UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Adriana Dallacosta

Modelo SMIL I-HTSPN: Especificação, Análise e Geração de Código SMIL Boston

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Roberto Willrich

Florianópolis, setembro de 2000.

Modelo SMIL I-HTSPN: especificação, análise e geração de código SMIL Boston

Adriana Dallacosta

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação Área de Concentração Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Roberto Willrich

Fernando Álvaro Ostuni Gauthier

Banca Examinadora

Roberto Willrich

Rosvelter Coelho da Costa

Vitório Bruno Mazzola

Com muito amor e carinho aos meus queridos pais Nadir João Dallacosta e Lourdes Morales Dallacosta.
Ao meu noivo Gustavo Zanini Kantoski fonte de amor e inspiração.

Agradecimentos

Ao professor Dr. Roberto Willrich pela amizade, compreensão, apoio, paciência, dedicação e orientação que permitiu a busca de conhecimentos.

À banca examinadora, Prof. Rosvelter Coelho da Costa, Prof.Vitório Bruno Mazzola e Prof. Murilo Silva de Camargo pela dedicação na análise e enriquecimento deste trabalho.

Ao Programa de Pós-graduação da Universidade Federal de Santa Catarina, pela oportunidade de realização do mestrado.

Ao corpo docente do Departamento de Ciência da Computação, pela troca de experiências e idéias que contribuíram para o aprimoramento profissional e pessoal.

Ao Colégio Militar de Santa Maria pelo apoio e oportunidade concedida nesta busca de aperfeiçoamento.

À direção, professores, funcionários, colegas e amigos do Colégio Militar de Santa Maria, pela motivação, incentivo e colaboração dedicados.

Em especial, as amigas que me apoiaram e estimularam, as que comigo estudaram, discutiram e riram, enfim, as que fizeram parte da minha história de vida.

Aos meus pais Nadir e Lourdes pelo incentivo que sempre me deram aos estudos, pelo amor e dedicação.

Ao meu noivo, Gustavo Zanini Kantorsk, por ter sido incentivador, companheiro, conselheiro, pela ajuda em cada etapa desse sonho. Por existir em minha vida e iluminar o meu caminho.

A Deus, pela luz que ilumina os meus atos.

E a todas as pessoas que de alguma forma contribuiram para a realização deste trabalho.

Sumário

Lista de Figuras

Lista de Abreviaturas

DTD	*Data Type Document*
EBNF	*Extended Backus – Naur Form*
GRiNS	*Graphical Interface to SMIL*
HTML	*Hypertext Markup Language*
HTSPN	*Hierarchical Time Streams Petri Net*
I-HTSPN	*Interpreted Hierarchical Time Streams Petri Net*
RTP	*Real Time Transport Protocol*
SGML	*Standard Generalized Markup Language*
SMIL	*Synchronized Multimedia Integration Language*
SYMM	*Synchronized Multimedia*
TSPN	*Time Streams Petri Net*
URI	*Universal Resource Identifier*
URL	*Universal Resource Location*
XML	*Extensible Markup Language*
W3C	*World-Wide Consortium*
WG	*Working Group*
WWW	*World Wide Web*

Resumo

A linguagem SMIL (*Synchronized Multimedia Integration Language*) tem sido projetada de maneira a facilitar a autoria de apresentações multimídia na Web. Com um editor de texto ela permite definir os componentes de documentos multimídia como texto, imagens, áudio e vídeo na forma de URIs (*Universal Resource Identifier*) e escalona suas apresentações em paralelo ou em seqüência. A desvantagem é que a manipulação da noção de tempo na forma textual é difícil de manipular e modificar. Ao contrário, o modelo I-HTSPN (*Interpreted Hierarchical Time Streams Petri Net*) facilita a autoria de apresentações multimídia por ser um sistema gráfico e, além disso, por ser um método formal, permite realizar uma análise do documento, a fim de verificar inconsistências temporais. Esta dissertação tem por objetivo definir um modelo formal gráfico de mais alto nível, que permita a especificação e análise do documento, para posterior tradução para o código SMIL Boston. A especificação gráfica de documentos SMIL será obtida graças a uma nova interpretação do modelo I-HTSPN voltada para a especificação, análise e geração de código SMIL Boston. Esta nova interpretação se chama SMIL I-HTSPN e herdará todas as técnicas de análise do HTSPN.

Abstract

Language SMIL (Synchronized Multimedia Integration Language) has been projected in way to facilitate the authorship of presentations multimedia in the Web. With a text publisher it allows to define the document components multimedia as text, pictures, audio and video in the form of URIs (Universal Resource Identifier) and schedules its presentations in parallel or sequence. The disadvantage is that the manipulation of the notion of time in the literal form is difficult to manipulate and to modify. In contrast, model I-HTSPN (Interpreted Hierarchical Time Streams Petri Net) facilitates the authorship of presentations multimedia for being a graphical system and, moreover, for being a formal method, it allows to carry through an analysis of the document, in order to verify secular inconsistencies. This dissertation has for objective to define a graphical formal model of higher level, that allows to the specification and analysis of the document, for further translation for code SMIL Boston. The graphical document specification SMIL will be acquired thanks to a new interpretation of model I-HTSPN directed to the specification, analysis and generation of code SMIL Boston. This new interpretation is called SMIL I-HTSPN and will inherit all the techniques of analysis of the HTSPN.

Capítulo 1. Introdução

Multimídia é um dos termos mais usados nos últimos anos. Este termo está no cruzamento de seis grandes indústrias [Fluckiger, 95]: informática, telecomunicações, publicidade, consumidores de dispositivos de áudio e vídeo, indústria de televisão e cinema. Este grande interesse destas indústrias poderosas contribuiu para a grande evolução da multimídia.

A motivação do uso de dados multimídia em sistemas computacionais é o aumento da transferência de informações pelo uso simultâneo de um ou mais sensos do usuário. Isto, pois humanos aprendem mais, e mais rapidamente, quando vários dos seus sentidos (visão, audição, etc.) são utilizados.

Por outro lado, a Web é um meio interessante de distribuição de informações na Internet. Através dela, as informações sobre os mais variados assuntos se tornam de fácil acesso e de rápida manipulação.

No entanto, o HTML (*Hypertext Markup Language*), linguagem amplamente utilizada como formato para hiperdocumentos disponibilizados na WWW (*World Wide Web*), apesar de simples e de fácil uso, não apresenta suporte para definição de relacionamentos de sincronização entre mídias. Para isto é necessário que se faça uso de uma integração com outras linguagens de programação, tais como *Java* e *JavaScript*. A desvantagem dessa abordagem é que essas são linguagens de propósito genérico, apresentando uma alta complexidade para autores que não possuem conhecimentos em programação [Rodrigues, 99].

Uma alternativa é utilizar uma linguagem declarativa específica para autoria de documentos multimídia, diferente do HTML, mas seguindo seu mesmo objetivo de simplicidade. Essa proposta foi adotada como recomendação pelo W3C através da linguagem SMIL (*Synchronized Multimedia Integration Language*) [W3C 98].

A idéia básica da linguagem SMIL é nomear componentes do documento como texto, imagens, áudio e vídeo como URIs (*Uniform Resource Identification* - um termo mais genérico que URL usado no momento na Web) e escalonar suas apresentações em

paralelo ou em seqüência. Esta linguagem tem sido projetada de maneira a facilitar a autoria das apresentações com um editor de texto. A desvantagem é que a manipulação da noção de tempo na forma textual é difícil de manipular e modificar.

Ao contrário, o modelo I-HTSPN (*Interpreted Hierarchical Time Streams Petri Net*), além de permitir uma especificação completa de documentos multimídia a partir da especificação das estruturas conceptuais, de apresentação e de conteúdo, facilita a autoria de apresentações multimídia, por ser um sistema gráfico e, além disso, por ser um método formal que permite realizar uma análise do documento a fim de verificar inconsistências temporais.

Essa dissertação tem por objetivo definir um modelo gráfico de mais alto nível, que permita a especificação gráfica e análise do documento, para posterior tradução para o código SMIL Boston. A especificação gráfica de documentos SMIL será obtida graças a uma nova interpretação do modelo I-HTSPN, voltada para a especificação, análise e geração de código SMIL Boston. Esta nova interpretação se chama SMIL I-HTSPN.

Além deste capítulo introdutório, o trabalho apresenta-se constituído de sete capítulos.

O capítulo 2 trata da problemática de criação de documentos multimídia. Será apresentado o que são documentos multimídia e como realizar a criação desses documentos. Serão vistos os requisitos para um modelo multimídia e serão apresentadas as abordagens para autoria de documentos multimídia. O capítulo 3 apresenta uma introdução à linguagem SMIL. O capítulo 4 apresenta os modelos HTSPN, I-HTSPN e *Java* I-HTSPN. O capítulo 5 apresenta um comparativo da linguagem SMIL Boston x I-HTSPN. O capítulo 6 define o modelo SMIL I-HTSPN, interpretação do modelo I-HTSPN voltada para a especificação, análise e geração de código SMIL Boston. As conclusões e os trabalhos futuros são abordados no capítulo 7.

Capítulo 2. Modelos de Autoria - Requisitos

Este capítulo é consagrado ao estudo da problemática de criação de documentos multimídia. Um sistema multimídia pode ser definido como um sistema capaz de manipular ao menos um tipo de mídia discreta (texto, imagem, ...) e um tipo de mídia contínua (vídeo, áudio, ...), as duas numa forma digital. Neste trabalho, considera-se que um sistema hipermídia é um sistema multimídia no qual as informações monomídia e multimídia são acessadas e apresentadas com a ajuda de mecanismos de navegação baseados em ligações (*Links*).

Documentos multimídia/hipermídia existem em várias áreas e níveis. Tipicamente estas aplicações multimídia são aplicadas nas áreas de educação, treinamento profissional, quiosque de informações públicas, ou mercado varejista. Um exemplo de uma aplicação ou documento multimídia é um jornal de biologia que inclui a visualização de moléculas, funcionalidades para acessar e buscar artigos de pesquisas associados, funcionalidades de comunicação colaborativa, impressão, anotação, ou animação.

Em muitos trabalhos não há a distinção entre documentos multimídia e hipermídia. Neste caso o termo documento multimídia engloba os documentos multimídia da forma apresentada na seção 2.1.2 e os documentos hipermídia da forma apresentada na seção 2.1.3. Neste trabalho, por simplificação, utiliza-se o termo documento multimídia para especificar os documentos multimídia e hipermídia.

2.1 Documento hipertexto, multimídia e hipermídia

2.1.1 Documentos Hipertexto

Um documento hipertexto é uma estrutura de informação organizada de maneira não linear: os dados são armazenados em uma rede de nós conectada por ligações ou *links* (Figura 1):

- os **nós** contêm as unidades de informação compostas por texto e outras informações gráficas. Em geral, um nó representa um conceito ou uma idéia expressa de uma maneira textual ou gráfica;
- os ***links*** definem as relações lógicas (ou semânticas) entre os nós, isto é, eles definem relações entre conceitos e idéias. A noção de âncora permite a especificação de uma parte da informação que será fonte ou destino de um *link*. Uma âncora hipertexto é descrita por uma seqüência de caracteres ou o nó inteiro. A Figura 1 apresenta uma correspondência entre as informações e as âncoras na tela, e os nós, os *links* e as âncoras na base de dados hipertexto.

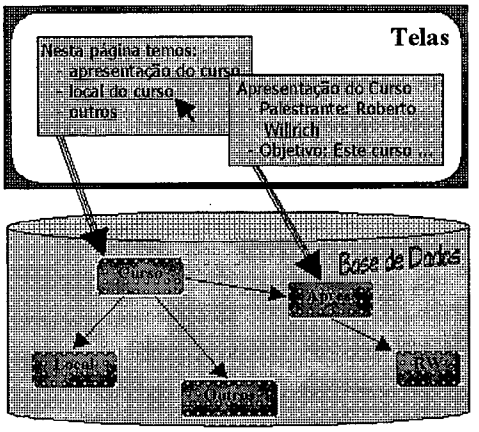


Figura 1. Base de dados hipertexto

Em geral, quando o usuário de uma aplicação hipertexto clica sobre uma âncora, o *link* é disparado (seguido), causando a apresentação da *âncora* destino. Dessa forma, o usuário pode "navegar" no documento através das âncoras e dos *links*.

Como os textos e imagens são informações estáticas (as informações não evoluem no tempo), a noção de tempo não é utilizada quando da especificação de documentos hipertextos clássicos. No hipertexto, o tempo de apresentação é determinado pelo leitor do documento.

2.1.2 Documentos Multimídia

Um documento multimídia pode ser visto como uma estrutura descrevendo a coordenação e o estilo de apresentação de uma coleção de componentes constituídos de mídias estáticas e dinâmicas. Mídias estáticas são aquelas com dimensões unicamente

espaciais (imagens e textos) e as dinâmicas, com dimensões temporais (como sons e vídeos).

Um exemplo de documento multimídia real pode ser a apresentação dos pontos turísticos de uma cidade, constituído de textos, imagens e vídeos de cada ponto turístico, apresentados em uma ordem seqüencial e possivelmente com alguns mecanismos de interação.

Quando o autor cria um documento multimídia, ele define a **orquestração** da apresentação dos componentes a partir da definição dos instantes de início e fim das apresentações e das relações condicionais e temporais entre e no interior dos componentes. Esta descrição da ordem temporal relativa de apresentação dos componentes é chamada de **cenário multimídia**. A Figura 2 apresenta um pequeno documento multimídia.

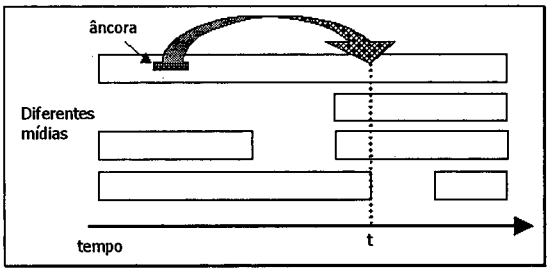


Figura 2. Documentos Multimídia [Hardman, 94]

Documentos multimídia podem ser interativos. Tipicamente, existem dois métodos de interação a partir dos quais o leitor pode controlar a apresentação [Hardman, 94]:

- um método de controle de apresentação que permite o ajuste da referência temporal utilizada pela apresentação. Com este mecanismo o leitor pode interromper, repartir, fazer avanços ou retornos rápidos na apresentação do documento; isto a partir de uma interface similar aos controles de um videocassete;
- um método de interação similar a um *link* hipertexto rudimentar definido por uma âncora e um *link*. Seguindo o *link*, a apresentação salta para uma outra parte do documento. Este método é ilustrado na Figura 2: quando o leitor clica sobre a âncora, a apresentação salta para o ponto t.

2.1.3 *Documentos Hipermídia*

Um documento hipermídia é uma combinação de documentos hipertexto e multimídia (ilustrado na Figura 3). Esta classe de aplicação representa uma evolução natural do hipertexto, na qual as noções ou conceitos de nós hipertexto podem agora ser expressos por diferentes tipos de mídia de apresentação. A inclusão de dados multimídia aumenta o poder de expressão da informação contida em uma aplicação e torna a apresentação mais atrativa e realista.

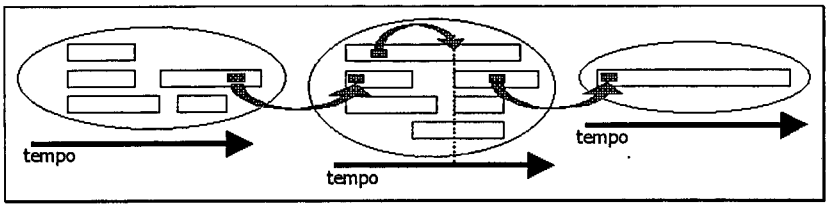


Figura 3. Documentos Hipermídia

Como nos documentos hipertextos, as âncoras permitem a especificação de uma parte da apresentação de uma mídia que será fonte ou destino de um *link* hipermídia. Por exemplo, uma âncora pode ser uma seqüência de quadros de um vídeo ou uma região de uma imagem.

Os *links* hipermídia podem ser temporizados. Este tipo de *link* é habilitado durante um determinado intervalo temporal e pode ser ativado automaticamente em função das suas restrições temporais. Por exemplo, um *link* temporizado pode levar a uma página de ajuda caso o leitor não ative nenhum outro *link* em 30 segundos. Este tipo de *link* introduz a noção de sistemas hipermídia ativos [Stotts, 90].

A inclusão de dados multimídia em estruturas hipertexto introduz um aspecto temporal na especificação de aplicações hipermídia. Dessa forma, é necessário que os modelos de especificação hipermídia forneçam mecanismos que permitam a integração temporal de uma variedade de mídias contínuas e discretas. Além disto, ele deve fornecer mecanismos de descrição das possíveis interações com o usuário do documento, isto é, ele deve permitir a descrição dos *links* hipermídia, extensão dos *links* hipertextos (com um caráter temporal), e outros métodos de interação.

A maior parte da literatura da área não sugere esta distinção entre documentos multimídia e hipermídia. Por simplificação neste trabalho o termo documento multimídia englobará tanto documento multimídia quanto hipermídia.

2.2 Autoria de Documentos Multimídia

Ambientes de desenvolvimento multimídia facilitam e automatizam a autoria (criação) de documentos multimídia. Há uma grande variedade de tais ambientes, também chamados de sistemas de autoria. Estes sistemas de autoria são projetados para fornecer ferramentas de criação e organização de uma variedade de elementos de mídia como texto, gráficos, imagens, animações, áudio e vídeo, a fim de produzir documentos multimídia. Os usuários deste tipo software, os autores dos documentos, são profissionais que desenvolvem apresentações educacionais, de *marketing* e artistas gráficos que fazem decisões acerca do *layout* gráfico e estilo de interação que o usuário final real (estudantes ou indivíduos no público) vêem e ouvem [Buford, 94].

Sistemas de autoria geralmente fornecem meios para gerar certos tipos de interações comuns através do uso de *templates* reutilizáveis ou módulos de software, que podem ser personalizados pelo projetor para um objetivo particular. Como exemplo, muitos programas de treinamento usam um formato de questão múltipla escolha.

2.3 Requisitos para um Modelo Multimídia

Ferramentas de autoria de documentos multimídia são baseadas em um modelo multimídia que induz uma técnica de descrição de documentos multimídia. Baseado na ISO ODA (*Office Document Architecture*) [ISO 8613] várias abordagens dividem a descrição de documentos em três partes:

- a **estrutura lógica**, que descreve as diferentes partes lógicas de um documento (ou componentes) e suas relações lógicas;
- a **estrutura de apresentação**, que descreve como, onde e quando os diferentes componentes serão apresentados;
- a **estrutura de conteúdo** descreve as informações que constituem os componentes.

[Fluckiger, 95] afirma que a partição da descrição do documento nestas três estruturas oferece várias vantagens. A separação entre a estrutura lógica e de apresentação, por exemplo, permite a reutilização da estrutura lógica quando da apresentação do documento em dispositivos diferentes, onde somente uma modificação

simples ou adaptação da estrutura de apresentação deve ser realizada; e a separação entre a estrutura de apresentação e do conteúdo é necessária, pois os documentos podem utilizar os mesmos dados em diferentes contextos [Schloss, 94] ou por diferentes documentos.

A separação entre a semântica pura (a estrutura lógica) e a definição dos instantes de aparição dos componentes (o quando, definido pela estrutura de apresentação) é contestada por vários especialistas. Isto, pois um documento multimídia não é somente aquilo que ele contém, mas também quando seus conteúdos são apresentados. Assim, baseado na arquitetura proposta por [Klas, 90], um modelo multimídia ideal deveria permitir uma descrição multinível incluindo as estruturas seguintes:

- a **estrutura conceitual**, que descreve as diferentes partes lógicas do documento ou componentes, suas relações lógicas, e em que instante estes componentes serão apresentados;
- a **estrutura de apresentação**, que descreve como e onde os diferentes componentes serão apresentados;
- a **estrutura de conteúdo**, que descreve as informações que constituem os componentes.

A única diferença entre esta classificação e a primeira é que o "quando" da apresentação dos componentes é transferido da estrutura de apresentação para a estrutura lógica. Assim, contrariamente à afirmação de [Fluckiger, 95], quando da apresentação do documento em diferentes sistemas-clientes, a estrutura lógica e o comportamento temporal (a estrutura conceitual) podem ser reutilizados. As apresentações (a estrutura de apresentação e do conteúdo) não suportadas pelo sistema cliente devem ser adaptadas e o "quando" da apresentação destes componentes deve ser respeitado. Por exemplo, se um componente é constituído de uma apresentação de áudio e se o sistema-cliente não suporta a apresentação deste tipo de mídia, o conteúdo deste componente deve ser substituído por uma informação textual. Mas os comportamentos temporais destas apresentações devem ser equivalentes.

Além da descrição dessas três estruturas, um modelo para essas aplicações deve permitir também a especificação dos possíveis métodos de interação com o usuário.

Dessa forma, um modelo deve permitir a definição de *âncoras*, de *links* hipermídia e de outros métodos de interação de documentos multimídia.

2.3.1 Estrutura Conceitual

A estrutura conceitual especifica os componentes e os grupos de componentes de um documento, e a composição lógica e temporal destes componentes.

Os componentes e grupos de componentes

A estrutura conceitual contém as definições dos componentes semânticos de um documento, permitindo a divisão do documento em, por exemplo, capítulos e parágrafos, ou uma série de seqüências de vídeo e de títulos. Um exemplo desta estruturação é apresentado na Figura 4.

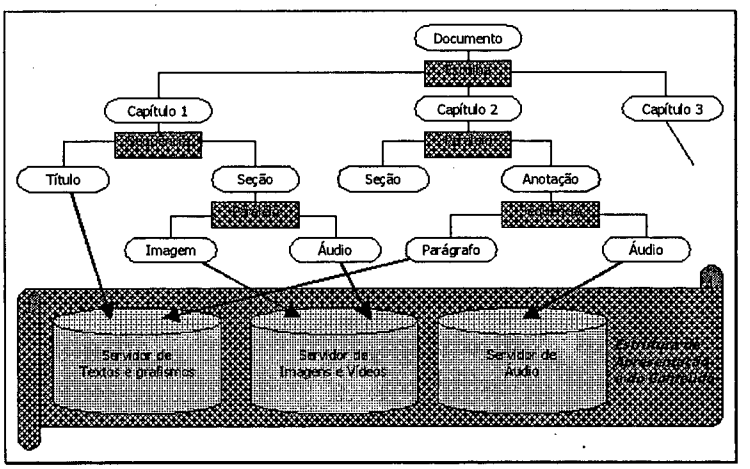


Figura 4. Componentes e grupos de componentes

A tarefa de especificação de documentos multimídia se torna delicada e complexa com o aumento do tamanho do documento. A estrutura conceitual é utilizada para a construção de apresentações complexas a partir de pequenos grupos. Assim, a descrição do documento pode ser dividida em seções, cada seção podendo ser criada de maneira independente das outras seções [Hardman, 95]: as apresentações que exprimem juntas uma idéia ou um conceito podem ser agrupadas para posterior reutilização. Além disso, estes mecanismos de estruturação permitem a composição do documento a partir de técnicas *top-down* e/ou *bottom-up*. Eles permitem também a definição de relações lógicas e temporais entre estes grupos e a definição de restrições temporais associadas a um grupo de apresentações. Concluindo, a estrutura conceitual introduz o benefício da modularidade, da encapsulação e mecanismos de abstração.

Independência dos componentes e seus conteúdos

A definição do componente do documento deve ser realizada independentemente das informações acerca da representação do seu conteúdo [Klas, 90], que são descritas pelas estruturas de apresentação e do conteúdo (como ilustrado na Figura 4). Assim, os componentes da estrutura conceitual devem fazer unicamente menção às informações que estes componentes representam. A fim de simplificar a especificação do documento, esta menção deve ser independente do tipo de informação (mídia ou não-mídia) ou de sua localização geográfica (local ou remota).

Quando do armazenamento e da transmissão do documento, certas partes da estrutura conceitual e/ou do conteúdo dos componentes podem ser incluídas explicitamente ou implicitamente na estrutura do documento:

- **inclusão explícita**: a estrutura do documento contém os dados primitivos. Assim, a estrutura do documento e os conteúdos incluídos explicitamente são armazenados no mesmo arquivo e transferidos conjuntamente;
- **inclusão implícita**: a estrutura do documento faz referência aos dados primitivos ou outras partes da estrutura conceitual. Assim, no modo de transmissão em tempo real e de telecarga somente uma parte do documento é transferida. Quando da apresentação do documento o sistema-cliente deve acessar de maneira remota as partes incluídas implicitamente no documento.

Os caminhos de percurso e as estruturas de Informação

Estrutura conceitual define os caminhos de percurso do documento. Estes caminhos de percurso são definidos pelos *links*. Fundamentalmente, existem dois tipos de *links* [Shackelford, 93]:

- ***link* estrutural,** que fornece uma estrutura de base de um documento. Quando o usuário segue esses *links*, o discurso (ou a estrutura) de base é preservado como ele foi criado pelo autor do documento;
- ***link* hiperestrutural,** que permite a definição das relações que transpassam a estrutura de base do documento. Estes *links* podem ser divididos em *links* associativos e referenciais [Ginige, 95]. Os ***links* associativos** conectam conceitos associados e os ***links* referenciais** conectam informações adicionais a um conceito.

