo descida normal do trem de aterrissagem do avião Airbus A340. Este módulo foi escolhido por causa de sua representatividade em termos de tamanho e complexidade.

A Figura 9 apresenta a interface do protótipo do módulo de descida normal do trem de aterrissagem implementado a partir do ambiente Java I-HTSPN, ela contém:

- uma animação gráfica da descida do trem de aterrissagem;
- instruções textuais para guiar o estudante e ajudá-lo em caso de operações incorretas;
- informações sonoras, geralmente retomando as instruções textuais;
- animações de imagens (simulando vídeos), mostrando as imagens da abertura das portas e da descida do trem de aterrissagem;
- um conjunto do botões permitindo que o estudante interaja com o módulo.

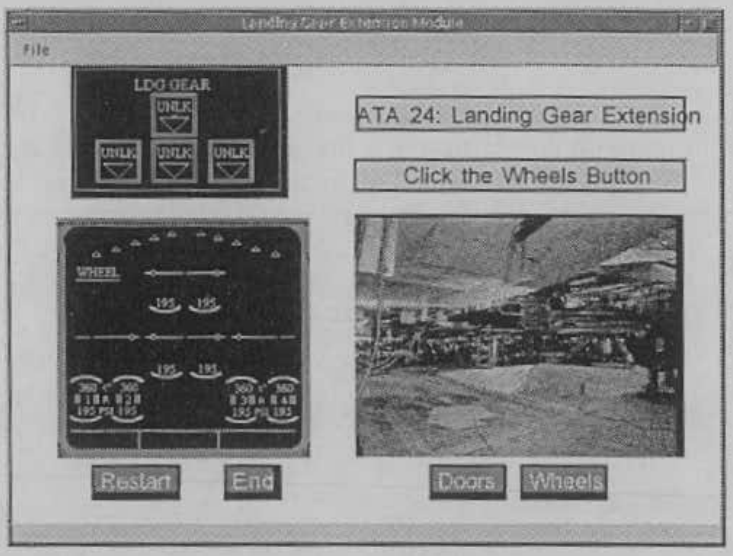

Figura 9. Interface do Módulo VACBI

Em seguida, será apresentada cada um das etapas de desenvolvimento do módulo "Descida normal do trem de aterrissagem.

6.1 Especificação Formal do Módulo

A primeira etapa da metodologia de desenvolvimento do protótipo do módulo descida do trem de aterrissagem é a realização de sua especificação Java I-HTSPN.

Especificação da Estrutura Conceitual

A estrutura conceitual do módulo descida do trem de aterrissagem foi especificada por uma rede HTSPN. A especificação completa deste módulo possui 5 sub-redes, 69 lugares e 61 transições. Duas destas sub-redes são apresentadas nas figuras 10 e 11. Nestas redes, os arcos de saída dos lugares que não possuem IVT associados têm este intervalo igual a [∞,.,∞].

A Figura 10 apresenta a rede raiz da especificação HTSPN. O comportamento é o seguinte: no início, várias imagens ($Pict_i$), textos (Tx_i) e um áudio ($Au1$) são apresentados em paralelo com dois botões. Estes botões permitem ao aluno iniciar o estudo (*Start*) ou encerrar a apresentação (*End*). Note que a ligação *Start* é temporizada, seu intervalo de validade temporal [0,*,20] especifica que esta ligação é seguida automaticamente se o aluno não ativar nenhum botão após 20 segundos do início da apresentação. Se o botão *Start* é ativado, o procedimento de descida do trem de aterrissagem é apresentado (especificado pelo lugar comporto *Procedure*) em paralelo com os botões *Restart* e *End*. Desta forma, durante todo o procedimento de descida do trem, o aluno pode recomeçar o estudo ou encerrá-lo.

A Figura 11 contém a especificação da sub-rede associada ao lugar composto *Procedure* da rede raiz (10th). Esta sub-rede especifica o procedimento de descida do trem de aterrissagem que deve ser comandado pelo aluno. Este procedimento é composto pelas seguintes operações: abertura das portas (lugar *OD - Opening Doors*), descida do trem de aterrissagem (*LGE - Landing Gear Extension*), e fechamento das portas (*lugar CD - Closing Doors*). Para comandar estas operações, o aluno dispõe de um botão *Wheels* para comandar o trem de aterrissagem (descida), e um botão *Doors* para comandar a porta (abertura e fechamento). Informações sonoras (lugares Au_i) e textuais (lugares Tx_i) são apresentadas entre cada fase deste procedimento. Durante a operação de descida do trem de aterrissagem, caso o aluno realizar uma operação incorreta (por exemplo, acionar o botão *Wheels* no lugar de *Doors*), um áudio é apresentado indicando o erro.