Composição temporal do documento

A estrutura conceitual define também a estrutura temporal de uma aplicação, que consiste na descrição dos instantes de partida e parada das apresentações dos componentes e de suas relações temporais e condicionais (ou causais). Essas relações são estabelecidas por eventos definidos no interior de diferentes apresentações. Existem dois tipos de eventos que podem ocorrer durante uma apresentação:

- os **eventos síncronos** (previsíveis): são os eventos cujas posições no tempo são determinadas previamente (e.g., início da apresentação de uma seqüência de áudio ou vídeo). A posição destes eventos podem ser determinada somente sob condições ideais (sem considerar os atrasos imprevisíveis, como sobrecarga na rede);
- os **eventos assíncronos** (imprevisíveis): são os eventos cujas posições no tempo não podem ser determinadas previamente. Como o instante em que a aplicação chega a um determinado estado ou a interação do usuário.

As Relações Temporais

As relações temporais definem as posições temporais relativas entre e no interior dos componentes de um documento. Estas relações temporais definem sincronizações que podem ser definidas entre eventos definidos dentro de uma apresentação ou entre eventos definidos em apresentações distintas:

- **sincronização intramídia**: são as relações temporais entre os eventos ou intervalos definidos no interior de uma mídia contínua. Por exemplo, definidos entre os quadros de uma seqüência de vídeo;
- **sincronização intermídia**: são as relações temporais entre os eventos ou intervalos definidos em diferentes apresentações (mídias).

Em geral, as sincronizações intramídia são dependências temporais naturais que são definidas implicitamente quando da produção dos dados primitivos (na estrutura do conteúdo). O esquema de sincronização intramídia pode ser alterado pelo autor do documento quando da definição da estrutura de apresentação. Por exemplo, a sincronização entre dois quadros de uma seqüência de vídeo pode ser definida pela velocidade de apresentação desta seqüência. As sincronizações intermídia são geralmente dependências temporais artificiais especificadas explicitamente pelo autor da estrutura conceitual de um documento. Assim, em geral as sincronizações intermídia

são descritas pela estrutura conceitual do documento, e as sincronizações intramídia fazem parte da estrutura de apresentação.

As relações temporais de um documento podem ser expressas de forma estática, que consiste em organizar explicitamente as relações temporais entre os componentes (*orquestração*); ou de forma dinâmica, em que as relações temporais denominadas relações *ao vivo* são definidas [Fluckiger, 95]. Um exemplo de relação ao vivo é a sincronização de um trabalho cooperativo. Um modelo multimídia trata unicamente da orquestração, já que as relações ao vivo não podem ser especificadas em avanço, no momento da criação do documento. Assim, a estrutura conceitual define apenas os aspectos de orquestração (sincronização estática) da apresentação multimídia.

Modelo Temporal

Para a descrição das relações temporais, um modelo de composição requer um **modelo temporal**. Com relação a sua unidade elementar, duas classes de modelos temporais podem ser identificados [Wahl, 94]:

- **modelos temporais baseados em pontos**: são os modelos cuja unidade temporal são os eventos. Existem três relações temporais que podem ser definidas entre dois eventos: um evento pode ocorrer antes, em simultâneo e após outro evento;
- **modelos temporais baseados em intervalos**: são os modelos cuja unidade temporal são os intervalos. [Hamblin, 72] e [Allen, 83] definem que existem 13 relações temporais possíveis entre dois intervalos (Figura 5): antes, precede, após, durante, começa, acaba e igual, aos quais se ajuntam as relações inversas (com exceção da relação igual).

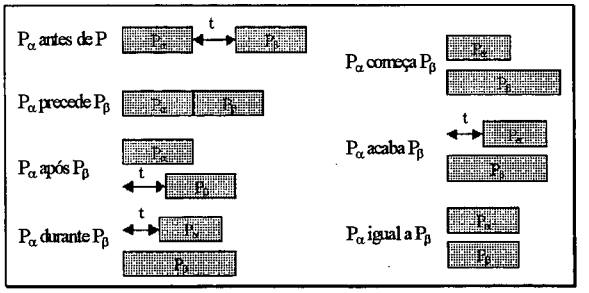


Figura 5. Relações temporais básicas entre intervalos

Para a multimídia, modelos temporais baseados em intervalos são melhores que aqueles baseados em pontos. Isto, pois em mais alto nível de abstração, as apresentações

podem ser vistas como intervalos temporais, com um início, fim e uma duração. Isto simplifica a especificação das relações temporais e condicionais.

Relações condicionais

Uma relação condicional é definida como uma condição associada a um conjunto de componentes, e ações que serão aplicadas a um conjunto de componentes quando esta condição é satisfeita. Por exemplo, "após o término da apresentação de A, se o *link* L está ativado, então apresentar B" é uma relação condicional.

Um modelo hipermídia ideal deveria fornecer mecanismos para especificar este tipo de relação.

Sincronização em sistemas distribuídos

Por causa do não determinismo da duração de tratamento das informações em sistemas multimídia distribuídos (causado pela rede de comunicação, atrasos de acesso a base de dados, etc.), as relações temporais desejadas podem não ser sempre garantidas. É por isso que um modelo multimídia deve permitir a especificação de **métodos de tolerância** de sincronização [Wynblatt, 95]. Assim, o autor pode expressar quais compromissos de sincronização são aceitáveis e os meios de tratar as exceções quando da violação.

2.3.2 Estrutura de Apresentação

A estrutura de apresentação de um documento descreve as características espaciais, sonoras e temporais de cada apresentação que compõem o documento. A este nível, o autor deve especificar as características de apresentação de cada componente e a composição espacial destas apresentações em um instante dado. A especificação das características de apresentação inclui a descrição da maneira como o componente será visto (descrição das características espaciais) e/ou ouvido (descrição das características sonoras) pelo leitor do documento. A composição espacial descreve também as relações espaciais entre as apresentações.

Especificação das características de Apresentação

A estrutura conceitual define os componentes do documento e suas relações. A partir disto, o autor deve definir o conteúdo dos componentes (isto é, associar um objeto

multimídia ao componente) e particularizar e/ou definir suas características de apresentação. Por exemplo, o autor pode trocar as características originais de apresentação dos objetos multimídia (como o volume sonoro, velocidade de apresentação, etc.), ou ele pode definir novas características, como a posição espacial de apresentação de uma imagem.

A fim de modelar uma apresentação, um modelo multimídia deve permitir a especificação das seguintes informações:

- características temporais de apresentação das informações dinâmicas, como a velocidade, posição de início e de fim de um vídeo e o número de repetições;
- características espaciais de apresentação de informações visuais, como o tamanho, a posição e o estilo de apresentação;
- características das apresentações sonoras, como o volume de apresentação;
- dispositivos de saída, chamados aqui de canais, nos quais as informações serão apresentadas e vistas pelo leitor (por exemplo, uma janela, um canal de áudio);
- apresentações alternativas podem ser definidas a fim de repor uma apresentação principal se ela não puder ser apresentada em um certo sistema (permitindo a criação de documentos adaptáveis aos recursos disponíveis), se existirem problemas de acesso ou restrições temporais não satisfeitas.

Composição espacial

O método mais usual de posicionamento espacial da apresentação dos componentes na tela do computador é a definição da posição espacial absoluta (Figura 6a) ou relativa (Figura 6b) de cada apresentação em um sistema de coordenadas virtuais.

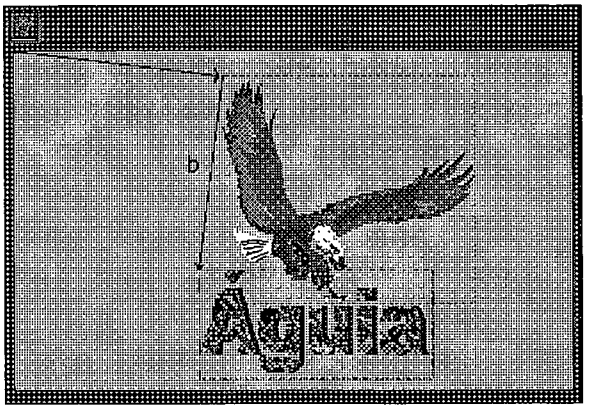


Figura 6. Composição Espacial

2.3.3 Estrutura de Conteúdo

Uma das primeiras etapas da realização de um documento multimídia é a geração ou captura dos materiais (textos, imagens, áudios, vídeos, etc.) que vão ser utilizados para compor o documento. Por exemplo, o autor pode usar uma câmera de vídeo para registrar uma seqüência de vídeo, criar uma informação gráfica a partir de um editor. Neste trabalho, estas informações são chamadas de **dados primitivos**.

A estrutura do conteúdo é responsável pela descrição dos dados primitivos. Por exemplo, ela pode ser constituída por um conjunto de dados primitivos e seus descritores. O par dados primitivos e descritores destes dados é chamado de **objeto multimídia**.

Ao nível da estrutura do conteúdo, um modelo multimídia deve permitir, entre outros, a especificação das informações de acesso e de manipulação dos dados primitivos e os valores originais das características espaciais, sonoras e temporais de apresentação.

2.3.4 Interações

Em aplicações interativas (p.e., multimídia interativa, hipertexto e hipermídia) o usuário deve dispor de um conjunto de mecanismos que permitam o controle da apresentação. Existem basicamente quatro métodos de interação [Hardman, 95]:

- **navegação**: este método permite a especificação de um conjunto de escolhas dado ao usuário a fim de que ele possa selecionar um contexto entre vários. Ela é geralmente definida através da criação de um *link* ou *script*[1] que liga as âncoras-origens às âncoras-destinos;
- **controle da apresentação**: este método de interação é freqüentemente encontrado em documentos multimídia, onde o leitor pode parar, recomeçar, avançar ou retroceder a apresentação do componente multimídia;
- **controle do ambiente**: este método permite a particularização do ambiente de apresentação do documento. Por exemplo, o leitor pode desativar o canal de áudio ou ainda pode alterar o tamanho de uma janela;

[1] Refere-se à execução de um procedimento em linguagem de alto nível.

- **interações da aplicação**: nos métodos anteriores, o autor cria o documento e o leitor apenas interage com ele. Existem aplicações que requerem mecanismos específicos, por exemplo, nas aplicações de tele-ensino, o modelo deve permitir a especificação da noção de acompanhamento dos alunos (por exemplo, histórico das atividades do aluno) e de avaliação (por exemplo, a partir de um campo de entrada de dados). Outro método de interação é a pesquisa por palavras-chave. Este tipo de mecanismo de interação é suportado por ferramentas especializadas.

Um modelo multimídia ideal deveria permitir a especificação de todos estes tipos de interação. Além disso, a fim de simplificar a tarefa de estruturação conceitual de um documento multimídia, um modelo deveria fornecer uma abordagem uniforme de representação dos componentes, das relações lógicas e temporais, e dos mecanismos de interação.

2.4 Abordagens para Autoria de Documentos Multimídia

Existem várias ferramentas de autoria. Uma lista completa com *links* para as diversas ferramentas de autoria comercial e de domínio público pode ser encontrada em [FAQ, 97]. Além destas, existem outras ferramentas que são encontradas na literatura acadêmica que exploram abordagens mais inovadoras. Este trabalho trata destas duas categorias de ferramentas de autoria.

Vários paradigmas básicos de autoria são suportados pelos sistemas de autoria multimídia. Este trabalho apresenta alguns destes paradigmas ou abordagens.

2.4.1 Linguagens Scripting

O paradigma *Scripting*, ou baseada em linguagens, é o método de autoria no estilo da programação tradicional (Figura 7). Neste caso tem-se uma linguagem de programação que especifica elementos multimídia (por nome de arquivo), seqüenciamento, sincronização, etc. Esta forma de concepção de documento é adotada por [Gibbs, 91], [Herman, 94], [Klas, 90] e [Vazirgiannis, 93].

```
set win=main_win
set cursor=wait
clear win
put background "pastel.gif"
put text "heading1.txt" at 10,0
```

```
put picture "gables.pic" at 20,0
put picture "logo.pic" at 40,10
put text "contents.txt" at 20,10
set cursor=active
```

Figura 7. Exemplo de *Script*

Estes modelos têm um poder de expressão muito grande, mas a especificação da composição de um documento multimídia na forma textual é difícil de produzir e modificar ([Ackermann, 94], [Hudson, 93]). Além disso, a composição temporal dos componentes é difícil de identificar.

Os modelos gráficos, vistos mais adiante, têm a vantagem de ilustrar de maneira gráfica a semântica das relações temporais. Este tipo de modelo simplifica a especificação das restrições temporais e reduz o tempo de criação do documento. Como deficiência, modelos gráficos têm um poder de expressão geralmente menor que os modelos orientados à linguagem. Para os sistemas de autoria há, portanto, um dilema acerca de como balancear a facilidade de uso com poder e flexibilidade.

Fazer um software extremamente fácil para aprender e utilizar, risca em restringir a ação de autores experientes ou limitar as possibilidades interativas para o usuário final. Prover grande flexibilidade e poder risca em tornar o software de difícil manipulação. Uma solução tem sido combinar ferramentas de construção simples, similar aos programas de desenho dirigidos a menus com linguagens *scripting*. *Scripting* é a maneira de associar um *script*, um conjunto de comandos escritos numa forma semelhante a programa de computador, com um elemento interativo numa tela, tal como um botão. Alguns exemplos de linguagens de *scripting* são *Apple HyperTalk* para *HyperCard*, *Lingo Macromedia* para *Director* e *Asymetrix OpenScript* para *Toolbook*. Esta solução permite aos autores inexperientes começar a trabalhar rapidamente em uma apresentação e permite aos autores mais avançados criar comportamentos personalizados sofisticados.

2.4.2 Abordagem Baseada em Informação

Nesta abordagem, também chamada de centrada na informação, o conteúdo é obtido de informações existentes, se disponível. Estas informações são então estruturadas e armazenadas em uma base de dados. A estruturação envolve a divisão da

informação em nós e identificadores-chaves e em seguida a ligação destes conceitos [Ginige, 95].

O WWW é um exemplo de um sistema centrado em informação. O autor primeiro cria o texto e outras mídias, então ele estrutura estes usando um editor HTML. Ligações para outros documentos são embutidas no documento durante o processo de *Markup*. Usuários podem ver este documento usando um sistema de apresentação (*Browser*) tal como NCSA *Mosaic*, *Netscape Navigator* ou *Internet Explorer*.

Multimedia Viewer da *Microsoft* é outro exemplo de um sistema combinado de autoria e apresentação baseado na abordagem centrada na informação.

2.4.3 Linha Temporal (Timeline)

A linha temporal permite o alinhamento das apresentações em um eixo temporal. A Figura 8 apresenta um exemplo de linha temporal. Ela indica que um áudio será apresentada de 4 até 16 unidades de tempo, uma imagem será apresentado de 8 até 16 unidades de tempo, e, finalmente, um vídeo será apresentado de 16 até 28 unidades de tempo.

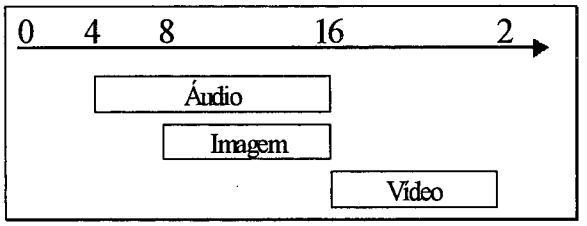


Figura 8. Exemplo de Linha Temporal

Esta técnica oferece como principal vantagem uma grande simplicidade de expressão dos esquemas de sincronização. Além disso, o autor tem uma visão muito clara das informações que serão apresentadas e em que momento. Infelizmente ela apresenta várias limitações:

- ela permite somente a especificação do alinhamento temporal ideal das apresentações, na medida em que ela define pontos de partida e fim ideais das apresentações dos componentes do documento. Não é possível especificar métodos de tolerância da sincronização ou ações a perdas de sincronismo;
- ela requer o conhecimento exato da duração das apresentações. Geralmente o autor deve manipular os segmentos de informação de maneira manual, a fim de obter o

comportamento temporal desejado. Isto através da edição dos dados ou pela mudança da velocidade de apresentação;

- como esta abordagem não fornece mecanismos de estruturação, nem a representação das relações lógicas, ela não permite a definição da estrutura conceitual completa de documentos hipermídia e multimídia interativos.

Alguns modelos multimídia e hipermídia utilizam versões estendidas da abordagem linha temporal, permitindo a definição de relações lógicas e temporais entre apresentações (por exemplo, a abordagem proposta por [Hirzalla, 95]). Em todo caso, os modelos resultantes destas extensões tornam-se geralmente complexos e perdem a vantagem principal do modelo linha temporal de base que é a simplicidade de compreensão do comportamento temporal do documento.

O sistema *MAEstro* [Drapeau, 91], *QuickTime* [Apple, 91], *Macromedia Director* (*Macintosh* e *Windows*), *Animation Works Interactive* (*Windows*), *MediaBlitz!* (*Windows*), *Producer* (*Macintosh* e *Windows*), e a norma *HyTime* [Newcomb, 91] adotam este paradigma.

2.4.4 Modelo de Composição Via Pontos de Referência

Na abordagem de composição via pontos de referência, as apresentações são vistas como seqüências de subunidades discretas [Blakowski, 92]. A posição de uma subunidade (p.e. um quadro de um vídeo ou uma amostragem de áudio) em um objeto é chamada de **ponto de referência**. As mídias discretas (p.e. texto) apresentam somente dois pontos de referência: início e fim da apresentação. A sincronização é definida por conexões entre pontos de referência. A Figura 9 ilustra a composição via pontos de referência.

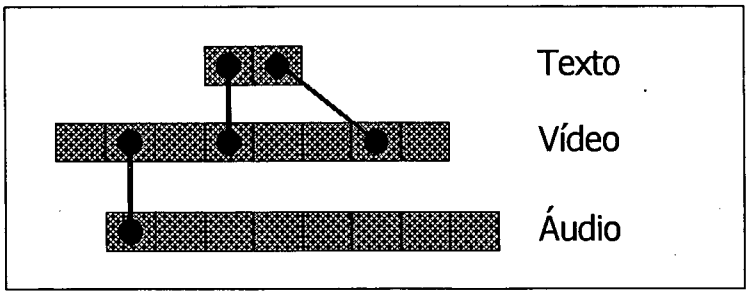


Figura 9. Sincronização Via Pontos de Referência

Uma vantagem do conceito de ponto de referência é a facilidade da mudança da velocidade de apresentação [Haindl, 96], porque não existe referência explícita ao tempo. As limitações desta abordagem são as seguintes:

- as sincronizações são definidas por um conjunto de conexões entre pontos de referência. Quando da apresentação de um documento, as irregularidades de apresentação são ignoradas. Não existe a possibilidade de tolerância a este nível;
- a utilização única de uma abordagem de composição via pontos de referência não permite a especificação de retardos e nem a definição de sincronizações condicionais;
- esta abordagem não é apropriada para a descrição da estrutura conceitual dos documentos multimídia. Isto, pois ela não propõe nenhum mecanismo de estruturação e não há a distinção entre o componente e seu conteúdo.

2.4.5 Composição Hierárquica

Na composição hierárquica de documentos multimídia, os componentes de um documento são organizados na forma de uma árvore, como ilustrado na Figura 10. Nesta árvore os nós internos representam relações temporais entre as subárvores de saída. Estas relações temporais são definidas por operadores de sincronização. Os principais operadores de sincronização são os operadores série e paralelo. Estes operadores definem que um conjunto de ações serão executados em série ou paralelo. Estas ações podem ser atômicas ou compostas [Blakowski, 92]: uma ação atômica manipula a apresentação de uma informação, a interação com o utilizador ou um atraso; as ações compostas são combinações de operadores de sincronização e ações atômicas.

Uma das vantagens desta abordagem é a possibilidade de agrupar itens em "mini-apresentações" que podem ser manipuladas como um todo [Hardman, 95]. Esta abordagem apresenta também algumas limitações:

- ela não permite a representação de relações temporais fora das fronteiras hierárquicas. Assim, os *links* hiperestruturais não podem ser especificados;
- as relações lógicas e temporais não são representadas de uma maneira natural. Ao contrário da abordagem linha temporal, o autor não tem a visão do conjunto das informações apresentadas e suas datas de apresentação;

- um conjunto de ações pode ser sincronizado apenas com relação ao início ou fim de um conjunto de ações. Isto significa que, por exemplo, a apresentação de legendas em uma certa parte de um vídeo requer o corte da seqüência de imagens em vários componentes consecutivos.

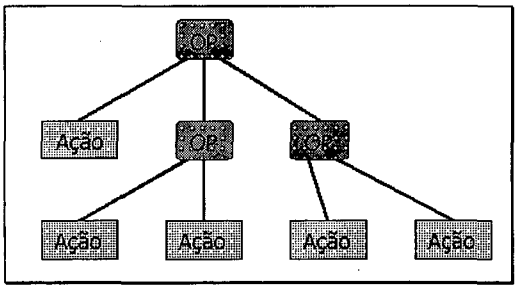


Figura 10. Composição Hierárquica

2.4.6 Modelos baseados em ícones

A criação de um documento multimídia pela utilização de uma abordagem baseada em ícones é similar à programação, por exemplo, utilizando a linguagem C, mas com a ajuda de uma interface gráfica. Esta interface fornece ícones de alto nível, a fim de visualizar e manipular a estrutura do documento.

Um conjunto de ícones é arranjado em um grafo que especifica interações e caminhos de controle de apresentação de um documento multimídia. Em geral, as funcionalidades associadas a cada ícone podem ser modificadas utilizando menus e editores associados.

A Figura 11 apresenta um exemplo de utilização desta abordagem. Nesta aplicação, o leitor pode escolher, a partir de um menu, entre a apresentação de um áudio, a apresentação de uma imagem e o término da aplicação. Ao fim da apresentação do áudio ou da imagem, o menu é apresentado novamente.

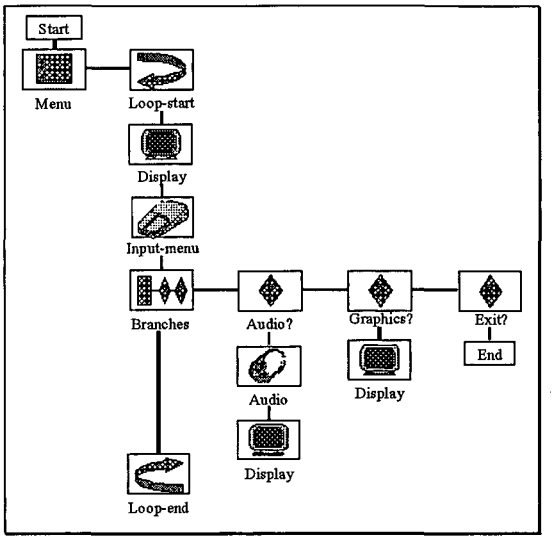


Figura 11. Exemplo do uso de ícones para a autoria de documentos multimídia

Esta abordagem é de utilização simples para pequenas aplicações, mas para aplicações complexas, a compreensão e a manipulação são dificultadas.

A abordagem de composição de documentos multimídia baseada em ícones é adotada, por exemplo, pelos softwares *AimTech IconAuthor*, *Eyes* M/M [Eyes, 92], *Authorware Professional* [Authorware, 89], *mTropolis* e HSC *Interactive*.

2.4.7 Modelos Baseados em Cartões ou Páginas

Neste paradigma os elementos são organizados em páginas de um livro ou uma pilha de cartões. Ferramentas de autoria baseadas neste paradigma permitem que o autor ligue as páginas ou cartões formando uma estrutura de páginas ou cartões.

Sistemas de autoria baseados em cartões ou páginas fornecem um paradigma simples para organizar elementos multimídia. Estes tipos de ferramentas de autoria permitem que o autor organize os elementos em seqüências ou agrupamentos lógicos tal como capítulos e páginas de um livro ou cartões em um catálogo de cartões.

Sistemas de autoria baseados em páginas são orientados a objetos [Apple, 94]: objetos são botões, campos de texto, objetos gráficos, fundos, páginas e cartões, e mesmo o projeto em si. Cada objeto pode conter um *script*, ativado quando ocorre um evento (tal como um clique no mouse) relacionado ao objeto.

Exemplos de ferramentas de autoria adotando este paradigma incluem: *HyperCard* (*Macintosh*), *SuperCard* (*Macintosh*), *ToolBook* (*Windows*) e *VisualBasic* (*Windows*).

2.4.8 Redes de Petri

Rede de Petri é uma técnica de descrição formal muito utilizada na engenharia de protocolos, automação industrial e muitas outras áreas. Ela permite a realização de uma especificação formal de um sistema e permite também a aplicação de técnicas de análise sobre o sistema a fim de verificar certas propriedades [Murata, 89].

Redes de Petri também são utilizadas para a modelização de documentos multimídia, isto devido a sua representação gráfica, à facilidade de modelização dos esquemas de sincronização, e à possibilidade de analisar propriedades importantes do comportamento lógico e temporal do documento. Existem vários modelos multimídia baseados em Redes de Petri. Uma lista não exaustiva inclui: OCPN [Little, 90], TSPN [Diaz, 93], RTSM [Yang, 96], Trellis [Stotts, 90], HTSPN [Sénac, 95], MORENA [Botafogo, 95], PHPN [Wang, 95]. Nestes modelos, geralmente um lugar da Rede de Petri representa uma apresentação, as transições e arcos definem relações lógicas e temporais.

A Figura 12 ilustra a utilização de Redes de Petri para especificar um cenário multimídia. Esta especificação indica que inicialmente AP1 será apresentada; terminada esta apresentação, AP2 e AP3 serão apresentadas em paralelo; tão logo estas duas apresentações terminem, AP4 será apresentada.

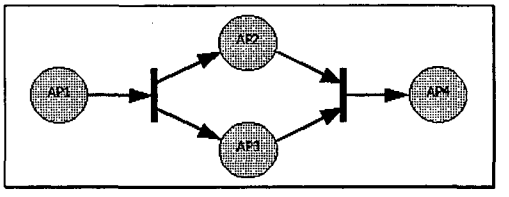


Figura 12. Composição usando Redes de Petri

Esta abordagem apresenta algumas desvantagens:

- de maneira similar à composição hierárquica, na maior parte dos modelos baseados em Redes de Petri, um conjunto de ações pode ser sincronizado com relação ao início ou fim de um conjunto de ações;

- esta abordagem não contém os conceitos apropriados para a representação das sincronizações condicionais mais complexas nem para os métodos de interação de aplicação (seção 2.3.4);
- em [Hudson, 93], os autores afirmam que uma outra limitação desta abordagem é sua complexidade: este formalismo tende a ser muito potente e de muito baixo nível para sua utilização direta pelo autor do documento.

A grande vantagem do uso de um modelo de autoria baseado em Redes de Petri é o caráter formal desta técnica de descrição. A especificação de documentos multimídia utilizando modelos informais ou semiformais pode ser fonte de ambigüidades e geralmente não permite a utilização de ferramentas de análises avançadas. Ao contrário do anterior, modelos multimídia baseados em métodos formais, como aqueles baseados em Redes de Petri, permite a construção de uma semântica precisa e não ambígua de um documento e eles permitem a utilização de técnicas de análises sofisticadas, como a simulação, validação, verificação do comportamento do documento.

Nem todos os modelos ditos baseados em Redes de Petri apresentados na literatura podem ser considerados modelos formais. Assim, alguns destes modelos não permitem a aplicação de técnicas de análise para validar o comportamento do documento multimídia.

2.5 Conclusão

Este capítulo mostrou a problemática de criação de documentos multimídia. Foi apresentado que, para criar ou realizar a autoria de um documento multimídia, é necessário uma ferramenta de autoria que facilita e automatiza a criação. Além disso, é importante respeitar as três estruturas básicas de criação de um documento multimídia interativo [Willrich, 96a]: estrutura conceitual, estrutura de apresentação e estrutura de conteúdo.

Neste capítulo foram apresentados os seguintes paradigmas de autoria: linguagens *scripting*, abordagem baseada em informação, linha temporal (*Timeline*), modelo de composição via pontos de referência, composição hierárquica, modelos

baseados em ícones, modelos baseados em cartões ou páginas e Redes de *Petri*. Cada um desses paradigmas possui as suas vantagens e desvantagens.

O próximo capítulo apresentará a linguagem SMIL 1.0 utilizada para a autoria de documentos multimídia para a Web. Ela baseia-se no paradigma de autoria de linguagens *scripting*.

Capítulo 3. SMIL

Este capítulo apresenta a linguagem SMIL (*Synchronized Multimedia Integration Language)*, a linguagem definida pelo Consórcio WWW (W3C) para descrição de cenários multimídia na Web [W3C 00].

3.1 Origem e Necessidade

Após um *workshop* W3C sobre "Multimídia Tempo-Real e a Web" em outubro de 1996, o W3C estabeleceu um grupo de trabalho em multimídia sincronizada em março de 1997. Este grupo trabalha no projeto de uma linguagem declarativa para descrever apresentações multimídia na Web usando XML (*Extensible Markup Language*). Esta linguagem foi chamada de SMIL (pronunciada *smile*) [W3C 98].

Uma apresentação SMIL tem as seguintes características:

- é uma linguagem declarativa, com características de interoperabilidade e perenidade;
- possibilidade de ser editada manualmente;
- a apresentação é composta de vários componentes que são acessíveis via URIs, por exemplo, arquivos armazenados em um servidor Web (os elementos são referenciados e não incluídos);
- os componentes têm diferentes tipos de mídia de apresentação, tal como áudio, vídeo, imagem, texto;
- os instantes de início e fim dos diferentes componentes são especificados relativo a eventos em outros componentes. Por exemplo, na apresentação de slides, um slide particular é apresentado quando um narrador inicia a fala acerca do slide;
- controles familiares via botões tal como parar, avanço e retrocesso rápidos e "rebobinar" permitem ao usuário controlar a apresentação;

- funções adicionais para acesso randômico (a apresentação pode ser partida de qualquer lugar) e apresentação "câmera lenta" (apresentação em velocidade reduzida);
- o usuário pode seguir hiperlinks embutidas na apresentação.

A chave do sucesso do HTML foi que documentos hipertextos poderiam ser criados sem a necessidade de ferramentas de autorias sofisticadas. SMIL permite o mesmo para hipermídia sincronizada.

O atual protocolo de transporte usado pela Web, o HTTP, é utilizável para carga completa de arquivos. Além da versão atual do HTTP, multimídia sincronizada no W3C é baseado nos protocolos RTP (*Real Time Transport Protocol*) e RTSP para transferência e controle de mídias contínuas.

3.2 SGML, XML e SMIL

O padrão internacional SGML (*Standard Generalized Markup Language*) [ISO 86] foi proposto para permitir que documentos eletrônicos pudessem ser definidos conforme seu conteúdo e estrutura, independentemente de sua forma de apresentação. Para um livro, por exemplo, são descritos seus capítulos e seções, sem especificar o formato das páginas, o número de colunas, ou o tipo de fonte utilizada. Mais exatamente, SGML é uma metalinguagem, ou seja, uma maneira de descrever formalmente uma linguagem, neste caso uma linguagem para marcação de textos eletrônicos. Uma descrição mais detalhada do SGML é disponível em [Brown, 89].

Uma propriedade importante de SGML é sua capacidade de descrever a estrutura lógica de um documento através de marcas padronizadas (*markup*). O termo *markup* refere-se à seqüência de caracteres ou de outros símbolos que são colocados em certos locais em um texto para indicar como ele deve ser apresentado ou como ele está estruturado. Os indicadores de *markup* são geralmente intitulados *tags*.

Os *tags* SGML dividem o documento em componentes, chamados **elementos SGML**, os quais podem conter uma combinação de texto e outros elementos. Cada elemento pode ter **atributos SGML**, que são variáveis e que descrevem um elemento e

seu conteúdo (por exemplo: um parágrafo que pode conter textos, uma figura que pode ser de um determinado formato, etc.).

SGML não é um conjunto predeterminado de marcas e sim uma linguagem para se definir quaisquer conjuntos de marcas, uma linguagem autodescritiva. Cada documento SGML carrega consigo sua própria especificação formal, o DTD (*Data Type Document*).

O DTD é uma espécie de gramática formal criada a partir da notação EBNF (*Extended Backus – Naur Form*) que define como as marcas devem ser interpretadas, quais as regras que restringem o uso de cada marca nos diferentes contextos do documento, e até mesmo, quando relevante for, a ordem em que as marcas devem aparecer no documento. Resumindo, SGML é uma linguagem para definir outras linguagens, ou ainda uma linguagem para conceber DTDs, tipos de documento.

HTML é um exemplo de linguagem originada de SGML. Ou seja, a definição formal (ou especificação, ou ainda o DTD) de HTML é construída em SGML.

XML é um subconjunto simplificado de SGML, otimizado para a utilização na Internet, que resultou do trabalho de um grupo de especialistas estabelecido em 1996 pelo W3C (*World-Wide Consortium*). Como o próprio nome sugere, XML é uma linguagem extensível (metalinguagem), a partir da qual os projetistas podem criar seus próprios elementos de acordo com a aplicação que está sendo modelada, adicionando significado semântico ao conteúdo e à estrutura da informação, sem se preocupar com a forma de apresentação. Através desta característica, a flexibilidade da linguagem é garantida, uma vez que o projetista pode criar uma nova linguagem a partir da metalinguagem XML [XML 98].

SMIL é uma aplicação de XML que descreve o *layout* e o padrão temporal de apresentações multimídia, e a maneira como os objetos de mídia usam *links* de hipertexto [Stanek, 99]. Os primeiros trabalhos com a SMIL 1.0 começaram em dezembro de 1995. A especificação passou por muitas revisões antes de se tornar uma recomendação oficial do World Wide Web Consortium, em 15 de junho de 1998 (www.w3.org/TR/rec-smil).

Devido às limitações do SMIL 1.0 e ao interesse pela integração dos conceitos de SMIL com os de HTML e de outras linguagens XML, começou em 20 de agosto de

1999 os primeiros trabalhos com o SMIL Boston (www.w3.org/TR/smil-boston). A especificação já passou por três revisões: uma em 15 de novembro de 1999, outra em 25 de fevereiro de 2000 e outra em 22 de junho de 2000. Antes de se tornar uma recomendação oficial do W3C, essa linguagem passará por outras revisões.

Por simplificação nesse capítulo, a linguagem SMIL Boston, às vezes, será referenciada simplesmente por SMIL.

3.3 SMIL Boston

As funcionalidades da linguagem SMIL Boston são divididas em nove áreas. Cada área é dividida em módulos que são complementares, ou seja, adiciona novos recursos ao módulo inferior. Por exemplo, o módulo de tempo nível 2 adiciona o atributo *syncBehavior* complementando os módulos de tempo níveis 0 e 1.

As áreas em que a linguagem SMIL Boston está dividida estão listadas abaixo. Está informado, nos parênteses, quantos módulos constituem cada uma dessas áreas:

- Funcionalidade de tempo (3 módulos);
- Funcionalidade de animação (2 módulos);
- Funcionalidade de efeitos de transição (2 módulos);
- Funcionalidade de mídia (2 módulos);
- Funcionalidade de controle de conteúdo (2 módulos);
- Funcionalidade de metainformação (1 módulo);
- Funcionalidade de estrutura (1 módulo);
- Funcionalidade de *layout* (3 módulos);
- Funcionalidade de ligação (2 módulos);

Os módulos de estrutura, de tempo e de objetos de mídia, nível 0, são módulos obrigatórios em qualquer perfil da linguagem SMIL Boston.

Cada módulo define seus elementos e atributos. Por exemplo, a funcionalidade de tempo inclui os módulos de tempo 0, 1 e 2. O módulo de tempo nível 0 define os elementos *par* e *seq*, e os atributos *begin* (condição simples), *end* (condição simples), *endsync*, *dur*, *repeat*, *repeatCount*, *repeatDur*, *timeAction* e *timeContainer*. Já o módulo de tempo nível 1 define os elementos *excl* e *priorityClass*, além disso, define os atributos *begin* (condições múltiplas), *end* (condições múltiplas), *restart*, *restartDefault* e *fill*. O módulo de tempo nível 2 define os atributos *begin* (com mídias marcadas), *end*

(com mídias marcadas), *syncMaster*, *syncTolerance*, *syncToleranceDefault*, *syncBehavior* e *syncBehaviorDefault*.

Quando uma linguagem incluir um módulo de mais alto nível, os módulos de mais baixo nível devem ser incluídos.

Perfis são construídos utilizando os módulos SMIL. Cinco perfis são definidos pela própria linguagem para informar aos usuários como os perfis podem ser construídos para solucionar problemas:

- Perfil SMIL Boston: inclui todas as funcionalidades definidas nessa linguagem.
- Perfil HTML + SMIL: integra a noção de tempo da linguagem SMIL com a linguagem HTML, incluindo as funcionalidades de animação, de controle de conteúdo, de ligações, de objetos de mídia, de efeitos de transição e a funcionalidade de tempo.
- Perfil SMIL Boston básico: inclui as funcionalidades de *layout*, de ligações, de objetos de mídia, de estrutura, de controle de conteúdo e a funcionalidade de tempo.
- Perfil de apresentações leves: utilizado para apresentações simples, suportando apenas texto como conteúdo. Este perfil pode ser utilizado para realizar uma seqüência de citações para ser utilizada em dispositivos como *palmtop*. Ele inclui as seguintes funcionalidades: de animação, de efeitos de transição e de tempo.
- Perfil intensificador de mídias na Web: suporta a integração de apresentações multimídia com *broadcast* ou mídias sob demanda. Inclui as funcionalidades de ligações, de efeitos de transição, de estrutura, de animação, de controle de conteúdo, de objetos de mídia, de *layout* e a funcionalidade de tempo.

Em vez de apresentar os elementos e atributos constituintes de cada módulo, optou-se em apresentar a linguagem SMIL Boston partindo-se de um arquivo fonte. Os elementos e atributos da linguagem serão apresentados conforme a sua posição em um arquivo SMIL: no cabeçalho ou no corpo do documento.

3.3.1 Estrutura Geral

Um documento SMIL Boston é dividido em duas seções: o cabeçalho (*head*) e o corpo (*body*). Ambas precisam estar incluídas dentro do *tag* <smil>. A Figura 13 apresenta um exemplo da disposição desses elementos em um arquivo SMIL Boston.

O corpo de um documento SMIL é uma composição seqüencial, ou seja, os componentes são apresentados em série.

```
<smil>
   <head>
      <meta name="titulo" content=" Exemplo de um código SMIL " />
      <meta name="autor" content="adriana@inf.ufsc.br" />
      <meta name="copyright" content="livre" />
      <layout >
         <root-layout width="400" height="200" background-color="black" />
         <region id="esquerda" title="região localizada a esquerda da janela principal"
                  left="0" top="0" height="144" width="176" background-color="#000000" fit="hidden"/>
         <region id="direita" title=" região localizada a direita da janela principal "
                  left="180" top="0" height="144" width="220" background-color="#000000" fit="hidden"/>
         <region id="inferior" title=" região localizada na parte inferior da janela principal "
                  left="100" top="150" height="40" width="300" background-color="#000000" fit="fill"/>
      </layout>
   </head>
   <body>
      <seq>
         <par title="Composicao 1 do seq">
            <video src="video.rm" id="video" region="esquerda"  title="video" alt="Amo Você" dur="5s"/>
            <audio src="audio1.rm" title="Parabéns a Você" dur="5s"/>
         </par>
         <seq title="Composicao 2 do seq">
            <img src=" http://www.inf.ufsc.br/~adriana/figura1.jpg" id="graficos0" region="esquerda"
                  title="imagem 1" alt="Feliz" dur="5s"/>
            <img src=" http://www.inf.ufsc.br/~adriana/figura2.jpg" id="graficos3" region="direita"
                  title="imagem 2" alt="Aniversário" dur="5s"/>
            <img src=" http://www.inf.ufsc.br/~adriana/figura3.jpg" id="graficos4" region="inferior"
                  title="imagem 3" alt="Mamãe" dur="5s"/>
         </seq>
         <par title="Composicao 3 do seq">
            <animation src="http://www.inf.ufsc.br/~adriana/animacao.gif" id="graficos9" region="direita"
                  title="Animacao" dur="20s"/>
            <audio src=" http://www.inf.ufsc.br/~adriana/audio2.rm" begin="0.4s"  title="Seu Aniversário"
                  dur="20s"/>
            <a show="replace" href="http://www.inf.ufsc.br/~adriana/figura4.gif">
                  <img src="http://www.inf.ufsc.br/~adriana/interacao.gif" id="graficos10" region="inferior"
                     title="imagem 4" dur="20s"/>
            </a>
         </par>
      </seq>
   </body>
</smil>
```

Figura 13. Exemplo de um arquivo SMIL

No exemplo da Figura 13 ocorre a apresentação de um áudio e um vídeo em paralelo (*audio1* e *video*) que é seguida por uma seqüência de slides (*figura1*, *figura2* e *figura3*) e uma animação (*animacao*). Essa animação é parcialmente comentada por um áudio (*audio2*) e, quando iniciada a sua execução, é exibida ao usuário uma solicitação de interação (*interacao*), se o usuário fizer a seleção, uma imagem final (*figura4*) é apresentada. A Figura 14 apresenta o comportamento temporal, através de uma linha temporal, do documento SMIL apresentado na Figura 13.

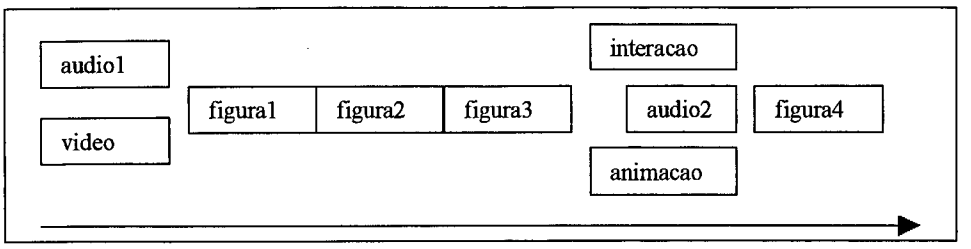


Figura 14. Especificação informal do código fonte da Figura 13

3.3.2 Cabeçalho

O cabeçalho contém a definição do *layout* espacial da apresentação e informações gerais relativas ao documento. Os elementos que podem ser encontrados no cabeçalho de um arquivo SMIL são os elementos ***layout*** e o ***meta***.

O elemento ***layout*** contém a definição da janela principal do documento (dimensão, cor de fundo) e das regiões onde os objetos (imagens, vídeo) serão apresentados. Podem ser encontrados, dentro desse *tag*, elementos como o ***root-layout,*** o ***viewport*** e o ***region.***

O elemento ***root-layout*** define uma zona retangular e representa a janela principal onde o documento será exibido, sendo referente a ela a definição de todas as demais regiões. O código seguinte criará uma janela com as dimensões de 400x200 *pixeis*, com a cor de fundo preta.

```
<layout>
    <root-layout width="400" height="200" background-color="black" />
    ...
</layout>
```

Figura 15. Fragmento do código da Figura 13, exemplificando a utilização do elemento *root-layout*

Se nenhuma definição de página ***root-layout*** for especificada, o tamanho da janela será calculado pelo software de visualização, de modo que ela seja tão ampla quanto o objeto multimídia mais amplo.

O elemento ***viewport*** permite a definição de múltiplas janelas, onde o documento será exibido, sendo referente a cada uma das janelas a definição de todas as demais regiões. Cada região pode pertencer a, no máximo, uma janela. Essas janelas funcionam como recipientes, elas não possuem significado temporal, ou seja, não é definida uma linha temporal individual para cada janela. Há uma linha temporal mestra para a apresentação do arquivo SMIL, não importando quantas janelas tenham sido criadas. A

vantagem dessa abordagem é a possibilidade de sincronizar mídias exibidas em múltiplas janelas. O código seguinte criará duas janelas com as dimensões de 320x240 e 320x60, contento respectivamente as regiões "figuras" e "legendas".

```
<layout>
    <viewport id="JanelaV" title="Video" width="320" height="240"/>
        <region id="figuras" title=" figuras " height="100%" fit="meet"/>
    </ viewport >

    < viewport id="JanelaC" title="Títulos" width="320" height="60">
        <region id="legendas" top="JanelaC" title=" legendas" top="90%" fit="meet"/>
    </ viewport >
</layout>
```

Figura 16. Exemplo de utilização do *tag top-layout*

O elemento ***region*** controla a posição, tamanho e escala dos objetos multimídia. Por exemplo, o código da Figura 17 criará três regiões na janela (definida na Figura 15): esquerda, direita e inferior, conforme ilustra a Figura 18. Os atributos utilizados no elemento *region* no código da Figura 17 foram:

- *id*, que permite a identificação de um elemento dentro de um documento. Um *id* para cada região é obrigatório;
- *title*, que permite a inclusão de uma descrição do significado do elemento que foi definido;
- *left*, que define a distância em porcentagem ou pixels do elemento definido ao lado esquerdo da janela principal.
- *top*, que define a distância em porcentagem ou pixels do elemento definido à parte superior da janela principal;
- *height*, que define a altura em porcentagem ou pixels do elemento definido;
- *width*, que define a largura em porcentagem ou pixels do elemento definido;
- *background-color*, que define a cor de fundo da região. Se este atributo estiver ausente, o fundo será transparente;
- *fit*, que especifica o comportamento no caso da altura ou largura intrínseca de um objeto multimídia ser diferente dos valores especificados pelos atributos de altura e largura do elemento *region*. O valor *hidden* para o atributo *fit* define que se a altura ou largura intrínseca do elemento for inferior à altura ou largura definida no elemento *region*, o objeto é reproduzido a partir da extremidade superior esquerda e

a altura restante é preenchida com a cor de fundo, por outro lado, se a altura ou largura intrínseca do elemento for superior à altura ou largura definida no elemento *region*, o objeto é reproduzido a partir da extremidade superior esquerda até se alcançar a altura ou largura definida no elemento *region*, sendo omitidas as partes do objeto que ficaram abaixo ou à direita. O valor *fill* dimensiona a altura e a largura do objeto de forma que toque todas as extremidades da caixa.

```
...
    <layout >
        <region id="esquerda" title="região localizada a esquerda da janela principal"
            left="0" top="0" height="144" width="176" background-color="#000000" fit="hidden"/>
        <region id="direita" title=" região localizada a direita da janela principal "
            left="180" top="0" height="144" width="220" background-color="#000000" fit="hidden"/>
        <region id="inferior" title=" região localizada na parte inferior da janela principal "
            left="100" top="150" height="40" width="300" background-color="#000000" fit="fill"/>

    </layout>
...
```

Figura 17. Fragmento do código da Figura 13, exemplificando a utilização do elemento *region*

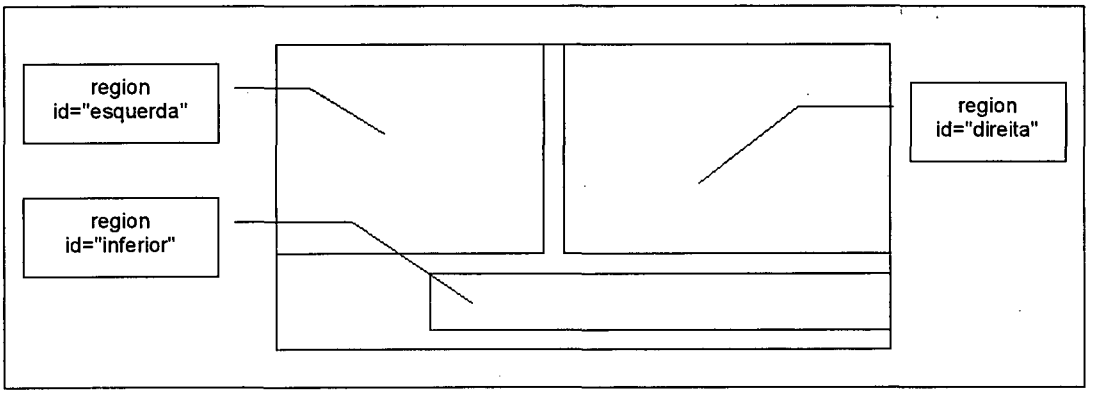


Figura 18. Figura mostrando as regiões

A linguagem SMIL permite criar um *layout* hierárquico através da definição de regiões no interior de uma outra região. O código da Figura 19 cria:

- uma janela com as dimensões de 640x480 pixeis;
- a região "left", com as dimensões 320x480 pixeis, posicionada na parte superior da janela principal e à esquerda;
- a região "right", com as dimensões 320x480 pixeis, posicionada na parte superior e a uma distância de 320 pixels do lado esquerdo da janela principal;

- a região "inset", com as dimensões 160x200 pixeis, posicionada a uma distância de 140 pixels da parte superior da região "right" e a uma distância de 80 pixels do lado esquerdo da região "right".

```
<layout>
    <top-layout width="640px" height="480px" />
    <region id="left" top="0px" left="0px" width="320px" height="480px" />
    <region id="right" top="0px" left="320px" width="320px" height="480px">
        <region id="inset" top="140px" left="80" width="160px" height="200px" />
    </region>
</layout>
```

Figura 19. Exemplo de código que cria um *layout* hierárquico

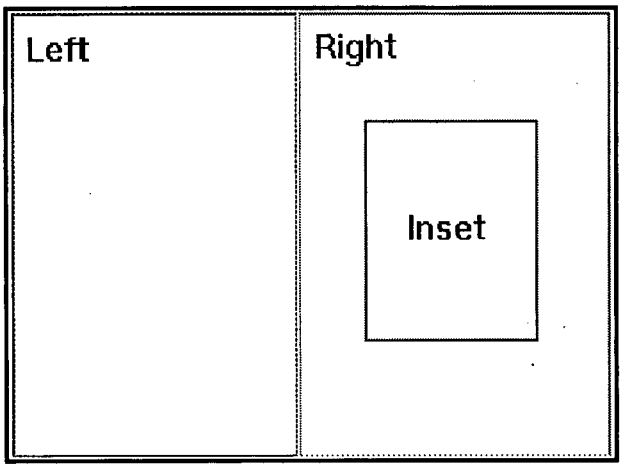


Figura 20. Ilustração do código da Figura 19

O elemento *meta* fornece informações adicionais sobre a apresentação corrente como o autor e a data. No exemplo abaixo são dadas as seguintes informações: o título da apresentação SMIL (exemplo de um código SMIL), o endereço eletrônico da autora da apresentação (adriana@inf.ufsc.br) e informações a respeito dos direitos autorais (livre).

```
...
    <head>
        <meta name="titulo" content=" Exemplo de um código SMIL " />
        <meta name="autor" content="adriana@inf.ufsc.br" />
        <meta name="copyright" content="livre" />
        ...
    </head>
...
```

Figura 21.* Fragmento do código da Figura 13, exemplificando a utilização do elemento *meta

3.3.3 Corpo

O *corpo* contém a definição dos objetos e seus relacionamentos temporais, além da especificação de elementos de hiperligação.