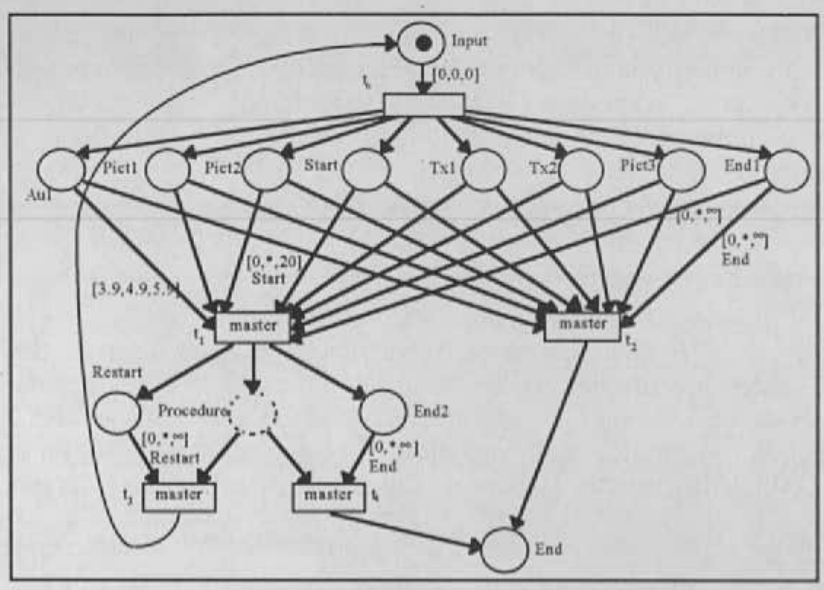

Figura 10 . Raiz da especificação I-HTSPN

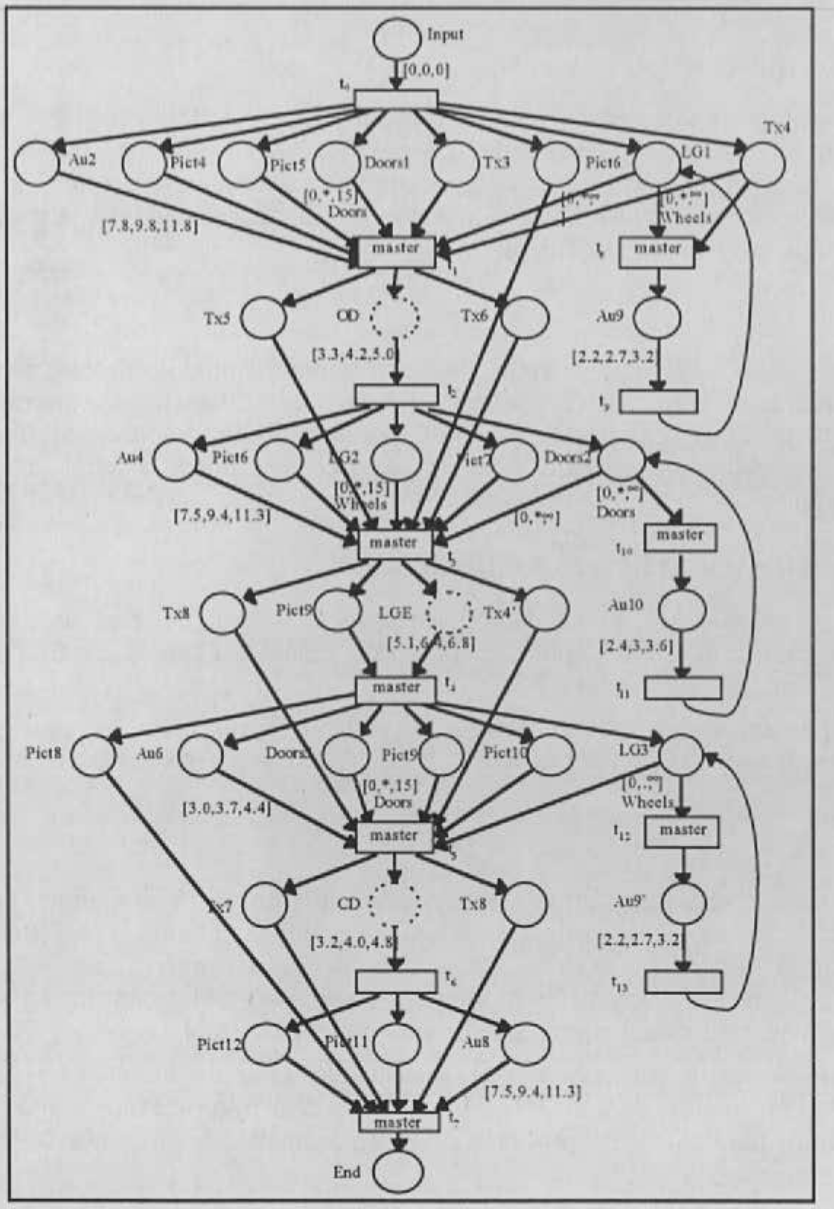

Figura11 . Especificação do procedimento de descida do trem de aterrissagem

A especificação HTSPN descreve os componentes do módulo descida do trem de aterrissagem e o seu comportamento lógico e temporal. Afim de completar esta especificação, o autor deve utilizar a versão interpretada do modelo HTSPN para descrever: os dados multimídia utilizados por este módulo (modelados por um conjunto de objetos *Data*) e as características de apresentação destes componentes (modelados por um conjunto de objetos *Presentation*).

Especificação da Estrutura do Conteúdo

O módulo descida do trem de aterrissagem contém 34 objetos *Data* que descrevem todos os dados utilizados: 2 animações, 10 arquivos de áudio, 9 textos e 13 imagens. Por exemplo, o objeto *PhotoA340* abaixo especifica a imagem que constitui o componente modelado pelo lugar *Pict1*.

```
StillPicture PhotoA340 ::= {
        URL http: www.inf.ufsc.br ~willrich htspn Spec IMG A340.gif
}
```

Especificação da Estrutura de Apresentação

Além da especificação dos dados, o autor deve especificar as características de apresentação dos componentes do módulo. Assim, um objeto *Presentation* deve ser associado a cada lugar do tipo atômico e ligação da especificação HTSPN. Estes objetos descrevem os dados utilizados e as características das apresentações modeladas pelos lugares do tipo atômico, e as características de apresentação dos botões modelados pelos lugares do tipo ligação. Por exemplo, o objeto *Pict1* abaixo é associado ao lugar *Pict1* da rede raiz da especificação HTSPN (Figura 10) para especificar os dados e as características da apresentação modelada por este lugar.

```
PSStillPicture Pict1:: = {
        data                    PhotoA340
        presentation_position   [ 340 175 ]
}
```

6.2 Análise da Especificação