Definição dos Objetos

A definição dos objetos é realizada através dos seguintes elementos: *animation*, *audio*, *img*, *video*, *text*, *textstream* e *ref*. Eles respectivamente definem a inclusão de: uma animação, um áudio, um vídeo, um texto, um texto em movimento. O elemento *ref* é utilizado quando o autor tiver dúvida em relação ao nome de um objeto multimídia.

A Figura 22 apresenta a definição de um objeto em um documento SMIL: *tag* <img> indica o tipo de objeto (imagem); *region* indica a região em que a figura ficará posicionada - "esquerda"; *src* é um URI que localiza o objeto; *alt* é utilizado para especificar um texto alternativo para ser utilizado se não for possível apresentar a imagem (por exemplo, se essa imagem apresentar algum problema de acesso).

```
...
    <img src=" http://www.inf.ufsc.br/~adriana/figura1.jpg" id="graficos0" region="esquerda"
                    title="imagem 1" alt="Feliz" dur="5s"/>
...
```

Figura 22. Fragmento do código da Figura 13, exemplificando a definição de um objeto

Os objetos de mídia são incluídos por referência, através de uma URI, portanto um documento SMIL não possui o conteúdo dos dados associados aos objetos de mídia, mas os faz referência.

Os objetos de mídia em uma apresentação SMIL podem ter uma duração intrínseca ou explícita.

- A duração intrínseca é deduzida do conteúdo, por exemplo, um áudio possui uma duração de 5 segundos, que corresponde a sua duração intrínseca. A duração intrínseca de objetos discretos, como texto ou uma imagem, é zero.
- A duração explícita é a duração definida pelo autor, utilizando-se o atributo denominado duração (dur). No código da Figura 22 foi adicionado no tag <img> esse atributo opcional, ajustado para "5s", o qual resultou na exibição da figura1 por 5 segundos.

Se a duração intrínseca de um vídeo, por exemplo, é de 4 segundos, e se atribuímos uma duração explícita de 6 segundos, o vídeo pára decorridos 4 segundos e congela a última imagem por 2 segundos.

Para se definir o tempo de fim de um elemento multimídia, é necessário identificar em qual das particularidades abaixo ele se encaixa e aplicar a fórmula correspondente. Um elemento multimídia que possui:

- uma duração implícita ou explícita e um tempo de início, tem um **tempo de fim = tempo de início + duração** (Figura 23);
- um tempo de início explícito e um tempo de fim explícito, tem uma **duração = tempo de fim – tempo de início**;
- um tempo de início explícito, uma duração e um tempo de fim explícito, tem um tempo de fim igual ao mínimo entre o **tempo de fim explícito** e a **soma do tempo de início + duração**.

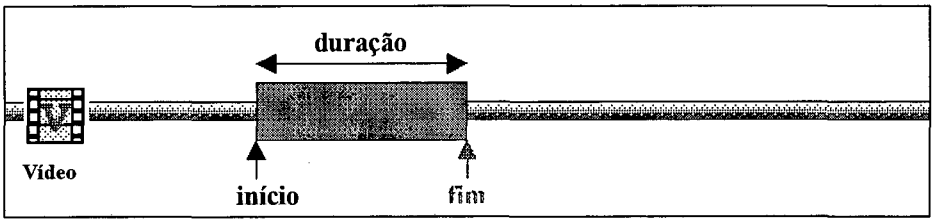


Figura 23. Inclusão de um vídeo com uma duração implícita e um tempo de início

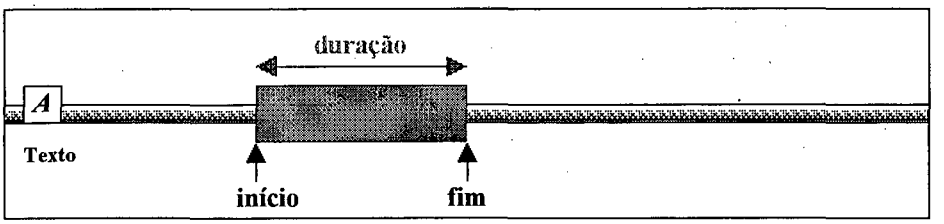


Figura 24. Inclusão de um texto que tem uma duração = tempo de fim – tempo de início

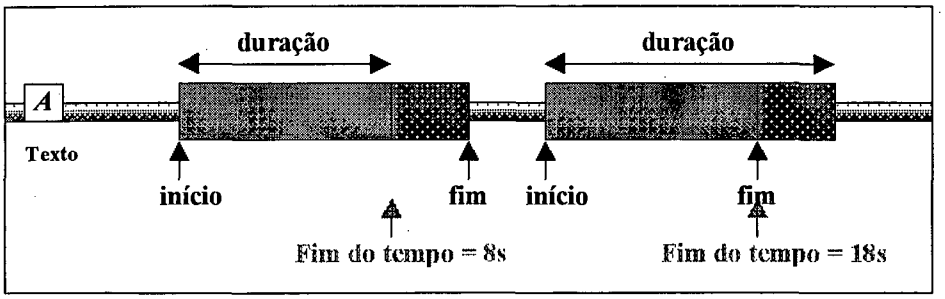


Figura 25. Inclusão de um texto que tem um tempo de fim igual ao mínimo entre o tempo de fim explícito e a soma do tempo de início + duração

No primeiro, o tempo de fim é dado somando-se o tempo de início mais a duração; no segundo, é dado através do tempo de fim explícito.

A duração de um objeto pode ser aumentada pela repetição do conteúdo. O número de exibições é determinado pelo atributo repetições (*repeat, repeatCount e repeatDur*), cujo valor pode ser um número inteiro ou indefinido. Neste último caso, o elemento será exibido repetidamente até o término da composição que o contém.

Ao ser executado o código da Figura 26, o áudio será executado 2 vezes, levando 5 segundos para o término da sua execução (dur * repeatCount).

```
<audio src="background.au" dur="2.5s" repeatCount="2" />
```

Figura 26. **Exemplo de utilização do atributo *repeatCount***

O código da Figura 27 define que a execução do áudio será repetida até se completar 7 segundos.

```
<audio src="musica.mp3" dur="2.5s" repeatDur="7s" />
```

Figura 27. **Exemplo de utilização do atributo *repeatDur***

Elementos podem ser especificados para iniciarem ou finalizarem a sua execução em resposta a um evento disparado pelo usuário.

Ao ser executado o código da Figura 28, o usuário terá a opção de clicar no texto, fazendo com que uma imagem seja apresentada por 3,5 segundos, em resposta ao evento.

```
<text id="mensagem" ... />
<img begin="mensagem.click" dur="3.5s" ... />
```

Figura 28. **Exemplo de uso do atributo *begin* com o valor *mensagem.click***

No exemplo da Figura 30, quando o usuário clicar na imagem, esta será finalizada.

```
<image src="imagem.jpg" end="click" />
```

Figura 29. **Exemplo de utilização do atributo *end* com o valor *click***

No código a seguir serão apresentados 5 vídeos, que terão uma duração total de apresentação de 25 segundos, caso não haja a interação do usuário. Para cada vídeo que é apresentado, o usuário poderá optar em finalizá-lo ou passar para o próximo vídeo, diminuindo assim o tempo total de apresentação do *slideshow*.

```
<seq>
    <video dur="5s" repeatCount="3" end="click; next.click" .../>
    <video dur="5s" repeatCount="3" end="click; next.click" .../>
    <video dur="5s" repeatCount="3" end="click; next.click" .../>
    <video dur="5s" repeatCount="3" end="click; next.click" .../>
    <video dur="5s" repeatCount="3" end="click; next.click" .../>
</seq>
```

Figura 30. Exemplo de utilização do atributo *end* com os valores *click* e *next.click*

No exemplo da Figura 31, são apresentados em paralelo, durante 30 segundos, uma imagem, um texto e um áudio. Se o usuário clicar na imagem, será finalizada a execução do áudio.

```
<par dur="30s">
    <img id="mutebutton" src="mute.jpg"/>
    <text src="descricao.html" />
    <audio src="audio.au" end="mutebutton.click"/>
</par>
```

Figura 31. **Exemplo de uso do atributo *end* com o valor *mutebutton.click***

Para a transferência de elementos de mídia, o SMIL especifica o uso dos protocolos HTTP, RTSP ou RTP. A definição do protocolo a ser utilizado é feita pelo campo *src* na definição da mídia.

O protocolo HTTP não é o melhor protocolo a ser usado para transferência de mídias contínuas, como o vídeo, pois ele realiza uma transferência assíncrona do arquivo: todo o arquivo de vídeo é transferido para o cliente antes da apresentação.

Já com o RTSP, os servidores podem manter registrados quando, onde e como o conteúdo será transmitido. Este protocolo permite uma interação como no modo de operação de videocassetes, como as teclas *Play*, *Fast-forward*, *Pause* e *Stop*, e identifica os registros de horário, que são utilizados para a realização de buscas baseadas em tempo dentro de uma apresentação. O protocolo RTSP também aceita multidifusão para distribuir diversos arquivos simultaneamente.

O protocolo RTP fornece um serviço de transporte fim-a-fim para dados com características tempo-real. Tem algumas das características desejáveis para a comunicação multimídia. Ele é um protocolo leve, assim uma alta vazão pode ser obtida, e transporta informações usadas para realizar sincronizações intramídia e intermídia para aplicações multimídia.

A figura 32 apresenta um exemplo de utilização do protocolo RTSP para possibilitar o controle de apresentação de um áudio, sendo que o protocolo definido para o transporte do áudio é o RTP. Neste exemplo, o atributo *port* define a porta UDP na qual o áudio vai ser transferido via *multicast* e esse valor da porta pode variar entre os valores estipulados nesse atributo. O *transport* define o protocolo usado para transferir a mídia (RTP). O campo *rtpformat* determina uma lista de formatos do *payload* que podem ser usados na sessão, o primeiro valor é o *default.*

```
<audio src="rtsp://www.w3.org/test.rtp" port="49170-49171" transport="RTP/AVP" rtpformat="96,97,98" />
```

Figura 32. Exemplo de utilização do protocolo RTSP

Composição Serial e Paralela

O SMIL possui o conceito de composição para estruturar temporalmente a apresentação do documento. Uma composição pode conter objetos de mídia e outras composições, determinando a forma como seus componentes deverão ser exibidos. Existem dois tipos de composição, a composição paralela (*par*) e a composição seqüencial (*seq*). Numa composição paralela, os componentes são exibidos simultaneamente (Figura 33), enquanto em uma composição seqüencial os componentes são apresentados um após o outro (Figura 39).

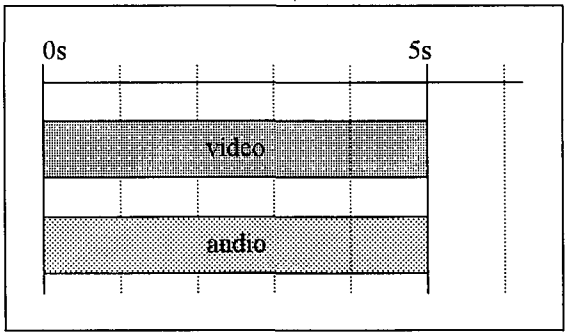


Figura 33. Linhas de tempo apresentando uma composição em paralelo

O *tag* <par> faz com que os objetos que venham entre os *tags* <par> e </par> sejam executados em paralelo. O código abaixo exemplifica o que foi ilustrado na linha de tempo da Figura 34.

```
...
    <body>
        ...
        <par title="Composicao 1 do seq">
            <video src="video.rm" id="video" region="esquerda"  title="video" alt="Amo Você" dur="5s"/>
            <audio src="audio1.rm" title="Parabéns a Você" dur="5s"/>
        </par>
        ...
    <body>
...
```

Figura 34. Fragmento do código da Figura 13, exemplificando o uso do *tag* <par>

No código da Figura 35 são adicionados ao elemento *par* os atributos *begin* e *dur*, fazendo com que a composição paralela comece a sua execução decorridos o tempo indicado no atributo *begin* (0 segundos) e durante o tempo especificado no atributo dur (33 segundos). Então, ao serem executados os elementos filhos dessa composição paralela, esses valores deverão ser considerados. Como o vídeo começará a sua apresentação decorridos 1 segundo (*begin*="1s") e levando 25 segundos para a sua execução (dur * *repeatCount*), então a última imagem do vídeo será congelada por 7 segundos (*fill*="freeze"), completando assim 33 segundos.

```
<par begin="0s" dur="33s">
    <video begin="1s" dur="10s" repeatCount="2.5" fill="freeze" .../>
</par>
```

Figura 35.* Uso do *tag* <par> com os atributos *begin* e *dur

No código da Figura 36 são adicionados no elemento *par* os atributos *begin*, *dur*, *repeatDur* e *fill*, fazendo com que a composição paralela comece a sua execução decorridos o tempo indicado no atributo *begin* (0 segundos) e durante o tempo especificado no atributo *repeatDur* (33 segundos). Então, ao serem executados os elementos filhos dessa composição paralela, o tempo estipulado no atributo *repeatDur* (tempo total da composição paralela) e o tempo estipulado pelo atributo *dur* (tempo de apresentação de um único elemento filho de uma composição paralela) deverão ser considerados. Ao ser executado o código da Figura 36 acontecerá a seguinte seqüência, que será finalizada quando forem completados 33 segundos:

- espera para o começo da apresentação do vídeo por um segundo;
- execução do vídeo por 9 segundos (dur * repeatCount);

- congelamento da última imagem do vídeo por 2 segundos.

```
<par begin="0s" dur="12s" repeatDur="33s" fill="freeze" >
    <video begin="1s" dur="5s" repeatCount="1.8" fill="freeze" .../>
</par>
```

Figura 36. Uso do *tag* <par> com os atributos *begin*, *repeatDur* e *fill*

Pode-se ter o *tag* <par> onde um (ou mais) de seus elementos tenha um tempo de início explícito. Para isso deve ser usado o atributo *begin*, que significa o valor de retardo do elemento. No exemplo a seguir, o vídeo é retardado em 1,4 segundos em relação ao começo do elemento *par*.

```
<par>
    <text src="carta.html" region="textos" dur="5s" />
    <video src="aniver.mpv" region="Videos" begin="1.4s" />
    <audio src="aniver.aiff" region="musicas" />
</par>
```

Figura 37. Exemplo de código lançando 3 elementos em paralelo em que um deles tem um tempo de início explícito

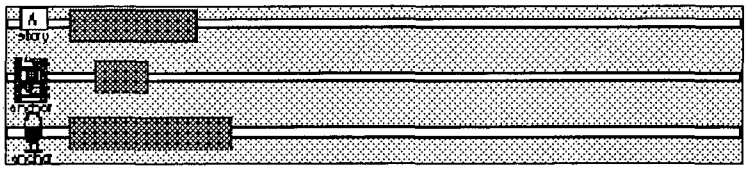


Figura 38. Linhas de tempo exemplificando o código da Figura 37

Um exemplo de composição seqüencial é fazer a *figura1* aparecer durante 2 segundos, seguida da *figura2* que será apresentada nos próximos 2 segundos e seguida da *figura3*, que também será apresentada durante 2 segundos, como mostra a linha de tempo abaixo:

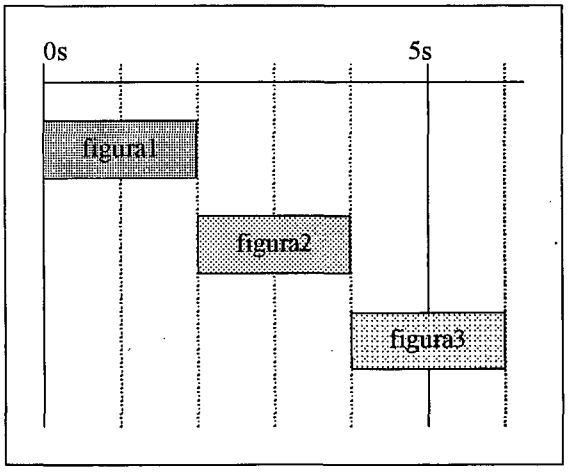


Figura 39. Linha de tempo exibindo uma composição em seqüência

O *tag* <seq> define uma **seqüência**. O que está incluído entre <seq> e </seq> são executados um após o outro. O código abaixo exemplifica o que foi ilustrado na linha de tempo da Figura 39.

```
...
    <body>
        <seq title="Composicao 2 do seq">
            <img src=" http://www.inf.ufsc.br/~adriana/figura1.jpg" id="graficos0" region="esquerda"
                titie="imagem 1" alt="Feiiz" dur="5s"/>
            <img src=" http://www.inf.ufsc.br/~adriana/figura2.jpg" id="graficos3" region="direita"
                title="imagem 2" alt="Aniversário" dur="5s"/>
            <img src=" http://www.inf.ufsc.br/~adriana/figura3.jpg" id="graficos4" region="inferior"
                title="imagem 3" alt="Mamãe" dur="5s"/>
        </seq>
    </body>
...
```

Figura 40. Fragmento do código da Figura 13, exemplificando o uso do *tag* <seq>

Pode-se combinar os *tags* <par> e <seq>, por exemplo, para lançar duas composições paralelas em seqüência. O código da Figura 41 e as linhas de tempo da Figura 42 exemplificam essa situação. São apresentados, em paralelo, um texto, um vídeo e um áudio, a seguir, após o término da apresentação dos elementos do primeiro *par*, são apresentados seqüencialmente outra composição paralela contento outro texto, outro vídeo e outro áudio.

```
<seq>
    <par>
        <text src="comunicado.html" region="textos" dur="5s" />
        <video src="aniver.mpv" region="Videos" />
        <audio src="aniver.aiff" region="musicas" />
    </par>
    <par>
        <text src="narrativa.html" region="textos" dur="2s" />
        <video src="festa.mpv" region="Videos" />
        <audio src="festa.aiff" region="musicas" />
    </par>
</seq>
```

Figura 41. Exemplo de código lançando duas composições paralelas em seqüência

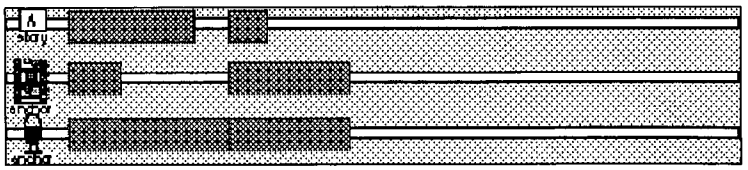


Figura 42. Linhas de tempo exemplificando o código da Figura 41

Tanto em composições paralelas quanto em seqüências, os atributos início e término dos elementos também podem ser especificados em relação ao início ou ao término de qualquer outro elemento da composição. No exemplo a seguir, o áudio é retardado em 2 segundos em relação ao início do elemento vídeo "v1".

```
<par>
    <text src="carta.html" region="textos" dur="5s" />
    <video id="v1" src="aniver.mpv" region="Videos" begin="1.4s" />
    <audio src="aniver.aiff" region="musicas" begin="v1.begin+5s" />
</par>
```

Figura 43. Exemplo de código lançando 3 elementos em paralelo em que o tempo de início de um elemento é relativo a um outro elemento

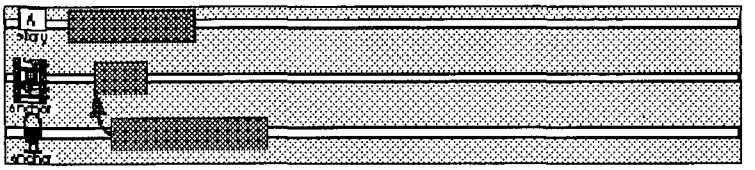


Figura 44. Linhas de tempo exemplificando o código da Figura 43

Adicionalmente, as composições paralelas possuem o atributo *endsync*, que determina o término da composição em relação ao término de um de seus componentes. O valor desse atributo pode identificar um dos componentes, indicando que todos os demais serão terminados quando o componente selecionado terminar. O atributo pode ainda assumir os valores *primeiro (first), elemento referenciado (id-ref)* ou *último (last)*.

O valor *first* especifica que a apresentação irá terminar quando o primeiro elemento multimídia terminar.

```
<par endsync="first">
    <text src="carta.html" region="textos" dur="5s" />
    <video id="v1" src="aniver.mpv" region="Videos" begin="1.4s" />
    <audio src="aniver.aiff" region="musicas" begin="id(v1)(0.5s)" />
</par>
```

Figura 45. Exemplo de uso do *tag* <par endsync="first">

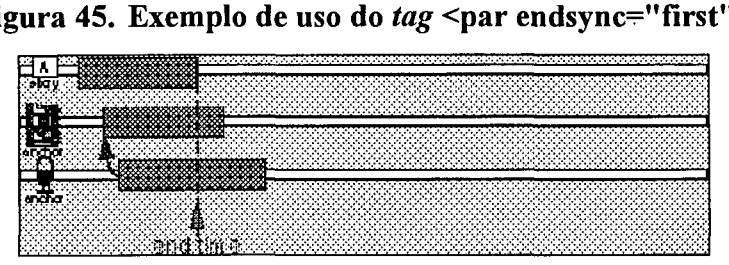


Figura 46. Linhas de tempo exemplificando o código da Figura 45

O valor *id-ref* especifica que a apresentação da composição paralela irá terminar quando o elemento multimídia referenciado terminar a sua execução.

```
<par endsync="id(v1)">
    <text src="carta.html" region="textos" dur="5s" />
    <video id="v1" src="aniver.mpv" region="Videos" begin="1.4s" />
    <audio src="aniver.aiff" region="musicas" begin="id(v1)(0.5s)" />
</par>
```

Figura 47. Exemplo de uso do *tag* <par endsync="id(v1)">

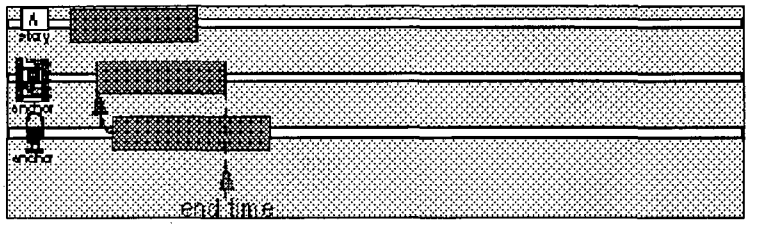


Figura 48. Linhas de tempo exemplificando o código da Figura 47

O valor *last* especifica que a apresentação irá terminar quando o último elemento multimídia terminar.

```
<par endsync="last">
    <text src="carta.html" region="textos" dur="5s" />
    <video id="v1" src="aniver.mpv" region="Videos" begin="1.4s" />
    <audio src="aniver.aiff" region="musicas" begin="id(v1)(0.5s)" />
</par>
```

Figura 49. Exemplo de uso do *tag* <par endsync="last"> slip

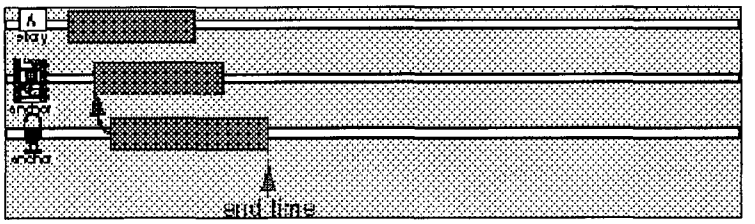


Figura 50. Linhas de tempo exemplificando o código da Figura 49

A linguagem SMIL permite manter a sincronização contínua entre elementos de mídia, já que o processamento dos componentes estão sujeitos a variações temporais (causadas pelo retardo da rede, tempo de acesso à base de dados, etc.) que provocam desvios temporais nos esquemas de sincronização. Um exemplo de sincronização contínua é a sincronização labial, onde o áudio e o vídeo de uma pessoa falando deveria ser sincronizado. Para isso, SMIL Boston definiu os seguintes atributos: o *syncBehavior*, o *syncTolerance* e o *syncMaster*.

O atributo *syncBehavior* define o nível de sincronização contínua. Os valores possíveis para este atributo são: *canSlip* e *locked*. O atributo *canSlip* permite que o elemento associado deslize com relação à sincronização. O atributo *locked* força o elemento associado a manter a sincronização com relação ao tempo do elemento pai.

No exemplo a seguir, se o vídeo ou o áudio tiver que parar devido ao congestionamentos na rede, o elemento par "speech" sofrerá uma pausa para manter a sincronização. Porém, o resto do documento, inclusive a animação, continuará a ser

executado normalmente. O atributo "locked" permite manter a sincronização labial, o atributo "speech", permite reajustar.

```
<par>
    <animation src="..." /> ...
    <par id="speech" syncBehavior="canSlip" >
        <video src="speech.mpg" syncBehavior="locked" />
        <audio src="speech.au" syncBehavior="locked" />
    </par> ...
</par>
```

Figura 51. Exemplo de uso do atributo syncBehavior com os valores canSlip e locked

O atributo *syncTolerance* define um valor de relógio para a tolerância de sincronização contínua do elemento e todos os seus descendentes. Ou seja, este atributo define um nível de tolerância em nível de sincronização contínua (por exemplo, a diferença máxima do tempo de apresentação da imagem e da voz em uma sincronização labial).

O atributo *syncMaster* permite definir o elemento que vai comandar a sincronização contínua.

O comando de tempo *excl* surgiu da necessidade de se apresentar conteúdo opcional, que pode ser mostrado em resposta ao usuário ou a outro evento, sem carregar uma outra apresentação SMIL. No exemplo seguinte são mostrados inicialmente os botões 1 e 2. Se o usuário clicar no botão 1 será apresentada a composição paralela identificada por "p1"; se clicar o botão 2, será apresentada a composição paralela identificada por "p2". Se o usuário seleciona "p1" e seleciona "p2", o "p1" é parado e "p2" é apresentado. Se nada é selecionado, o conteúdo opcional nunca será apresentado.

```
<par>
    <excl>
        <par id="p1">
        ...
        </par>
        <par id="p2">
        ...
        </par>
    </excl>
        <a href="p1"><img src="botao1.jpg"/></a>
        <a href="p2"><img src="botao2.jpg"/></a>
</par>
```

Figura 52. Exemplo de uso do *tag* <excl>

No exemplo a seguir, somente um dos três elementos pares, filhos do comando *excl*, pode ser executado em um determinado tempo. O áudio, elemento filho de cada

elemento *par*, é definido para iniciar a sua execução quando o vídeo id="vid" for executado; os dois deverão ser apresentados em paralelo. Serão apresentadas ao usuário três opções de línguas pela qual ele pode ouvir os áudios: em português, inglês ou francês. Se o usuário selecionar o "portuguesBtn" será apresentado em paralelo um vídeo e o áudio no idioma português.

```
<par>
    <video id="vid" .../>
    <excl>
        <par begin="inglesBtn.click" >
            <audio begin="vid1.begin" src="ingles.au" />
        </par>
        <par begin="francesBtn.click" >
            <audio begin="vid1.begin" src="frances.au" />
        </par>
        <par begin="portuguesBtn.click" >
            <audio begin="vid1.begin" src="portugues.au" />
        </par>
    </excl>
</par>
```

Figura 53. Exemplo de uso do *tag* <excl> onde o início de apresentações das mídias é definido em relação a uma outra mídia

Pode-se definir particularidades ao comando *excl*, utilizando o elemento *priorityClass*, com os atributos *peers*, *higher* e *lower*.

O atributo *peers* controla como os elementos filhos do comando *excl* se comportarão se outro elemento filho for ativado enquanto um outro está ativo. Este atributo admite os seguintes valores:

- o *stop*, o elemento que estava ativo é parado (*default*);
- o *pause*, o elemento que estava ativo sofrerá uma pausa e retomará sua apresentação após o término do elemento que foi ativado;
- o *defer*, o elemento que deseja ser ativado será adiado até o elemento ativo terminar a sua apresentação;
- o *never*, não permite o início de outro elemento enquanto algum outro está sendo ativado.

No exemplo da Figura 54, o usuário terá a possibilidade de selecionar qualquer um dos áudios para ser executado (som1, som2, ..., somN) através dos botões (botão1, botão2, ..., botãoN). Se o usuário selecionar o áudio "som1", através do "botão1" e, ainda durante a sua execução, selecionar o "som3" através do "botão3", isso fará com

que a execução do "som1" sofra uma pausa e retome sua apresentação após o término da execução do elemento "som3", pois foi adicionado ao atributo *peers* o valor *pause*.

```
<par>
    <excl dur="indefinite">
        <priorityClass peers="pause">
            <audio id="som1" .../>
            <audio id="som2" .../>
            <audio id="som3" .../>
            ...
            <audio id="somN" .../>
        </priorityClass>
    </excl>
    <a href="som1"><img src="botao1.jpg"/></a>
    <a href="som2"><img src="botao2.jpg"/></a>
    <a href="som3"><img src="botao3.jpg"/></a>
    ...
    <a href="somN"><img src="botaoN.jpg"/></a>
</par>
```

Figura 54. **Exemplo de uso do tag <excl> com o elemento *priorityClass***

Âncoras

A especificação de âncoras e elos de navegação numa apresentação SMIL se dá através dos *tags a* e *area* e a navegação nos mesmos é sempre ativada por uma interação do usuário.

Um dos atributos de um elo SMIL é o atributo *show*, que controla o comportamento do objeto de origem quando o elo é percorrido. O valor desse atributo pode ser: *new*, que especifica que a apresentação do objeto de destino acontecerá em outro contexto, sem afetar o objeto de origem; *replace*, que especifica que a apresentação do objeto de origem será substituída pela apresentação do destino, e; *pause*, que especifica que a apresentação do objeto de origem será suspensa e depois retomada, sendo o objeto de destino apresentado em outro contexto. O valor predefinido para *show* é *replace*.

O código da Figura 55 mostra o elemento *a* contendo o atributo *show* com o valor *new*. Quando o usuário clicar no vídeo "zoomin.mpv" abrirá uma nova janela contendo a apresentação "archives-dcab.smi". A apresentação atual não será afetada. A Figura 56 ilustra o código da Figura 55.

```
<a show="new" href="arquivo.smi">
    <video src="aumentar.mpv" region="Videos" />
</a>
```

Figura 55. Exemplo de uso do *tag* <a> com o atributo *show* e o valor desse atributo *new*

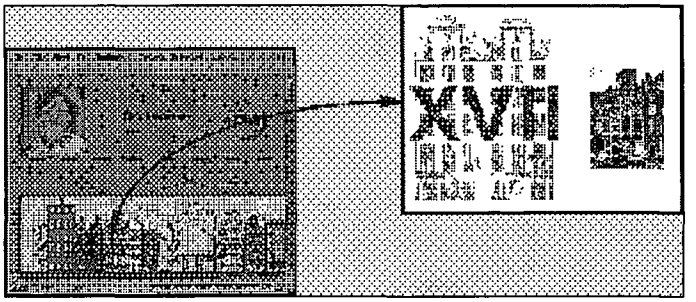


Figura 56. Exemplo do código da Figura 55

O código da Figura 57 mostra o elemento *a* contendo o atributo *show* com o valor *replace*. Quando o usuário clicar na imagem "interacao.gif" a apresentação atual será interrompida e será apresentado a "figura4.gif".

```
<a show="replace" href="http://www.inf.ufsc.br/~adriana/figura4.gif">
    <img src="http://www.inf.ufsc.br/~adriana/interacao.gif" id="graficos10" region="inferior"
        title="imagem 4" dur="20s"/>
</a>
```

Figura 57. **Fragmento do código da Figura 13, exemplificando a utilização do *tag* <a> com o atributo *show* e o valor desse atributo *replace***

A funcionalidade do elemento *a* é muito semelhante do elemento *a* de HTML 4.0. Os elementos *a* não podem ser colocados uns dentro dos outros, além disso, têm que possuir o atributo *href* que contém o URI do destino da ligação.

Uma ligação pode acessar o interior de uma outra apresentação. No código da Figura 58 tem-se duas apresentações: apresentação_1 e apresentação_2. A primeira apresenta um vídeo que ao ser selecionado vai para um determinado instante temporal (rotulado por *next*) da apresentação_2. Na apresentação_2, tem-se a exibição de um vídeo e após a apresentação em paralelo de dois vídeos e dois textos. Como no atributo *href* do *tag* <a> da apresentação_1 foi colocado, além do endereço da ligação, a especificação "#next", então quando o usuário clicar no vídeo "graficos.imf" da apresentação_1, acessará a apresentação_2 a partir do elemento identificado com o "#next". Será apresentado então os elementos que estão posicionados depois do "#next", um vídeo e dois textos em paralelo, em vez de ser apresentado um vídeo e depois a apresentação em paralelo de dois vídeos e dois textos. A apresentação de destino começa como se a apresentação tivesse sido avançada rapidamente até o início do elemento designado pelo fragmento.

```
Apresentação_1:

    <a href=" http://www.inf.ufsc.br/~adriana/apresentacao2#next">
```

```
        <video src="rtsp:/ www.inf.ufsc.br/~adriana/graficos.imf"/>
    </a>

Apresentação_2 (http://www.inf.ufsc.br/~adriana/apresentacao):
    ...
    <seq>
        <video src="rtsp://foo.com/graficos.imf"/>
        <par>
            <video src="rtsp:// www.inf.ufsc.br/~adriana/v2.rm" region="janeia_esquerda"/>
            <video id="next" src="rtsp:// www.inf.ufsc.br/~adriana/v1.rm" region="janeia_direita"/>
            <text src="rtsp:// www.inf.ufsc.br/~adriana/titulo1.html" region="titulo_esquerdo"/>
            <text src="rtsp:// www.inf.ufsc.br/~adriana/titulo2.rtx" region="titulo_direito"/>
        </par>
    </seq>
    ...
```

Figura 58. Exemplo de uso do *tag* <a> com o conector #

A funcionalidade do elemento *a* é restrita, pois só permite associar um vínculo a um objeto de mídia completo. O elemento *area* de HTML demonstrou que é útil associar vínculos com porções de espaço da exibição visual de um objeto.

A semântica do elemento *area*, que substitui o elemento *anchor* de SMIL 1.0, é semelhante ao HTML. O elemento *area* permite decompor um objeto multimídia em sub-partes espaciais ou temporais. As subpartes espaciais utilizam o atributo *coords* (Figura 59) e as subpartes temporais utilizam os atributos *begin* e *end* (Figura 61). Este elemento permite também designar um **pedaço de um objeto multimídia** como destino de uma ligação, utilizando o atributo "id" (Figura 63). Além disso, permite decompor um vídeo em segmentos temporais (Figura 64).

Se o usuário clicar o lado esquerdo da imagem do "botao2.jpg", do exemplo da figura abaixo, ativará a página "http://www.inf.ufsc.br" e se clicar o lado direito, ativará a página "http://www.inf.ufsc.br/~adriana". A Figura 60 ilustra essa operação.

```
<img src="botao2.jpg" id="graficos21" region="botao2" >
    <area href="http://www.inf.ufsc.br" coords="0%,0%,50%,50%"/>
    <area href="http://www.inf.ufsc.br/~adriana" coords="50%,50%,100%,100%"/>
</img>
```

Figura 59. Exemplo de uso do atributo *coords* no elemento *area*

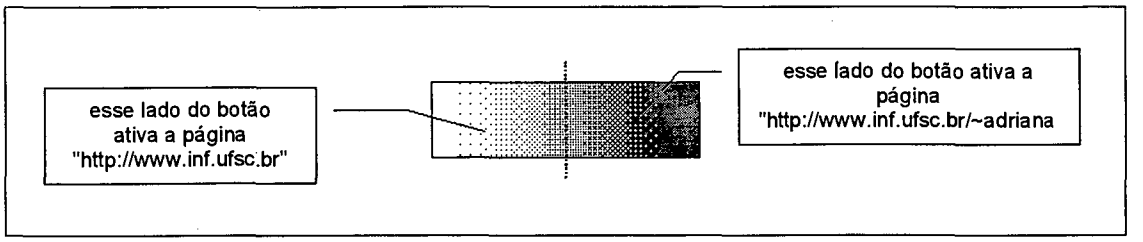


Figura 60. Exemplo ilustrativo do código da Figura 59

No código da Figura 61, se o usuário clicar no vídeo "Calmo" localizado no endereço "http://www.inf.ufsc.br/~adriana/" nos primeiros 5 segundos, ativará a página "http://www.inf.ufsc.br/~adriana/AudioVideo" e se clicar nos outros 5 segundos, ativará a página "http://www.inf.ufsc.br/~adriana/Estilo".

```
<video src=" http://www.inf.ufsc.br/~adriana/Calmo">
    <area href=" http://www.inf.ufsc.br/~adriana/AudioVideo" begin="0s" end="5s"/>
    <area href=" http://www.inf.ufsc.br/~adriana/Estiio" begin="5s" end="10s"/>
</video>
```

Figura 61. Exemplo de uso dos atributos *begin* e *end* no elemento *area*

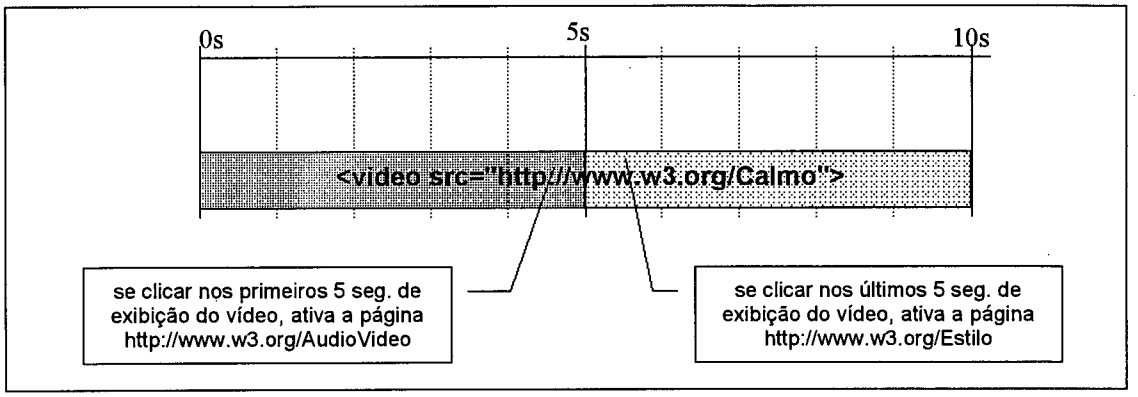


Figura 62. Exemplo ilustrativo do código da Figura 61

No código da Figura 63, o vídeo da apresentação 2 foi dividido em dois intervalos temporais e identificado com um nome exclusivo: nos primeiros 5 segundos, com o identificador id="joe" e nos segundos posteriores, com o identificador id="tim". Na apresentação 1, o destino da ligação é o quinto segundo de "http://www.inf.ufsc.br/~adriana/apresentação2", os primeiros 5 segundos serão descartados.

```
Apresentação 1:

<a href=" http://www.inf.ufsc.br/~adriana/apresentação2#tim">
    <video id="graficos" src="rtsp://foo.com/graficos.imf" region="janela_esquerda"/>
</a>

Apresentação 2:

<video src=" http://www.inf.ufsc.br/~adriana/Caimo">
    <area id="joe" begin="0s" end="5s"/>
    <area id="tim" begin="5s" end="10s"/>
</video>
```

Figura 63. Exemplo de uso do elemento *area* associando o destino de uma ligação a pedaços temporais

A Figura 64 apresenta como um vídeo é decomposto em segmentos temporais. É tomada como exemplo a estrutura temporal de uma entrevista, em que uma entrevistadora faz uma pergunta seguida de uma resposta de uma pessoa, exposta através de fragmentos.

```
<smil>
    <body>
        <video src="video" title="Entrevistando Tom Cruise" >
            <seq>
                <area id="primeiraPergunta" dur="20s" title="primeira pergunta" />
                <area id="primeiraResposta" dur="50s" title="primeira resposta" />
            </seq>
        </video>
    </body>
</smil>
```

Figura 64. Exemplo de um vídeo decomposto em segmentos temporais

Elementos Switch, user_attributes e u_group

O elemento *switch* é o único elemento da linguagem SMIL que pode ser usado tanto no cabeçalho quanto no corpo do documento. Esse comando provê meios para definir comportamentos alternativos da apresentação de um documento devido aos seguintes fatores:

- conexões diferentes podem ter velocidades diferentes;
- estações de trabalho podem ter capacidades diferentes;
- usuários diferentes podem ter tarefas diferentes ou diferentes preferências em relação às mídias.

O autor do documento especifica um conjunto de elementos alternativos, dos quais no máximo um do conjunto de elementos alternativos será escolhido, de acordo com um ou mais testes a serem realizados em tempo de execução. Entre esses fatores estão, por exemplo, a largura de banda disponível para a transmissão dos objetos (Figura 65), recursos de áudio com taxas de bits diferentes (Figura 66), o idioma desejado pelo usuário (Figura 67), se o usuário prefere a apresentação de dublagem ou a apresentação de subtítulos (Figura 68), a apresentação de legendas (Figura 68), as dimensões da janela de apresentação e o número de cores disponíveis (Figura 69).

A primeira condição de um *tag switch* cujos atributos de teste forem avaliados como verdadeiros é executada, por isso é preciso colocar os elementos de melhor qualidade primeiro.

A medição da largura de banda é especificada para cada aplicação, pois as aplicações tanto podem empregar medições sofisticadas da conectividade ponto-a-ponto, como simples definições estáticas controladas pelo utilizador. Este último caso poderia ser utilizado, por exemplo, para efetuar uma escolha com base no tipo de ligação dos usuários à rede. Os valores mais característicos para os utilizadores de modems seriam 14.400, 28.800, 56.000 bit/s. É avaliado como "true" se a taxa de bits disponível no sistema for igual ou superior ao valor especificado.

```
<par>
    <text .../>
    <switch>
            <par system-bitrate="40000">
                ...
            </par>
            <par system-bitrate="24000">
                ...
            </par>
            <par system-bitrate="10000">
                ...
            </par>
    </switch>
</par>
```

Figura 65. Exemplo de uso do atributo de teste *system-bitrate*

Os elementos no interior de um *switch* podem constituir uma combinação de elementos. Com isso pode-se, por exemplo, especificar uma faixa de áudio alternativa, levando-se em consideração a velocidade do modem.

```
<switch>
    <audio src="audio-otima-qualidade" system-bitrate="16000" />
    <audio src="audio" system-bitrate="8000" />
</switch>
```

Figura 66. Exemplo de uso de uma combinação de elementos no interior de um *switch*

No exemplo que se segue, um recurso de áudio está disponível tanto em francês como em inglês. Com base no idioma preferido pelo usuário, o reprodutor pode escolher um dos recursos de áudio.

```
<switch>
    <audio src="audio-em-frances" system-language="fr"/>
    <audio src="audio-em-ingles" system-language="en"/>
</switch>
```

Figura 67. Exemplo de uso do atributo de teste *system-language*

No exemplo seguinte, um filme no idioma francês está disponível com dublagens e subtítulos em inglês, alemão e holandês. No primeiro *switch* o usuário poderá optar qual o idioma que ele quer assistir as dublagens: inglês, alemão ou holandês. Se ele não optar por nenhum desses idiomas, a dublagem será apresentada no

idioma francês. No segundo *switch* o usuário poderá optar se ele quer assistir a apresentação com subtítulos em inglês, alemão ou holandês. Se ele não optar por nenhum desses, será dada a oportunidade do usuário escolher se ele quer ou não assistir a apresentação com legendas no idioma francês.

```
...
<par>
    <switch>
        <audio src="audio-em-ingles.rm" systemLanguage="en" systemOverdubOrSubtitle="overdub"/>
        <audio src="audio-em-holandes.rm" systemLanguage="de" systemOverdubOrSubtitle="overdub"/>
        <audio src="audio-em-frances.rm"/>
    </switch>
    <video src="movie-vid.rm"/>
    <switch>
        <textstream src="texto-em-ingles.rt" systemLanguage="en" systemOverdubOrSubtitle="subtitle"/>
        <textstream src=" texto-em-holandes.rt" systemLanguage="de" systemOverdubOrSubtitle="subtitle"/>
        <!-- A apresentadação de legendas no idioma francês-->
        <textstream src=" texto-em-frances.rt" systemCaptions="on"/>
    </switch>
</par>
...
```

Figura 68. Exemplo de uso dos atributos de teste *systemOverdubOrSubtitle* e *systemCaptions*

No próximo exemplo, a apresentação contém partes alternativas como as dimensões da janela de apresentação e o número de cores disponíveis, na qual o reprodutor escolherá uma das alternativas.

```
...
<par>
    <text .../>
    <switch>
        <par system-screen-size="1280X1024" system-screen-depth="16">
            ....
        </par>
        <par system-screen-size="640X480" system-screen-depth="32">
            ...
        </par>
        <par system-screen-size="640X480" system-screen-depth="16">
            ...
        </par>
    </switch>
</par>
...
```

Figura 69.* Exemplo de uso dos atributos de teste *system-screen-size* e *system-screen-depth

Os elementos *user_attributes* e *u_group* estendem a noção da definição de atributos de teste de usuário. Atributos de teste de usuário (*user_attributes*) permitem aos autores definir grupos relacionados com os objetos de mídia. Esses grupos são definidos no cabeçalho do documento SMIL, através do elemento *user_attributes* e referenciados dentro de um objeto de mídia, através do elemento *u_group*.

```
1 <smil>
2  <head>
```

```
3       <layout>
4           <!—definição das regiões -->
5       </layout>
6       <user_attributes>
7           <u_group id="hol_aud" u_state="RENDERED" title="Audio em holandes" override="allowed" />
8           <u_group id="ing_aud" u_state="NOT_RENDERED" title="Audio em Ingles" override="allowed" />
9           <u_group id="hol_txt" u_state="NOT_RENDERED" title="Texto em holandes" override="allowed" />
10          <u_group id="ing_txt" u_state="NOT_RENDERED" title="Texto em Ingles" override="allowed" />
11      </user_attributes>
12  </head>
13  <body>
14      ...
15      <par>
16          <video src="anunciar.rm" region="a"/>
17          <text src="noticias_manchete.html" region="b"/>
18          <audio src="narrativa_1_hol.rm" u_group="hol_aud"/>
19          <audio src="narrativa_1_ing.rm" u_group="ing_aud"/>
20          <text src="narrativa_1_hol.html" u_group="hol_txt" region="c"/>
21          <text src="narrativa_1_ing.html" u_group="ing_txt" region="d"/>
22      </par>
23      ...
24  </body>
25  </smil>
```

Figura 70. **Exemplo de uso dos elementos *u_group* e *user_attributes***

Da linha 6 a 11 são definidos os grupos disponíveis. Cada grupo contém um identificador, um título (que pode ser usado pelo usuário da interface para nomear o grupo) e a definição do estatal inicial (*u_state*).

Na linha 7, um <u_group> nomeado "hol_aud" é definido para executar um áudio em holandês.

Entre as linhas 15 e 22, o elemento <par> de SMIL é usado para identificar uma porção de uma apresentação. Nesse <par>, um único vídeo (linha 16) é acompanhado por dois fluxos de áudio (18,19) e dois textos em movimentos (20,21), um em inglês e outro em holandês. O <par> também contém um título exibindo um cabeçalho.

A vantagem da utilização do elemento *user_attributes* é que o usuário poderá escolher as combinações de fluxos que ele quer executar, por exemplo, uma áudio em inglês e um vídeo em holandês. É importante observar que, se o usuário selecionar dois áudios, o *player* precisará determinar como estes serão processados para entrega.

O elemento *prefetch* dará uma sugestão ou indicará ao usuário que um recurso de mídia será usado no futuro, e questiona se o usuário gostaria que parte ou todo recurso fosse buscado de antemão para fazer a apresentação do documento mais uniforme. Os usuários podem ignorar esse elemento causando uma interrupção na apresentação do documento quando o recurso for utilizado.

No exemplo da Figura 71, a imagem pode ser exibida imediatamente depois do fim do vídeo. Como o texto é uma mídia discreta, ele é imediatamente baixado e, enquanto o usuário está lendo o texto, o elemento *prefetch* utiliza os recursos da rede para carregar a imagem. Após ter sido carregada a imagem, a composição paralela é finalizada e o vídeo começa a ser apresentado. Quando o vídeo chegar ao fim a imagem será mostrada imediatamente, pois ela estará na memória.

```
<smil>
    <body>
        <seq>
            <par>
                <prefetch id="intervalo" src=" http://www.inf.ufsc.br/~adriana/logo.gif"/>
                <text id="t1" src=" http://www.inf.ufsc.br/~adriana/espere.html" fill="freeze"/>
            </par>
            <video id="v1" src="rtsp://www.inf.ufsc.br/~adriana/video.mpg"/>
            <image src=" http://www.inf.ufsc.br/~adriana/logo.gif" fill="freeze"/>
        </seq>
    </body>
</smil>
```

Figura 71. **Exemplo de uso do elemento *prefetch***

3.4 Players, Editores e Servidores SMIL

Os *Players* SMIL são aplicativos clientes que recebem e exibem apresentações multimídia integradas executadas na linguagem SMIL. Os editores permitem ao autor criar um documento SMIL sem precisar conhecer a sintaxe desta linguagem. Os servidores SMIL são responsáveis pelo fornecimento de canais de conteúdo e apresentações para os clientes.

Os *Players*, editores e servidores apresentados abaixo são para o SMIL 1.0, eles ainda não existem para o SMIL Boston.

3.4.1 Players

Os *players* SMIL usados com mais freqüência são o GRiNS (*Graphical iNterface to SMIL*), o SOJA, o RealPlayer G2 e o RealPlayer Plus G2.

O GRiNS combina um editor SMIL e um *player* em um sistema integrado para trabalhar com apresentações em SMIL. O componente editor do GRiNS é utilizado para criar os códigos necessários para as apresentações, enquanto o componente *player*, para executá-los. O GRiNS está disponível para sistemas Macintosh, Windows e UNIX pelo

National Research Institute for Mathematics and Computer Science in the Netherlands (www.cwi.nl/grins).