A utilização da metodologia de desenvolvimento proposta por este artigo permite a utilização de técnicas de análise do comportamento lógico e temporal da especificação I-HTSPN. Esta fase de análise permite a obtenção de uma especificação precisa do comportamento do módulo descida do trem de aterrissagem. Quando do projeto de aplicações multimídia interativas complexas, a metodologia proposta é capaz de reduzir de maneira significativa os erros de projeto.

6.3 Interpretação

Utilizando o ambiente Java I-HTSPN, o autor pode apresentar o módulo descida do trem de aterrissagem a partir de sua especificação I-HTSPN. A etapa de interpretação da especificação permite ao autor realizar um teste final de seu trabalho a partir da apresentação efetiva do módulo em desenvolvimento. Se o aspecto final ou o comportamento desta apresentação não são aqueles desejados, o autor deve retornar à etapa de especificação do módulo.

6.4 Geração Automática da Aplicação Multimídia Java

Após a obtenção de uma especificação validada do módulo de descida do trem de aterrissagem, o autor pode gerar uma aplicação multimídia Java implementando este módulo. Esta operação pode ser requisitada pelo autor via o menu principal do Editor Java I-HTSPN.

A aplicação Java implementando o módulo deste EOC pode ser apresentado por qualquer sistema ou máquina dispondo de uma máquina virtual Java e um acesso Internet para o acesso aos dados multimídia.

7. Conclusão

Este artigo propôs uma abordagem formal de desenvolvimento de documentos multimídia interativos e distribuídos. Nesta abordagem, a especificação e a análise do documento são feitas com a ajuda de uma extensão do modelo HTSPN (*Hierarchical Time Stream Petri Nets*) [Sénac, 95] e a implementação é suportada pela produção de uma aplicação multimídia Java. Esta aplicação Java pode ser interpretada em qualquer máquina ou sistema operacional contendo uma máquina virtual Java.

Este artigo apresentou também um ambiente Java, chamado de ambiente Java I-HTSPN, implementando a metodologia de desenvolvimento de aplicações multimídia Java proposta no artigo. Esse ambiente contém um conjunto de módulos para ajudar no processo de criação, análise e apresentação de aplicações multimídia Java..