O SOJA foi escrito por HELIO (associação francesa com sede em Melun, França) em 1998. O *player* é grátis. É um *applet* que exibe SMIL em uma página web ou em uma janela separada. O SOJA aceita os formatos de imagens: GIF, JPEG e os formatos de áudio: AU e AUZ (AU zipado). O SOJA não usa "streaming": tem-se que esperar até que cada um dos componentes tenha sido totalmente baixado, com isso o tempo de partida da apresentação é maior, mas a apresentação nunca pára. O *player* está disponível no site http://www.helio.org/products/smil/download/mirrors.html.

A Real participou na criação da linguagem SMIL e lançou o RealPlayer G2 em julho de 1998. Os principais formatos de imagens suportadas são GIF e JPEG e áudio são AU, WAV e MIDI. O RealPlayer G2 é um *player* repleto de recursos compatíveis com a SMIL, assim como outros formatos de fluxo de mídia da RealNetwork, como o RealText e o RealPix. O *player* oferece controles, é de fácil personalização e é gratuito. O RealPlayer Plus G2 é uma versão comercial mais completa e aperfeiçoada para reprodução de som estéreo e de alta resolução. Ambos os *players* estão disponíveis para Windows 95, 98 e NT ou para a plataforma Macintosh, no site www.real.com/g2.

O RealPlayer usa *streaming* para exibir apresentações. Isso significa que não é necessário esperar que todos os componentes sejam carregados para exibir algo na tela. A apresentação é lida e exibida ao mesmo tempo, graças a um *buffer*. O *player* trabalha melhor quando acessa um arquivo SMIL fornecido por um *RealServer*, pois a exibição da mídia é em tempo real.

3.4.2 Editores

As ferramentas de autoria que geram automaticamente o código SMIL usadas com mais freqüência são o SMIL Composer da Sausage e o GriNS da Oratrix Development.

O SuperTool SMIL Composer permite criar conteúdos multimídia sincronizados sem que o autor conheça o código SMIL.

A abordagem de autoria que esse *software* utiliza está mais próxima da linguagem *scripting*, com a facilidade de ser orientada a *menu* (Figura 72), mas

perdendo uma das principais vantagens dessa abordagem de autoria: o poder de expressão. Muitos comandos que são definidos na especificação SMIL ela não suporta. *Tags* como *a e anchor* e atributos como *endsync* não possuem correspondentes nesse editor.

Outra desvantagem desse editor é que a composição temporal dos componentes é difícil de identificar: ele não ilustra de maneira gráfica a semântica das relações temporais, dificultando a especificação das restrições temporais e aumentando o tempo de criação do documento.

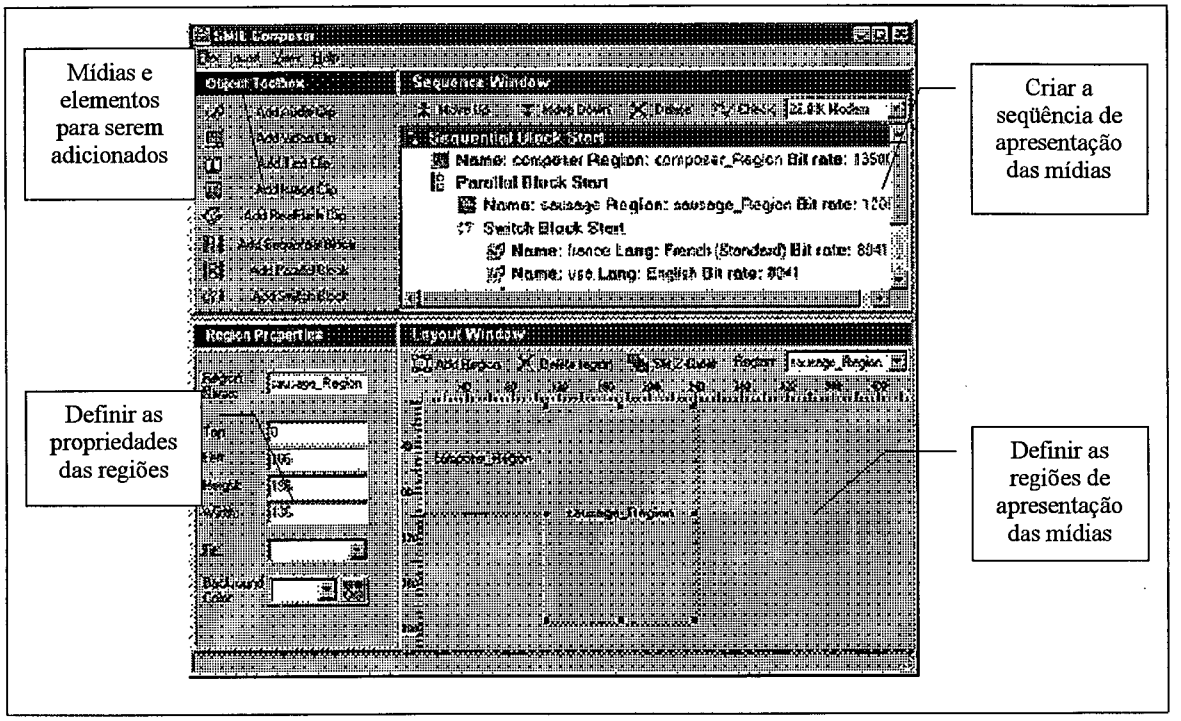


Figura 72. Tela do editor SMIL Composer

O GRiNS Editor utiliza a abordagem de autoria de documentos multimídia linha temporal. O autor tem uma visão muito clara das informações que serão apresentadas e em que momento – Figura 73. Para facilitar ainda mais a autoria de um documento SMIL, o GRiNS oferece outras visões do mesmo arquivo: das regiões (Figura 74), do arquivo fonte SMIL (Figura 75), da estrutura (Figura 76) e das ligações.

O GRiNS Editor é um *software* fácil de aprender e manipular. Todos os comandos que são definidos na especificação SMIL possuem, de forma gráfica, funções

correspondentes nesse editor. Além disso, apresenta facilidades para desenvolver aplicações complexas.

Uma das desvantagens desse editor é que ele permite especificar somente o alinhamento temporal ideal das apresentações, na medida em que ele define pontos de partida e fim ideais das apresentações dos componentes do documento. Não é possível especificar métodos de tolerância da sincronização ou ações a perdas de sincronismo.

Outra desvantagem é que o GRiNS requer o conhecimento exato da duração das apresentações. Geralmente o autor deve manipular os segmentos de informação de maneira manual a fim de obter o comportamento temporal desejado. Isto através da edição dos dados ou pela mudança da velocidade de apresentação.

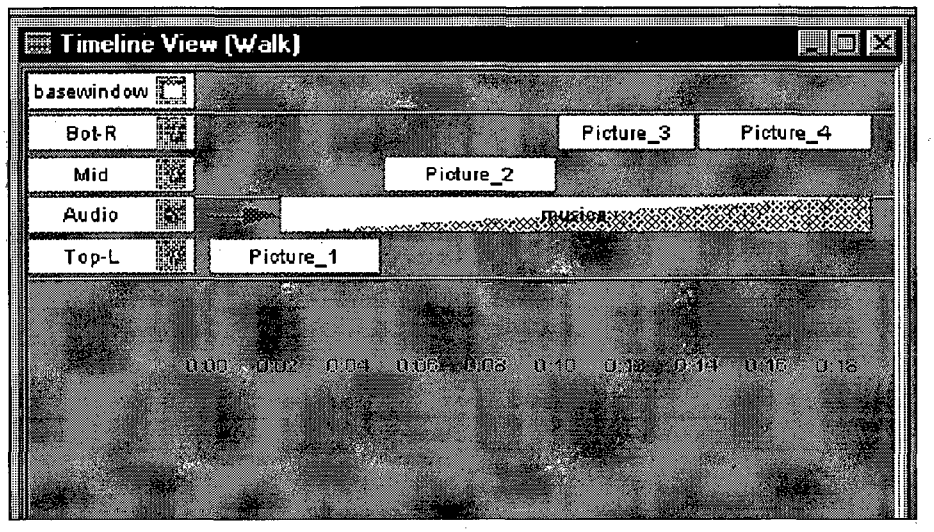


Figura 73. Visão linha temporal de um arquivo SMIL no GRiNS

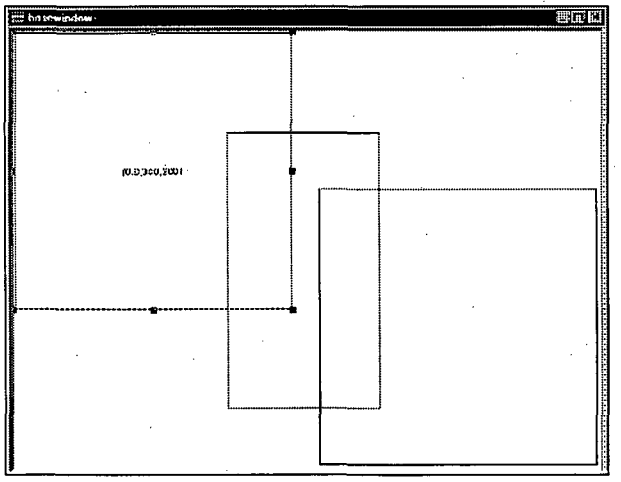


Figura 74. Visão das regiões de um arquivo SMIL no GRiNS

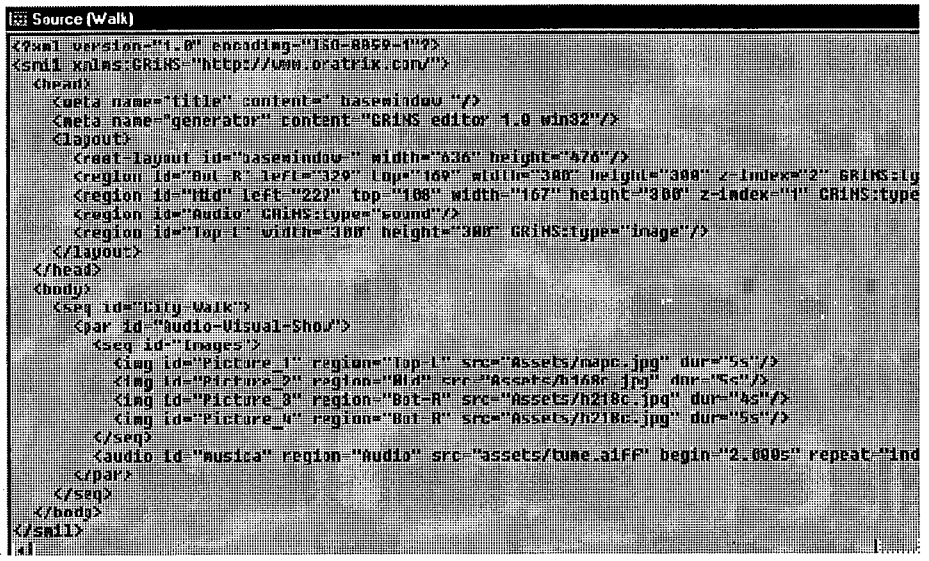


Figura 75. Visão do arquivo fonte SMIL no GRiNS

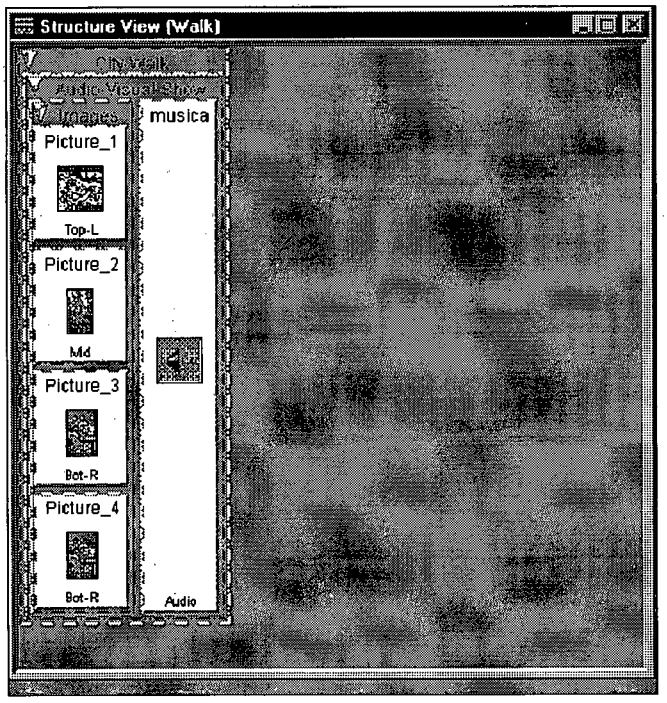


Figura 76. Visão da estrutura de um arquivo SMIL no GriNS

3.4.3 Servidores

Como a SMIL é uma tecnologia muito recente, ainda não existem muitos servidores compatíveis. Servidores multimídia especialmente configurados são projetados para operar com diversos formatos de mídia, incluindo texto, áudio, vídeo e animação; além disso, eles reconhecem *links* de hipertexto. Dessa forma, é possível

inserir *links* dentro de uma apresentação que dão acesso a outras, ou a outros arquivos de mídia. Por exemplo, uma apresentação sobre John Fitzgerald Kennedy (JFK) poderia oferecer *links* para informações resumidas da vida dele.

O *RealServer*, atualmente um dos poucos disponíveis, apresenta uma variedade de configurações projetadas para necessidades específicas. Como há uma versão específica para os provedores de acesso, pode-se achar que a conta do usuário pode ser atualizada para incluir compatibilidade com apresentações em SMIL. Todavia, para trabalhar com a SMIL, a versão do *RealServer* precisará ser a última compatível com o G2.

Embora a própria SMIL seja uma tecnologia aberta, alguns dos *players* e servidores usam técnicas proprietárias para manipular o fluxo e a codificação multimídia. Isso significa que se pode ficar atrelado a um determinado fabricante depois de adquirir a solução de publicação e distribuição SMIL.

3.5 Conclusões

A idéia básica da linguagem SMIL é nomear componentes do documento, como texto, imagens, áudio e vídeo como URIs, e escalonar suas apresentações ou em paralelo ou em seqüência.

Esta linguagem tem sido projetada de maneira a facilitar a autoria de apresentações multimídia na Web utilizando-se apenas um editor de texto, pois ela utiliza a abordagem de autoria de linguagens *scripting*. Estes modelos têm um poder de expressão muito grande, mas a especificação da composição de um documento multimídia na forma textual é difícil de produzir e modificar. Além disso, a composição temporal dos componentes é difícil de identificar.

Os modelos gráficos têm a vantagem de ilustrar de maneira gráfica a semântica das relações temporais. Este tipo de modelo simplifica a especificação das restrições temporais e reduz o tempo de criação do documento.

No próximo capítulo será apresentado o HTSPN, que facilita a autoria de apresentações multimídia por ser um sistema gráfico e, além disso, por ser um método

formal que permite realizar uma análise do documento a fim de verificar inconsistências temporais.

Capítulo 4. HTSPN

O HTSPN (*Hierarchical Time Streams Petri Nets*) é um modelo formal, baseado no no modelo de Redes de Petri a arco temporizado, que possibilita 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.

A fim de permitir a passagem automática de uma especificação (uma descrição abstrata) para uma representação final de documentos hipermídia, em [Willrich, 96b], os autores propuseram uma extensão ao modelo HTSPN. Esta extensão, chamada de *Interpreted* HTSPN (I-HTSPN) permite a especificação completa de documentos hipermídia e a produção automática de representações ISO MHEG dos documentos especificados. Estas representações MHEG podem então ser transferidas e apresentadas em um sistema aberto multimídia. Devido a pouca repercussão da norma ISO MHEG e dos poucos trabalhos de implementação de interpretadores da representação MHEG, [Willrich, 97] definiu uma nova extensão ao modelo HTSPN orientada à produção de aplicações multimídia *Java*, chamado de *Java* I-HTSPN. Esta nova extensão é justificada pela globalização dos recursos necessários à produção e apresentação de aplicações multimídia *Java*, possibilitando, assim, que um maior número de pessoas possam utilizar as aplicações desenvolvidas.

Neste capítulo serão apresentados os modelos HTSPN, I-HTSPN e *Java* I-HTSPN.

4.1 Modelo HTSPN

O Modelo HTSPN [Sénac, 95] repousa sobre os conceitos estabelecidos pelo modelo de referência Dexter [Halaz, 94] e sobre o modelo TSPN [Diaz, 93] para a especificação da estrutura conceitual de documentos hipermídia.

4.1.1 Elementos básicos do modelo HTSPN

O modelo HTSPN permite a modelagem formal, com extensões temporais, dos três principais conceitos do modelo Dexter: componentes atômicos, ligações e composto.

Componentes Atômicos

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 77, o lugar *Video*1 modela uma apresentação de um componente atômico, um vídeo, tendo um IVT de [9,10,11].

Componente Ligação

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 à ligação introduz o conceito de ligação temporizada [Sénac, 95]. Por exemplo, o lugar L da Figura 77 especifica uma ligação.

Componente Composto

Um componente composto do modelo Dexter fornece um mecanismo de estruturação hierárquica 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 77, 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 77). 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).

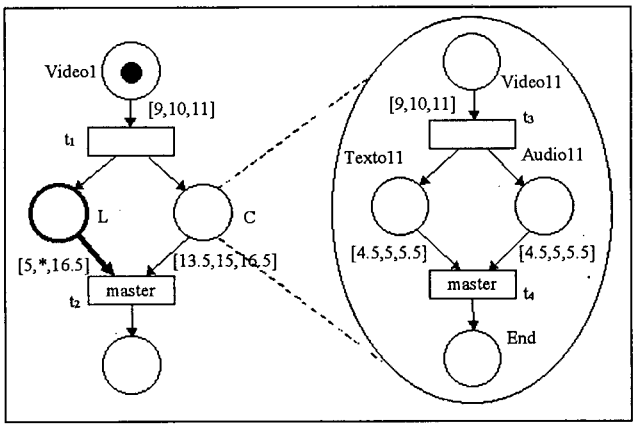


Figura 77. Modelagem dos componentes do modelo Dexter

4.1.2 Relações Lógicas e Temporais

As relações lógicas e temporais entre e no interior dos componentes são modeladas por transições tipadas. Por exemplo, a transição t2 da Figura 77 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 78 apresenta os intervalos de tiro da transição t5 para todas as estratégias de sincronização que podem ser associadas a esta transição.

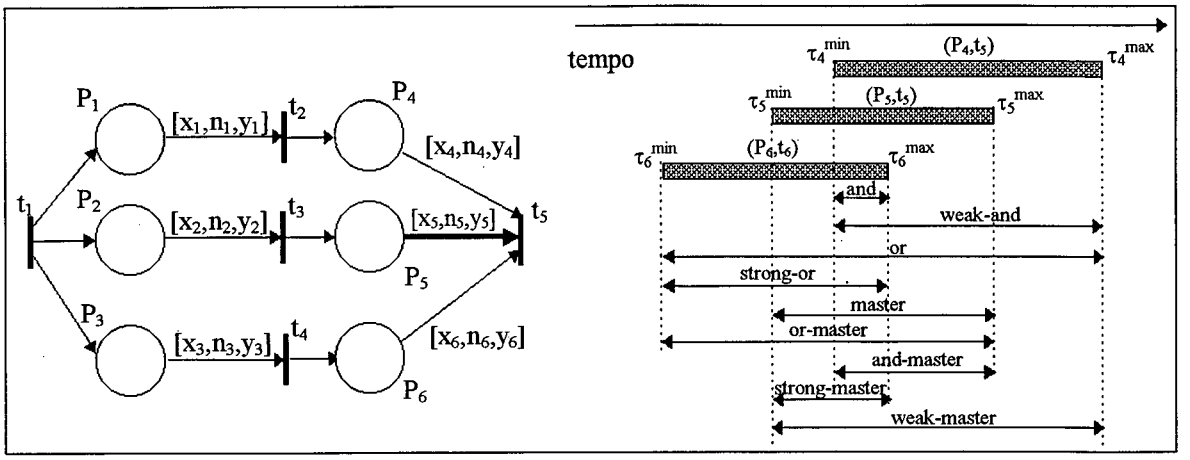


Figura 78. Semântica de sincronização do modelo HTSPN

4.1.3 Técnicas de Análise

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 a fim de validar a especificação.

4.2 Modelo I-HTSPN

Em [Willrich, 96], os autores propõem uma extensão ao modelo HTSPN a fim 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).

4.2.1 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 vídeos, 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 (V);
original-duration      10 s;
                            }
}
```

Figura 79. Especificação de um vídeo pelo objeto instância da classe Video

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 10s, armazenado em /home/dt/video11.mpg.

4.2.2 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 especificada pelo objeto instância da classe *PSVideo* abaixo:

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

Figura 80. Especificação do objeto *DataVideo* pelo objeto instância da classe *PSVideo*

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).

4.2.3 Modelagem da Estrutura Conceitual

Como apresentado na seção 3.1, a Rede HTSPN permite a especificação da estrutura conceitual de documentos hipermídia. A fim 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 FPS, que associa um lugar do tipo atômico ou ligação a um objeto definido em PS. Além disso, duas outras funções, chamadas FDS e FCS, foram adicionadas permitindo a associação de objetos *Presentation* com objetos Data e *Channel*, respectivamente.

4.3 Modelo JAVA I-HTSPN

O Modelo *Java* I-HTSPN é um modelo que foi criado para fazer uma interpretação do modelo HTSPN, gerando um código *Java* I-HTSPN, que poderá ser interpretado por uma ferramenta desenvolvida com a linguagem de programação *Java*, gerando automaticamente uma aplicação multimídia.

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

Modelo *Java* I-HTSPN especifica as três modelizações do modelo HTSPN, a modelização da estrutura do conteúdo, modelização da estrutura de apresentação e a modelização da estrutura conceitual.

4.3.1 Especificação da Estrutura do Conteúdo

No modelo HTSPN, a classe Data é utilizada para especificar um tipo de informação: imagens, áudios, textos e animações. O Modelo *Java* I-HTSPN utiliza as classes *StillPicture*, *Audio*, *Text* e *Animation* da linguagem *Java* para implementar as especificações desses elementos. Um atributo comum em todas essas classes *Java* é o URL (*Uniform Resources Locator*), sendo essa a informação necessária para o acesso aos dados. Por exemplo, o objeto *StillPicture* abaixo especifica uma imagem.

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

Figura 81. Especificação de uma imagem através do objeto *StillPicture*

4.3.2 Especificação da Estrutura de Apresentação

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;
- *PSAudio*, permite a especificação das características de uma apresentação de um áudio;
- *PSText*, permite a especificação das características de uma apresentação de um texto;
- *PSAnimation*, permite a especificação das características de apresentação de uma animação;
- *PSLink*, permite a especificação das características de apresentação de um botão da aplicação multimídia Java.

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 ]
}
```

Figura 82. Especificação da imagem MinhaImagem pelo objeto *PSStillPicture*

4.3.3 Especificação da Estrutura Conceitual

No modelo *Java* I-HTSPN, além da rede HTSPN, a estrutura conceitual do documento é especificada 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.

4.4 Conclusão

A linguagem SMIL permite realizar a autoria de apresentações multimídia na Web utilizando-se somente um editor de texto. A desvantagem é que a manipulação da noção de tempo na forma textual é difícil de manipular e modificar. Ao contrário, o modelo I-HTSPN facilita a autoria de apresentações multimídia por ser um sistema gráfico e, além disso, por ser um método formal que permite realizar uma análise do documento, a fim de verificar inconsistências temporais.

O objetivo da dissertação de mestrado é propor uma interpretação do I-HTSPN para gerar código SMIL, que será chamado SMIL I-HTSPN.

No próximo capítulo será apresentado um comparativo do modelo I-HTSPN com a linguagem SMIL Boston. Serão apresentados os componentes básicos do modelo I-HTSPN e identificados os seus correspondentes na linguagem SMIL Boston. Será apresentado também um estudo da viabilidade de tradução.

Capítulo 5. Comparativo I-HTSPN x SMIL

O principal objetivo desta dissertação é propor uma nova interpretação do modelo HTSPN para especificação de documentos SMIL Boston. Esta nova interpretação é chamada de SMIL I-HTSPN. Como etapa inicial para a definição do SMIL I-HTSPN foi realizado um estudo comparativo entre o modelo I-HTSPN e a linguagem SMIL Boston. Este comparativo é apresentado neste capítulo. Por simplificação, a linguagem SMIL Boston será referenciada simplesmente por SMIL.

Não será considerada a linguagem SMIL como um todo. Serão considerados os aspectos que possuem correspondentes no modelo I-HTSPN.

5.1 Equivalências do Modelo I-HTSPN e da Linguagem SMIL

Esta seção apresenta as correspondências dos elementos básicos do modelo I-HTSPN e da linguagem SMIL.

5.1.1 Componentes atômicos

Os componentes atômicos representam os dados codificados (texto, áudio, vídeo, etc.). No HTSPN, são modelados por um lugar do tipo atômico e arcos de saída associados.

O lugar do tipo atômico é representado graficamente por um círculo. A marcação de um lugar atômico representa o início do tratamento do componente atômico modelado, por exemplo, a chegada de uma ficha no lugar *Video1* representa o início do tratamento de um vídeo.

Os arcos definem os intervalos de validade temporal (IVT) do componente atômico. Esse IVT é uma tríplice [x, n, y], onde *x*, *n* e *y* especificam, respectivamente, a duração mínima, nominal e máxima de apresentação de um componente. Assim, o componente atômico terá uma duração mínima de x, e se caso o componente encerrar

sua apresentação antes deste tempo, esta terá que ser completada, por exemplo, congelando a última imagem do vídeo, repetindo a apresentação do vídeo até a duração mínima ou apresentando uma imagem. Além disso, caso o componente não encerrar até y, ele vai ser interrompido.