Graças a sua abordagem orientada-à-objetos, seu aspecto multi-tarefa e suas bibliotecas gráficas e de rede, a linguagem Java simplificou a tarefa de implementação do ambiente Java I-HTSPN. O principal interesse da utilização desta linguagem é sua portabilidade. Uma aplicação multimídia desenvolvida em Java pode acessar componentes estocados em qualquer lugar da rede Internet e ela pode ser transferida e apresentada sobre vários equipamentos e sistemas operacionais. Infelizmente, o fraco desempenho da versão do Kit de desenvolvimento Java utilizado (1.0.2) não permite o controle temporal estrito das aplicações multimídia (por exemplo, a notificação de que o tempo máximo de apresentação foi atingido pode sofrer atrasos

consideráveis se a carga do sistema é grande). Além disso, este fraco desempenho induz atrasos consideráveis entre o pedido de apresentação de uma informação e sua apresentação efetiva.

Este artigo apresentou também um exemplo de desenvolvimento de um módulo de ensino orientado por computador utilizando o ambiente Java I-HTSPN. Este exercício serviu para demonstrar que o modelo I-HTSPN permite de descrever eficazmente documentos multimídia complexos e em seguida de permitir a apresentação concreta deste documento.

O modelo I-HTSPN proposto neste artigo tende a ser de muito baixo nível para sua utilização por um autor de um documento, que em geral é alguém que não está habituado com o formalismo redes de Petri. Assim, um estudo poderá ser levado afim de descrever uma interface gráfica de mais alto nível, afim de simplificar a tarefa de concepção de documentos multimídia e permitindo em seguida a produção de redes I-HTSPN correspondentes.

Agradecimentos

Patrice Torguet contribuiu na concepção do editor Java I-HTSPN durante sua estadia no ENSICA.

8. Bibliografia

[Diaz, 93] M. Diaz, P. Sénac. Time Streams Petri Nets, a Model for Multimedia Streams Synchronization. Nos anais do *First International Conference on Multi-media Modelling* - Vol. 1, Singapore, World Scientific, Tat-Seng Chua & T. L. Kunii (Eds.), pp. 257-273, November 1993.

[Halasz, 94] F. Halasz, M. Schwartz. The Dexter Hypertext Reference Model. Communication of the ACM 37(2):29-39, February 1994.

[Hardman, 94] L. Hardman, D.C.A Bulterman, G Van Rossum. The Amsterdam Hypermedia Model: Adding Time, Structure and Context to Hypertext. Communication of the ACM 37(20):50-62. February 1994.

[Hudson, 93] S.E. Hudson, C.H. Hsi. A Framework for Low Level Analysis and Synthesis to Support High Level Authoring of Multimedia Documents. Graphics Visualization and Usability Center Technical Report GVU-93-14. Georgia Institute of Technology, 1993.

[ISO 13522] Committee Draft ISSO/IEC 13522-1 Information technology - Coding of Multimedia and Hypermedia - Part 1: MHEG Object Representation - Base Notation (ASN.1), October, 1994.

[Little, 90] T.C.D. Little, A. Ghafoor. Synchronization and Storage Models for Multimedia Objects. IEEE Journal on Selected Areas in Communication 8(3):413-427, April 1990.

[Murata, 89] T. Murata. Petri Nets: Properties, Analysis and Applications. IEEE 77(4):541-580. April, 1989.

[Sun, 95] Sun Microsystems. The Java Language Specification, 1995.

[Sénac, 95] P. Sénac, R. Willrich, P. de Saqui-Sannes. Hierarchical Time Stream Petri Nets: A Model for Hypermedia Systems. 16th International Conference on Application and Theory of Petri Nets. In Application and Theory of Petri Nets 1995, Lecture Notes in Computer Science no. 935, G. De Michelis and M. Diaz (Eds.), Springer, pp. 451-470.

[Sénac, 96] P. Sénac, M. Diaz, A Léger, P. de Saqui-Sannes. Modeling Logical and Temporal Synchronization in Hypermedia Systems. IEEE Journal on Selected Areas in Communication 14(1):84-103, 1996.

[VACBI] VACBI Utilization Guide. Aeroformation - Airbus Industrie/Flight Safety.

[Willrich, 96a] R. Willrich, P. Sénac, M. Diaz, P. de Saqui-Sannes. A Formal Framework for the Specification, Analysis and Generation of Standardized Hypermedia Documents. Third IEEE International Conference on Multimedia Computing and Systems (ICMCS'96), pp. 399-406, Hiroshima, Japão, 1996.

[Willrich, 96b] R. Willrich, P. Sénac, P. de Saqui-Sannes, M. Diaz. Towards Hypermedia Documents Design. 14o Simpósio Brasileiro de Redes de Computadores (SBRC'96), pp. 473-491. Fortaleza, 1996.

[Willrich, 96c] R. Willrich. Conception Formelle de Documents Hypermédias Portables. Tese apresentada no Laboratoire d'Analyse et d'Architecture des Systèmes du C.N.R.S. em vista de obtenção do título de Docteur pela Universidade Paul Sabatier. Toulouse (França), Setembro de 1996.