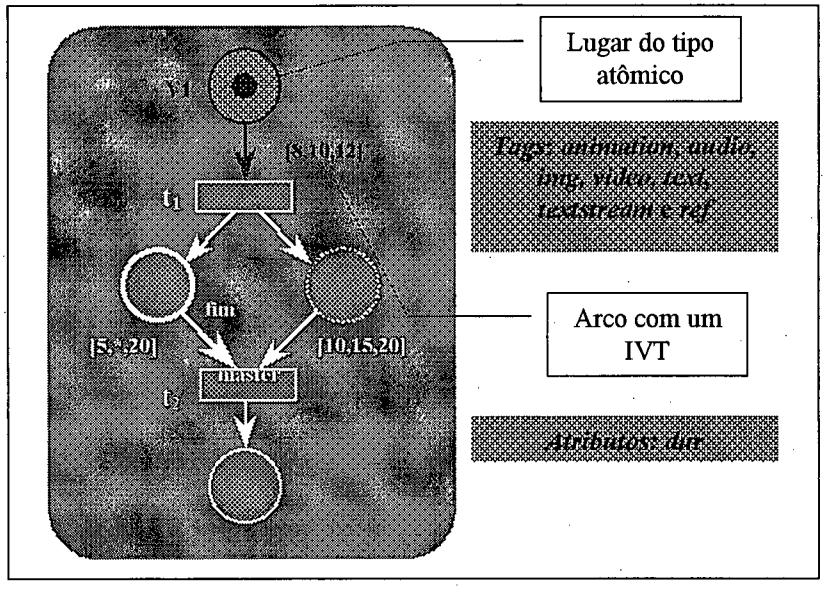


Figura 83. Ilustração de um componente atômico do modelo HTSPN modelado por um lugar do tipo atômico e arcos de saída associados, que são acompanhados de um quadro explicativo dos seus correspondentes na linguagem SMIL

Em SMIL, um lugar do tipo atômico do modelo I-HTSPN corresponde a um dos seguintes *tags*: *animation*, *audio*, *img*, *video*, *text*, *textstream* e *ref*, que permite a definição dos componentes numa apresentação SMIL.

A linguagem SMIL não tem mecanismos explícitos para especificar a validade temporal dos componentes de uma aplicação. Ela permite especificar apenas uma duração ideal através do atributo *dur* que fixa uma duração exata para a apresentação da mídia. Esse tipo de comportamento não é o ideal quando tratamos com mídias distribuídas em apresentações em tempo real, pois como mostra a Figura 85, podem ocorrer atrasos no envio da mídia devido ao congestionamento da rede.

Considerando um vídeo com duração nominal de 20s. Se este vídeo estiver armazenado localmente, o tempo de apresentação não sofrerá atrasos (Figura 84). Caso este mesmo vídeo seja recuperado em tempo real de um servidor, ele estará sujeito a variações temporais (Figura 85).

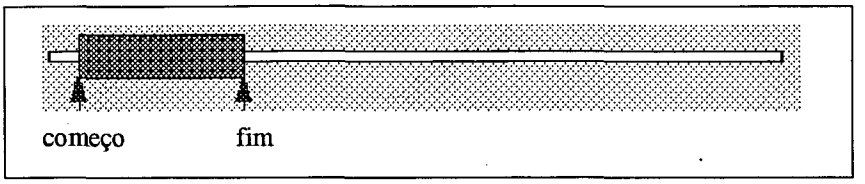


Figura 84. Tempo do documento

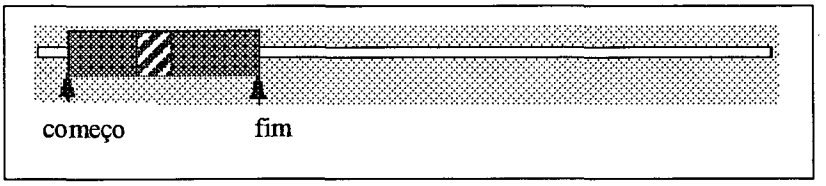


Figura 85. Tempo real da apresentação

5.1.2 Componentes ligações

Os componentes ligações definem a estrutura lógica, isto é, os relacionamentos lógicos entre os componentes do documento (definindo assim a semântica de percurso do documento). Esses componentes são modelados por um lugar do tipo ligação e arcos de saída associados.

O componente do tipo ligação define o mecanismo de interatividade do modelo I-HTSPN. SMIL apresenta três mecanismos de interatividade:

- A inclusão do evento *click* no atributo de um elemento (Figura 28, Figura 29, Figura 30 e Figura 31). Este evento pode, por exemplo, interromper a apresentação da mídia clicada, ou pode ser utilizado como condição de uma outra ação. Este mecanismo de interação pode ser utilizado para implementar lugares do tipo ligação em cenários sem exclusão mútua, do tipo apresentado na Figura 86.
- A utilização dos *tags* "a" (Figura 55, Figura 57 e Figura 58) e "area" (Figura 59, Figura 61, Figura 63 e Figura 64). Esses *tags* são ativados por interação do usuário e abrem um outro documento, incluindo a URI do destino da ligação em um dos seus atributos. Não é possível o deslocamento para um outro lugar do mesmo documento. Este mecanismo de interação (que executa outro documento) não é considerado no modelo I-HTSPN.
- O tag "excl" (Figura 52, Figura 53 e Figura 54). Esse *tag* permite apresentar conteúdo opcional, que pode ser mostrado em resposta ao usuário ou a outro evento,

sem carregar uma outra apresentação SMIL. Este mecanismo de interação pode ser utilizado para implementar lugares do tipo ligação em cenários com exclusão mútua (que será visto mais adiante).

Correspondente SMIL de um componente do tipo ligação sem exclusão mútua

A Figura 86 apresenta um exemplo de uma aplicação contendo um lugar do tipo ligação sem ocorrer a exclusão mútua. É apresentado ao usuário, em paralelo, um botão "fim", que possui o IVT de [5,*,20] e um cenário multimídia, decorrendo disso os seguintes comportamentos:

- se o usuário selecionar o botão de fim antes da quinta unidade de tempo após o início da sua apresentação, o cenário multimídia só será finalizado quando completar 5 unidades de tempo;
- se o usuário selecionar esse botão no intervalo de 5 a 20 unidades de tempo, a apresentação do cenário multimídia "C" será cancelada e será finalizada a sua execução;
- se o usuário não selecionar o botão passadas 20 unidades de tempo, será finalizada a apresentação do cenário multimídia "C" automaticamente.

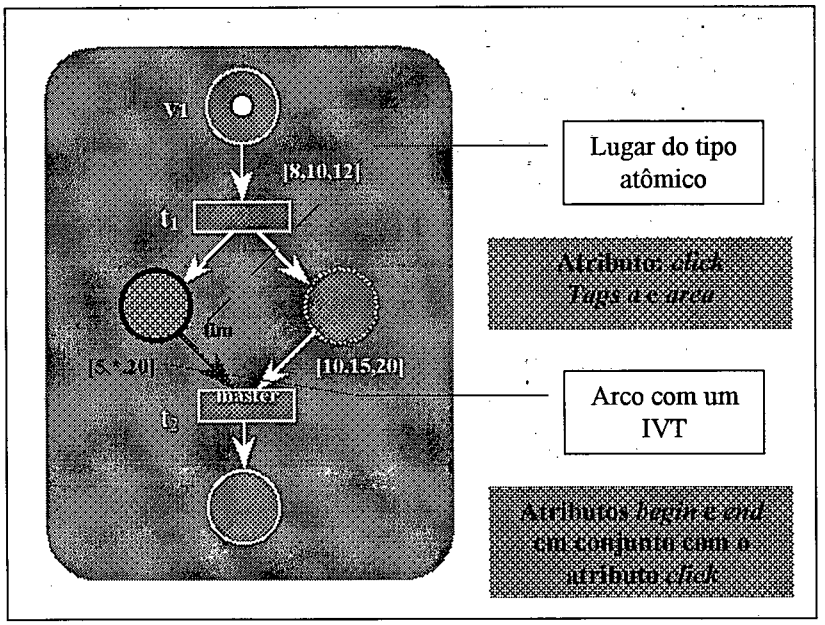


Figura 86. Ilustração de um componente ligação do modelo HTSPN modelado por um lugar do tipo ligação e arcos de saída associados, que são acompanhados de um quadro explicativo dos seus correspondentes na linguagem SMIL

Para traduzir o exemplo da Figura 86 para a linguagem SMIL, é preciso adicionar um intervalo de validade temporal no atributo *click*. Para isso, deverão ser incluídos os atributos *begin*, *dur* e *end* no elemento que será utilizado para cancelar a execução da composição paralela, ou seja, no *tag img*. O exemplo da Figura 86 pode ser representado da seguinte forma na linguagem SMIL.

```
...
1  <seq>
2      <vídeo ...
3      <par>
4          <img id = fim begin=5s end=click dur=20 >
5          <par id = composto end = fim.click dur=20 >
6          ....
7          </par>
8      </par>
9  </seq>
...
```

Figura 87. Estruturação informal de um arquivo SMIL ilustrando o exemplo da Figura 86

No *tag img* os atributos *id*, *begin*, *end* e *dur* representam respectivamente o identificador do objeto, o tempo de espera para o início da apresentação da imagem (5 segundos), a definição de um evento (quando o usuário clicar na figura essa terá a sua execução finalizada) e o tempo máximo de apresentação da figura.

No *tag par* os atributos *id* e *dur* possuem a mesma funcionalidade do apresentado no *tag img*. Já o *tag end* define um evento que quando o usuário clicar na imagem (id = fim) a composição paralela terá sua execução finalizada.

A tradução do exemplo da Figura 86 para a linguagem SMIL apresenta uma diferença causada pela limitação da linguagem SMIL. No documento especificado com HTSPN, o botão "fim" poderá ser selecionado antes de 5u.t., porém, se isso acontecer ele só será executado quando completar 5u.t., então ele é apresentado desde o início da apresentação. Já na tradução SMIL, o botão "fim" será exibido ao usuário somente depois de decorridos 5u.t. Esta solução foi adotada pois não há maneiras de habilitar/desabilitar botões em SMIL.

Correspondente de um componente ligação com exclusão mútua

Para representar em SMIL os componentes ligação com exclusão mútua é necessário utilizar o elemento *excl*. Este elemento permite apresentar conteúdo opcional em resposta ao usuário ou a outro evento, sem carregar uma outra apresentação SMIL.

A Figura 88 apresenta o exemplo de uma exclusão mútua no modelo I-HTSPN. Na figura seguinte, mostra a tradução desse mesmo exemplo utilizando-se o código SMIL Boston.

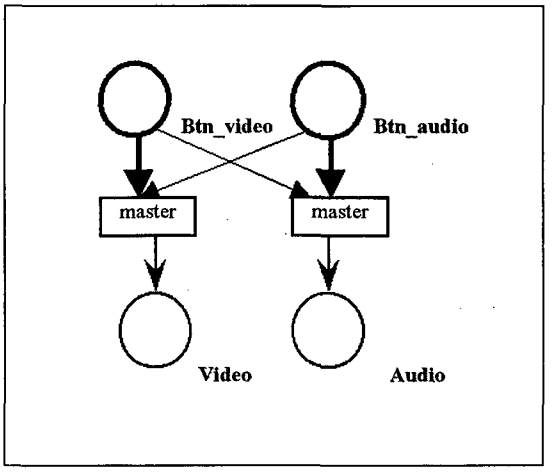


Figura 88. Exclusão mútua no modelo I-HTSPN

```
<par>
    <excl>
        <par id="btn_video">
            <video ... />
        </par>
        <par id="btn_audio">
            <audio ... />
        </par>
    </excl>
        <a href="btn_video"><img src="botao1.jpg"/> end=botao2.end; end=click</a>
        <a href="btn_audio"><img src="botao2.jpg"/> end= botao1.end; end=click</a>
</par>
```

Figura 89. Tradução para a linguagem SMIL da Figura 88

Componentes ligação definindo *loops*

A Figura 90 mostra dois tipos de *loops* em uma Rede HTSPN, um que leva a um outro ponto interno do documento (Figura 90a) e outro que leva ao reinício da apresentação (Figura 90b).

Na linguagem SMIL é possível descrever somente *loops* de reinicialização (Figura 90b), utilizando-se o atributo *a*. Essa linguagem não permite descrever *loops* que levam a um outro ponto interno do documento (Figura 90a).

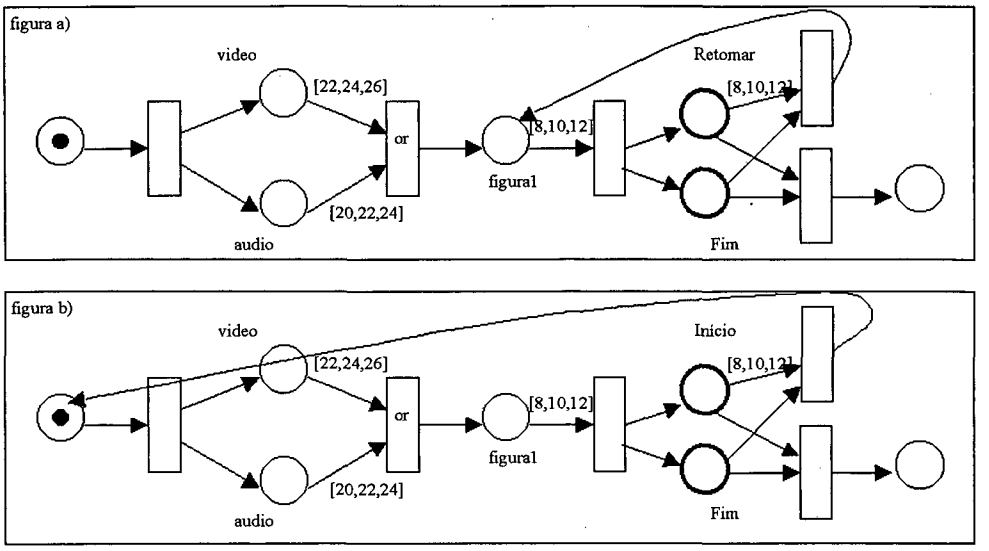


Figura 90. Componentes ligação definindo *loops* em uma Rede HTSPN

5.1.3 Componentes compostos

Os componentes compostos fornecem um mecanismo de estruturação hierárquica baseado na composição recursiva dos componentes atômicos, ligações e compostos. São modelados por um lugar abstrato, chamado lugar composto, que representa uma sub-rede.

O lugar do tipo composto é representado graficamente por um círculo pontilhado. 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 também é marcado (por exemplo, o lugar *Video11 da Figura 91*). A saída da ficha de um lugar composto implica na saída de todas 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 (por exemplo, quando o lugar *End* é marcado).

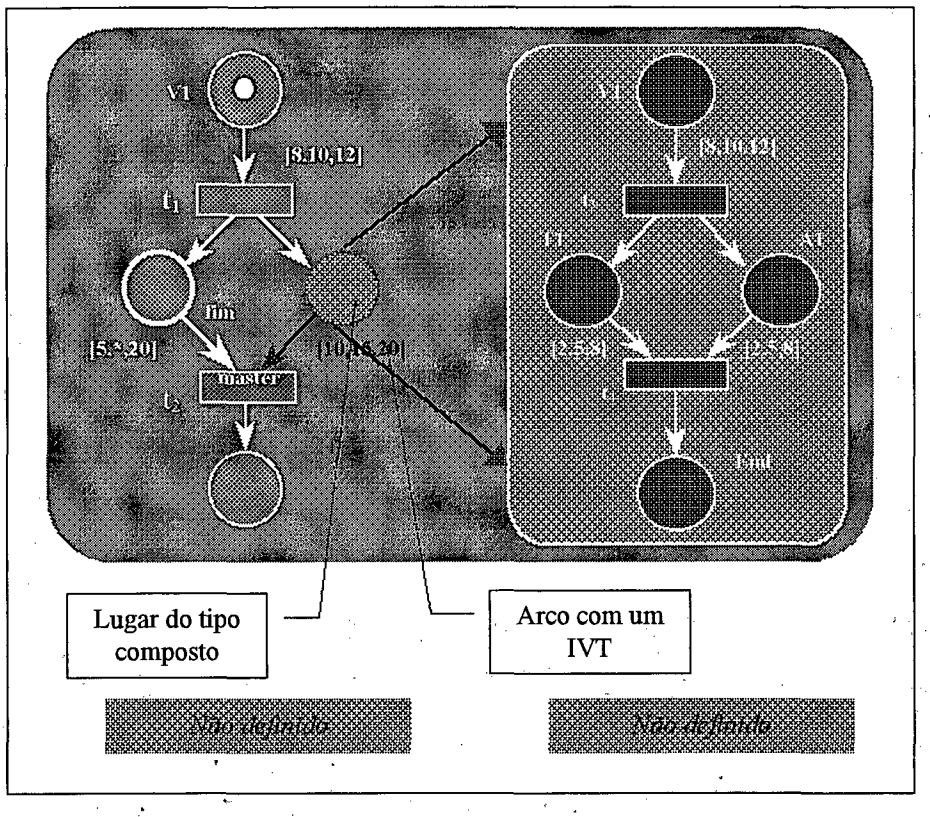


Figura 91. Ilustração de um componente composto do modelo HTSPN modelado por um lugar do tipo atômico e arcos de saída associados, que são acompanhados de um quadro explicativo dos seus correspondentes na linguagem SMIL

Um modelo gráfico de autonível para SMIL poderia fornecer o conceito de estruturação na qual o componente composto seria substituído pelo seu conteúdo. Ou seja, o componente composto seria uma forma de "alias", que no momento da tradução seria substituído pelo seu conteúdo. Caso o componente composto seja reutilizado, ocorrerá a replicação de código. Isto, pois a reutilização do código não é possível em SMIL.

5.1.4 Relações Temporais HTSPN

No modelo HTSPN as relações temporais entre os componentes do documento são definidas pelos arcos e transições tipadas. O autor associa às transições uma estratégia de sincronização entre: *and*, *weak-and*, *or*, *strong-or*, *master*, *or-master*, *and-master*, *strong-master* e *weak-master*. Essas estratégias de sincronização são utilizadas a fim de descrever os esquemas de sincronização, tomando em consideração o não-

determinismo introduzido pelo tratamento dos componentes do documento, já que eles estão distribuídos e dependem dos recursos da rede.

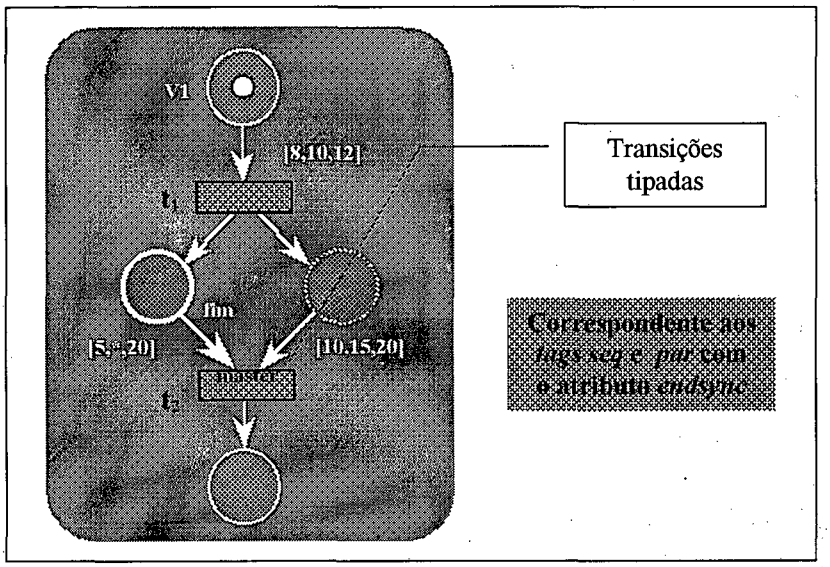


Figura 92. Ilustração da inclusão de relações temporais entre os componentes de um documento HTSPN

As transições tipadas são correspondentes na linguagem SMIL aos *tags seq* e *par*, incluindo o atributo *endsync* do *tag par* com os valores *first*, *id-ref* ou *last*. O *first* possui a mesma funcionalidade do *or* do I-HTSPN, o *id-ref* ao *master* e o *last* ao *and*, respectivamente. As outras estratégias de sincronização do modelo I-HTSPN, *weak-and*, *strong-or*, *or-master*, *and-master*, *strong-master* e *weak-master*, não possuem equivalentes na linguagem SMIL, pois ela não leva em consideração o não-determinismo temporal dos sistemas distribuídos. A Figura 93 exemplifica a comparação das transições tipadas do I-HTSPN com a linguagem SMIL.

I-HTSPN	SMIL
C_1 $[x_1,n_1,y_1]$ master C_2 $[x_2, n_2,y_2]$	<par **endsync="id(c1)"**> **<video id="c1"** src="aniver.mpv" region="Videos" *dur="n_1"* <audio id="c2" src="aniver.aiff" dur="n_2" /> </par>

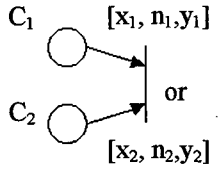
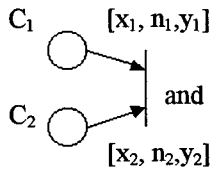
I-HTSPN	SMIL
	<par **endsync="first"**> <video id="c1" src="aniver.mpv" region="Videos" *dur="n_1"* <audio id="c2" src="aniver.aiff" dur="n_2" /> </par>
	<par **endsync="last"**> <video id="c1" src="aniver.mpv" region="Videos" *dur="n_1"* <audio id="c2" src="aniver.aiff" dur="n_2" /> </par>

Figura 93. Quadro comparativo das transições tipadas

Na Figura 93, os valores *id-ref*, *first* e *last* utilizados no atributo *endsync do tag par* definem que:

- o *id-ref* determina que a composição paralela será finalizada quando terminar a apresentação do elemento-mestre identificado, no exemplo acima, o id="c1";
- o *first* determina que a composição paralela será finalizada assim que o primeiro elemento terminar a sua apresentação;
- o *last* determina que a composição paralela será finalizada assim que o último elemento terminar a sua apresentação.

A linguagem SMIL não permite especificar um intervalo de validade temporal para as apresentações, mas apenas permite definir um tempo ideal. Assim, todo intervalo de validade temporal deve ser definido da seguinte forma: [d,d,d], onde d é a duração nominal do componente. Portanto, todo intervalo temporal definido na Figura 93 terá a duração mínima e máxima igual a duração nominal do componente (ou seja, $x_i=y_i=n_i$).

5.1.5 Estrutura de Conteúdo

Esta estrutura é responsável por descrever as informações que constituem os componentes.

No modelo HTSPN os dados multimídia são modelados por um conjunto de objetos *Data*. Para cada tipo de mídia existe uma classe do tipo *Data* que permite a descrição de um tipo de informação. São elas: *Video*, *Audio*, *Text* e *StillPicture* que permitem a especificação de vídeos, áudios, textos e imagens, respectivamente. Os objetos *Data* possuem atributos como *code*, *original-size*, *original-duraction* e *content* que definem o tipo de codificação da mídia, a largura e altura da mídia, o tempo da duração original e o local onde esta mídia está armazenada, respectivamente.

A estrutura de conteúdo na linguagem SMIL é especificada utilizando o atributo *src* no momento da definição da mídia, ou seja, quando se realiza a definição dos componentes atômicos do documento. Os atributos *original-size* e *original-duraction* da linguagem I-HTSPN não possuem correspondentes na linguagem SMIL. *Code* é representado diretamente pela extensão da URI.

I-HTSPN	SMIL
Media MinhaMedia ::= { **Content "x";** **}**	Adicionar o atributo abaixo na definição da mídia: **src="x"**
onde: • Media pode ser: Audio, Text, Video e StillPicture; • x é a URI.	

Figura 94. Quadro comparativo da definição da estrutura de conteúdo no modelo I-HTSPN e a forma correspondente de se definir na linguagem SMIL

5.1.6 Estrutura de Apresentação

A estrutura de apresentação descreve como e onde os diferentes componentes serão apresentados, além das características espacias da janela principal.

No modelo HTSPN as características da janela principal da apresentação são modeladas pelo objeto *Description*, incluindo os atributos *dimension* e *background* que definem respectivamente a dimensão da janela principal e a cor de fundo da janela.

As características de apresentação de cada componente atômico são modeladas por um conjunto de objetos *Presentation*. Para cada tipo de mídia existe uma classe do tipo *Presentation* que permite a descrição das características de apresentação. São elas: *PSVideo*, *PSAudio*, *PSText* e *PSStillPicture* que permitem a especificação das características de apresentação de vídeos, áudios, textos e imagens, respectivamente. Os objetos *Presentation* possuem atributos como *data*, *background* e *presentation-size* que definem o objeto *Data*, a cor de fundo e a altura e largura do componente atômico, respectivamente.

Na linguagem SMIL o objeto *Description* corresponde ao *tag root-layout*. Os atributos *dimension* e *background* do objeto *Description* correspondem aos atributos *width*, *height* e *background-color* do elemento *root-layout*.

Os atributos *data*, *background* e *presentation-position dos* objetos *Presentation Specification* da linguagem I-HTSPN são correspondentes na linguagem SMIL ao *tag layout* incluído no cabeçalho do documento SMIL e ao atributo *region* incluído na definição do componente atômico. Note que a forma da definição das características de apresentação das mídias é diferente na linguagem SMIL e no modelo I-HTSPN. Na linguagem SMIL cria-se regiões onde as mídias serão apresentadas (podendo serem reaproveitadas). No I-HTSPN cria-se um objeto *Presentation* para cada mídia, não importando se apresenta as mesmas características de outro objeto já criado anteriormente. A vantagem da utilização de regiões no SMIL, em relação à definição de objetos, como ocorre na linguagem I-HTSPN, é que uma mesma região pode ser reaproveitada.

O quadro abaixo resume as correspondências entre I-HTSPN e SMIL a nível da representação da estrutura de apresentação.

I-HTSPN	**SMIL**

Description {
Dimension [y z]
Background k
};
	<root-layout width="y" height="z" background-color="k" />

<table>
<tr><th>I-HTSPN</th><th>SMIL</th></tr>
<tr><td>Description {
Dimension [y z]
Background k
};</td><td><root-layout width="y" height="z" background-color="k" /></td></tr>
<tr><td>PSMedia PresMinhaMedia ::= {
data MinhaMedia
presentation-size y (H) x z (W):
background k
}</td><td>No cabeçalho do documento, define-se a região:

<layout >
<region id="w"
height="y" width="z"
background-color="k" />
</layout>

Adicionar o atributo abaixo na definição da mídia:

region="w"</td></tr>
<tr><td colspan="2">onde:
• Media pode ser: audio, text, video, image e animation;
• y e z é a distância em pixels;
• k é a cor.</td></tr>
</table>

Figura 95. Quadro comparativo da definição da estrutura de apresentação no modelo I-HTSPN e a forma correspondente de se definir na linguagem SMIL

5.1.7 Técnicas de Análise

O modelo HTSPN permite a verificação da consistência temporal dos cenários multimídia, a simulação das atividades dinâmicas do documento hipermídia modelado e permite utilizar todas as técnicas de análise de uma Rede de Petri.

A linguagem SMIL não permite usar nenhum tipo de técnica de análise.

5.2 Exemplo I-HTSPN →SMIL

Esta seção apresenta um exemplo de uma apresentação multimídia em I-HTSPN e sua correspondente representação usando a linguagem SMIL. Esta seção tem como

objetivo ilustrar as correspondências apresentadas na seção anterior.

Optou-se por escolher um exemplo que não tivesse componentes compostos, pois esse tipo de componente não tem correspondência na linguagem SMIL.

O exemplo utilizado para fazer a comparação entre as duas linguagens apresenta um áudio e um vídeo em paralelo. Foi incluída a estratégia de sincronização *or* que especifica que, quando a apresentação do primeiro elemento terminar, finaliza a apresentação do outro também. Em seguida é apresentado uma seqüência de slides: figura1 e figura 2.

5.2.1 I-HTSPN

O I-HTSPN é modelado seguindo uma descrição multinível, incluindo as estruturas: conceitual, de apresentação e de conteúdo.

Estrutura Conceitual

Essa estrutura realiza a organização semântica do documento, seus componentes e as relações lógicas e temporais entre estes componentes. Esta especificação é ilustrada abaixo, na Figura 96.

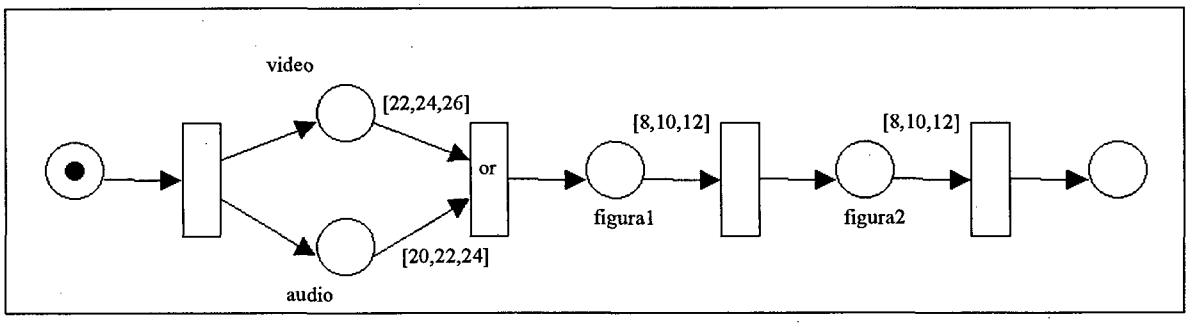


Figura 96. Definição da estrutura conceitual de um documento I-HTSPN

Foi adicionado às mídias um IVT, a fim de especificar uma tolerância de sincronização em sistemas multimídia distribuídos:

- no vídeo foi especificada uma duração mínima de 22u.t, nominal de 24u.t e máxima de 26u.t;
- no áudio foi especificada uma duração mínima de 20u.t, nominal de 22u.t e máxima de 24u.t;

- nas figuras foram especificadas uma duração mínima de 8u.t, nominal de 10u.t e máxima de 12u.t.

Se não ocorrerem atrasos na rede, o áudio terá uma duração de 22 unidades de tempo e, como foi inserida a estratégia de sincronização *or*, será cancelada a apresentação do vídeo e será apresentada a seqüência de imagens.

Estrutura de Conteúdo

Esta estrutura é responsável por descrever as informações que constituem os componentes.

Os objetos definidos abaixo especificam a estrutura de conteúdo da aplicação, ou seja, especificam os URIs de todas as mídias utilizadas.

```
Video DadosDoVideo ::= {
        content          "http://www.inf.ufsc.br/~adriana/video.rm";
    }


Audio DadosDoAudio ::= {
        content          "http://www.inf.ufsc.br/~adriana/audio1.rm";
    }

Picture DadosDaFigura1 ::= {
        content          "http://www.inf.ufsc.br/~adriana/figura1.jpg";
    }

Picture DadosDaFigura2 ::= {
        content          "http://www.inf.ufsc.br/~adriana/figura2.jpg";
    }
```

Estrutura de Apresentação

Os objetos abaixo definem a estrutura de apresentação da aplicação, ou seja, especifica o tamanho da mídia (H – altura e W – largura), atributo *presentation-size* e identifica a classe que dá as características do conteúdo do mesmo objeto (atributo *data*).

```
    PSVideo ApresentVideo ::= {
        data                    DadosDoVideo;
        presentation-size       144 (H) x 176 (W);
}
```

```
    PSAudio ApresentAudio ::= {
        data                    DadosDoAudio;
}

    PSPicture ApresentFigura1 ::= {
        data                    DadosDaFigura1;
        presentation-size       144 (H) x 176 (W);
}

    PSPicture ApresentFigura2 ::= {
        data                    DadosDaFigura2;
        presentation-size       144 (H) x 220 (W);
}
```

Além disso, esta estrutura utiliza o objeto *Description* que descreve as características de apresentação da janela principal do modelo I-HTSPN.

```
Description {
        dimension       [400 200]
        background      black
};
```

5.2.2 SMIL

Nessa seção é apresentado o código SMIL do exemplo I-HTSPN da Figura 96.

```
1<smil>
2   <head>
3       <layout >
4           <root-layout width="400" height="200" background-color="black" />
5           <region id="esquerda" title="região localizada a esquerda da janela principal"
6                   left="0" top="0" height="144" width="176" background-color="#000000" fit="hidden"/>
7           <region id="direita" title=" região localizada a direita da janela principal "
8                   left="180" top="0" height="144" width="220" background-color="#000000" fit="hidden"/>
9       </layout>
10  </head>
11  <body>
12      <seq>
13          <par title="Composicao 1 do seq" endsync="first">
14              <video src=" http://www.inf.ufsc.br/~adriana/video.rm" region="esquerda" />
15              <audio src=" http://www.inf.ufsc.br/~adriana/audio.rm" />
16          </par>
17          <seq title="Composicao 2 do seq">
18              <img src=" http://www.inf.ufsc.br/~adriana/figura1.jpg" region="esquerda" dur="10s"/>
19              <img src=" http://www.inf.ufsc.br/~adriana/figura2.jpg" region="direita" dur="10s"/>
20          </seq>
21      </seq>
22  </body>
23</smil>
```

Figura 97. Código SMIL equivalente ao exemplo do I-HTSPN

Nas linhas 14, 15, 18 e 19 foram definidas as mídias que farão parte do documento SMIL, respectivamente um vídeo, um áudio, uma imagem e outra imagem. Diferente do I-HTSPN, no mesmo momento em que definimos as mídias, determinamos a região onde elas serão apresentadas. No exemplo da Figura 97 definimos apenas duas

regiões e todas as mídias fizeram referência a uma delas. Já na linguagem I-HTSPN, não se leva em consideração se um objeto utiliza um mesmo tamanho de apresentação de outro: são criados objetos diferentes para se definir uma mesma região.

As relações lógicas no I-HTSPN foram feitas através de arcos e transições, como mostra a Figura 96, já na linguagem SMIL utilizou-se os *tags par* e *seq* (linhas 12, 13, 16, 17, 20 e 21). Percebe-se uma grande dificuldade de visualização na estrutura como um todo de um documento SMIL. Fica visível a dificuldade de manipulação de um arquivo multimídia na forma textual.

5.3 Conclusão

Este capítulo apresentou os elementos básicos do modelo I-HTSPN e seus correspondentes na linguagem SMIL Boston. Foi apresentado também um exemplo da tradução de uma apresentação multimídia em I-HTSPN para a linguagem SMIL.

Este estudo comparativo serviu como base para a definição de uma nova interpretação do HTSPN voltada à especificação de documentos SMIL. Através desse estudo foi possível identificar que alguns elementos do HTSPN não podem ser representados em SMIL e vice-versa. Com isso, foi necessário fazer limitações no modelo I-HTSPN para possibilitar a tradução para SMIL. Esta nova interpretação é apresentada no próximo capítulo.

As principais equivalências entre o modelo formal I-HTSPN e a linguagem SMIL são:

- os lugares do tipo atômico do modelo I-HTSPN correspondem a um dos seguintes *tags* da linguagem SMIL: *animation*, *audio*, *img*, *video*, *text*, *textstream* e *ref*. O IVT do modelo I-HTSPN não possui correspondente na linguagem SMIL, pois essa linguagem só permite especificar a duração ideal de apresentação da mídia;
- os componentes ligações do modelo I-HTSPN correspondem ao atributo *click* e aos *tags a* e *area* da linguagem SMIL;
- os componentes compostos do modelo I-HTSPN não possuem correspondentes na linguagem SMIL;

- as transições tipadas do modelo I-HTSPN que possuem correspondentes na linguagem SMIL são o *or*, *and* e o *master*. As outras estratégias de sincronização do modelo I-HTSPN, *weak-and, strong-or, or-master, and-master, strong-master e weak-master,* não possuem equivalentes na linguagem SMIL. As transições tipadas são correspondentes na linguagem SMIL aos *tags seq* e *par*, incluindo o atributo *endsync* do *tag par* com os valores *first*, *id-ref* ou *last*. O *first* possui a mesma funcionalidade do *or* do I-HTSPN, o *id-ref* ao *master* e o *last* ao *and*, respectivamente.

Capítulo 6. Modelo SMIL I-HTSPN

A linguagem SMIL herdou a principal característica da linguagem HTML, a simplicidade, além disso, supriu uma das limitações dessa linguagem, que é a definição de relacionamentos de sincronização temporal e espacial. Com isso, utilizando a linguagem SMIL, é possível criar documentos multimídia na Web seguindo a mesma simplicidade da linguagem HTML.

Apesar de SMIL ser dita uma linguagem simples, é difícil manipular a função do tempo na forma textual. A autoria de relacionamentos temporais na forma textual torna a autoria difícil e aumenta a possibilidade do autor definir incoerências temporais. Para detectá-las, o autor deve executar a apresentação. Os modelos gráficos têm a vantagem de ilustrar de maneira gráfica a semântica das relações temporais. Este tipo de modelo simplifica a especificação das restrições temporais e reduz o tempo de criação do documento.

Essa dissertação tem por objetivo definir um modelo gráfico de mais alto nível que permita a especificação gráfica e análise do documento, para posterior tradução para o código SMIL Boston.

A especificação gráfica de documentos SMIL será obtida graças a uma nova interpretação do modelo I-HTSPN voltada para a especificação, análise e geração de código SMIL Boston. Esta nova interpretação se chama SMIL I-HTSPN e herdará todas as técnicas de análise do HTSPN.

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

O modelo SMIL I-HTSPN não se trata de uma reformulação do modelo I-HTSPN apresentado na seção 4.2. Nesta nova extensão, somente os atributos dos objetos *Data* e *Presentation* foram modificados a fim de que as informações de acesso aos dados e as características de apresentação definidas por estes objetos tenham

elementos correspondentes na linguagem SMIL. Além disso, restrições ao poder de expressão do modelo HTSPN foram definidas. Isto foi necessário devido ao baixo poder de expressão da linguagem SMIL, identificada no capítulo anterior.

Esse capítulo também apresenta o procedimento de tradução de uma especificação SMIL I-HTSPN para o código SMIL.

6.1 Especificação da Estrutura do Conteúdo

Como apresentado na seção 4.2.1, os dados multimídia utilizados por um documento multimídia são especificados por uma lista de objetos das classes *Data*. Cada classe *Data* permite a especificação de um tipo de mídia. No modelo SMIL I-HTSPN são considerados seis tipos de informações: imagens, áudios, textos, textos em movimento, vídeos e animações, que são as informações tratadas explicitamente pela linguagem SMIL Boston. No modelo SMIL I-HTSPN estes dados podem ser descritos por objetos instâncias das classes *Data StillPicture*, *Audio*, *Text*, *Textstream*, *Video* e *Animation*.

As classes *Data* têm como atributo 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 mídia descrito (identificada pela extensão do arquivo). Além do URL, todas as classes *Data* têm como atributo o nome do autor do conteúdo (*author*) do elemento e informações sobre os direitos autorais (*copyright*).

Por exemplo, o objeto *Video* abaixo especifica um vídeo.

```
Video MeuVideo ::= {
        URI             http://www.inf.ufsc.br/~adriana/meuvideo.avi;
        author          "Adriana Dallacosta";
        copyright       "livre";
}
```

6.2 Especificação da Estrutura de Apresentação

Esta estrutura define as características de apresentação dos componentes do documento e os canais lógicos, ou regiões, na qual os componentes serão posicionados e apresentados.

6.2.1 Especificação das características de apresentação dos componentes

Como introduzido na seção 4.2.2, as apresentações que compõem uma aplicação multimídia são especificadas por um conjunto de objetos instâncias de uma das classes *Presentation*. No modelo SMIL I-HTSPN, são definidas as seguintes classes *Presentation*:

- *PSStillPicture*, permite a especificação das características de apresentação de uma imagem. Um objeto desta classe tem os seguintes atributos: o objeto *StillPicture* que descreve os dados a serem apresentados (*data*); o objeto *Channel* que descreve a região em que esse objeto será posicionado (*channel*); um título para a mídia (*title*) e uma apresentação alternativa (*alternative*). Esta apresentação alternativa pode repor a apresentação principal caso uma certa condição não seja atendida. Esta condição pode ser condicionada à velocidade da ligação, do tipo de idioma, propriedades de configuração do vídeo (dimensões da janela de apresentação ou número de cores disponível) ou se houver problemas de acesso ao recuperar a mídia.
- *PSAudio*, permite a especificação das características de apresentação de um áudio. Um objeto *PSAudio* especifica as seguintes informações: o objeto *Audio* que descreve os dados a serem apresentados (*data*); um título para a mídia (*title*) e uma apresentação alternativa (*alternative*).
- *PSText*, permite a especificação das características de apresentação de um texto. Um objeto instância desta classe especifica: o objeto *Text* que descreve os dados a serem apresentados (*data*); o objeto *Channel* que descreve a região em que esse objeto será posicionado (*channel*); um título para a mídia (*title*) e uma apresentação alternativa (*alternative*).
- *PSTextStream*, permite a especificação das características de apresentação de um texto em movimento. Um objeto instância desta classe especifica: o objeto *Textstream* que descreve os dados a serem apresentados (*data*); o objeto *Channel* que descreve a região em que esse objeto será posicionado (*channel*); um título para a mídia (*title*) e uma apresentação alternativa (*alternative*).
- *PSVideo*, permite a especificação das características de apresentação de um vídeo. Um objeto instância desta classe especifica: o objeto *Video* que descreve os dados a serem apresentados (*data*); o objeto *Channel* que descreve a região em que esse

objeto será posicionado (*channel*); um título para a mídia (*title*) e uma apresentação alternativa (*alternative*).

- *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 (*data*); o objeto *Channel* que descreve a região em que esse objeto será posicionado (*channel*); um título para a mídia (*title*) e uma apresentação alternativa (*alternative*).
- *PSHyperLink*, permite a especificação do elemento que deverá ser clicado para percorrer a ligação (*data*); o objeto *Channel* que descreve a região em que esse objeto será posicionado (*channel*); um título para a mídia (*title*); o valor do atributo *show* que controla o comportamento do objeto de origem quando o elo é percorrido; o *type* que determina o tipo de ligação (*a* ou *area*) e definir o endereço destino da ligação (*destination*).
- *PSButton*, permite a especificação das características de apresentação de um botão. Um objeto desta classe tem os seguintes atributos: o objeto *StillPicture* que descreve os dados a serem apresentados (*data*); o objeto *Channel* que descreve a região em que esse objeto será posicionado (*channel*) e um título para a mídia (*title*).

Por exemplo, o objeto *PSVideo*, descrito abaixo, especifica as características de apresentação de um vídeo.

```
PSVideo PresMeuVideo ::= {
        data                    MeuVideo;
        channel                 RegiaoA;
        title                   "Video da Minha Apresentação";
        alternative             {
                condition       minimum-bitrate 36000;
                alternativeData PresMinhaImagem;
                }
}
```

O objeto *PSHyperLink* apresentado a seguir especifica as características de uma ligação.

```
PSHyperLink MeuHiperlink ::= {
        data                    MinhaImagem;
        channel                 RegiaoA;
        title                   "Meu hiperlink definido na MinhaImagem";
        show                    replace;
        type                    a;
        destination             http://www.inf.ufsc.br/~adriana;
}
```

6.2.2 Especificações dos canais

No modelo SMIL I-HTSPN a estrutura de apresentação do documento define também os atributos das regiões de apresentação dos componentes do documento. Estas regiões são especificadas por uma lista de objetos da classe *Channel*. Um objeto desta classe permite a especificação dos atributos constituintes de uma região, que são:

- *title*, que define um título para a região;
- *position*, o *height* define a altura em porcentagem ou pixels do elemento definido e o *width* define a largura;
- *left*, que define a distância em porcentagem ou pixels do elemento definido ao lado esquerdo da janela principal;
- *top*, que define a distância em porcentagem ou pixels do elemento definido à parte superior da janela principal;
- *background-color*, que define a cor de fundo da região. Se este atributo estiver ausente o fundo será transparente;
- *fit*, que especifica o comportamento no caso da altura ou largura intrínseca de um objeto multimídia ser diferente dos valores especificados pelos atributos de altura e largura do elemento *region*. O valor *hidden* para o atributo *fit* define que se a altura ou largura intrínseca do elemento for inferior à altura ou largura definida no elemento *region*, o objeto é reproduzido a partir da extremidade superior esquerda e a altura restante é preenchida com a cor de fundo, por outro lado, se a altura ou largura intrínseca do elemento for superior à altura ou largura definida no elemento *region*, o objeto é reproduzido a partir da extremidade superior esquerda até se alcançar a altura ou largura definida no elemento *region*, sendo omitidas as partes do objeto que ficaram abaixo ou à direita. O valor *fill* dimensiona a altura e a largura do objeto de forma que toque todas as extremidades da caixa.

```
Channel definindoAsRegioes ::= {
        title                   "Região posicionada a esquerda da tela";
        position                144 (H) x 220 (W);
        left                    0;
        top                     0;
        background-color        black;
        fit                     fill;
}
```

Uma característica bastante desejável em uma linguagem de autoria de documentos multimídia é poder especificar as características de apresentação dos componentes de um documento em um objeto separado. A este nível, o modelo SMIL I-HTSPN apresenta melhorias em relação ao modelo I-HTSPN, pois nesse novo modelo tem-se o reaproveitamento das características das regiões onde serão posicionados os elementos. Caso um elemento tenha criado uma região para seu uso, outro elemento poderá utilizar a mesma região. Além disso, uma mesma região pode ser utilizada para apresentar diferentes componentes.

6.3 Especificação da Estrutura Conceitual

Como apresentado no capítulo 5, a linguagem SMIL não permite representar todos os elementos do modelo I-HTSPN. As limitações da Estrutura Conceitual do HTSPN que não se pode usar no SMIL I-HTSPN são: algumas estratégias de sincronização, o IVT e *loops* internos.

6.3.1 Especificação dos Componentes do Documento

Da mesma forma que o modelo I-HTSPN, o modelo SMIL I-HTSPN modela os componentes do documento através dos lugares atômicos, ligações e compostos. As representações gráficas adotadas são as mesmas do modelo I-HTSPN, como definido na sessão 4.1.1.

A única diferença do modelo SMIL I-HTSPN em relação ao modelo I-HTSPN, a esse nível, é que o modelo SMIL I-HTSPN não tem mecanismos explícitos para especificar a validade temporal dos componentes de uma aplicação. Ele permite especificar apenas uma duração ideal através do atributo *dur*, que fixa uma duração exata para a apresentação da mídia. Não será possível definir uma tríplice IVT [x, n, y], onde x, y e z definem valores distintos para a duração mínima, nominal e máxima admissível do tratamento associado ao componente atômico. No SMIL I-HTSPN, o valor de x, n e z são iguais. Portanto, na representação gráfica do modelo, aos arcos de saída dos lugares será associada apenas a duração nominal do componente.

6.3.2 Especificação das Relações Temporais

As estratégias de sincronização do modelo HTSPN que serão representadas no modelo SMIL I-HTSPN são o *or*, o *master* e o *and*. As outras estratégias, como o *weak-and, strong-or, or-master, and-master, strong-master e weak-master* não serão representadas, pois ele não possui equivalentes na linguagem SMIL.

Outra limitação do modelo SMIL I-HTSPN em relação ao modelo HTSPN é não permitir descrever *loops* que levam a um outro ponto interno do documento. O modelo SMIL I-HTSPN só permite descrever *loops* de reinicialização (Figura 90b).

6.3.3 Especificação da Janela Principal

Além da Rede HTSPN, a estrutura conceitual do documento é especificada por um objeto da classe *Description*. Este objeto permite a especificação da janela principal do documento, onde serão incluídas todas as regiões. Os atributos da classe *Description* são o *dimension,* que define a altura e a largura em pixels da janela principal e o *background-color* que define a cor de fundo dessa janela.

```
Description {
        dimension               [400 200];
        background-color        write;
};
```

6.4 Definição Formal do Modelo SMIL I-HTSPN

Para a definição formal do modelo SMIL I-HTSPN é necessário definir inicialmente os modelos TSPN e HTSPN, mas considerando uma duração ideal de apresentação (sem intervalo de validade temporal).

Definição 1. Uma TSPN para SMIL é uma octupla 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 uma duração nominal (n] a um arco de saída de um lugar, onde:

- $A = \{ a = (p, t) \in PxT \mid \beta(p, t) \neq 0\}$
- $IM:A \rightarrow (Q^+ \cup \infty)$, $IM(a)=n$

- SYN é uma aplicação que associa um tipo de sincronização a uma transição:
- $SYN:T \rightarrow \{and, or, master\}$
- $MA:T_m \rightarrow A$ é uma aplicação que associa uma transição de um dos tipos mestre com um arco-mestre (representado graficamente por uma flecha em negrito).
- $T_m = \{t \in T \mid SYN(t) \in \{master\}\}$

Definição 2. ***Uma Rede HTSPN para SMIL é uma qui-tupla 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 às 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.

Definição 3. Uma HTSPN interpretada é uma septupla SMIL 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.*
- $F_{PS}:P_{al}\rightarrow PS$, é uma função que associa um objeto *Presentation* com um objeto atômico ou ligação definido em R ou S, onde $P_{al} = \{p\in P_r \mid (\exists\, PT_r(p)\in\{atomic, link\})\} \cup \{p\in P \mid (\exists\, i\in I, PT_i(p)\in\{atomic, link\})\}\}$
- $F_{CS}:PS\rightarrow CS$ é uma função que associa um objeto *Presentation* com um objeto *Channel.*
- $F_{DS}:PS\rightarrow DS$ é uma função que associa um objeto *Presentation* com um objeto *Data.*

6.5 Tradução SMIL I-HTSPN para SMIL

Esta seção apresenta o procedimento de tradução de uma especificação SMIL I-HTSPN para uma aplicação SMIL. Os documentos SMIL são criados a partir da tradução da estrutura de conteúdo, de apresentação e conceitual do modelo SMIL I-HTSPN.

6.5.1 Estrutura de Conteúdo

Em SMIL I-HTSPN a estrutura do conteúdo é especificada pela lista de objetos *Data.* Estes objetos especificam a URI dos objetos. Na tradução de uma especificação SMIL I-HTSPN para SMIL, estas informações somente serão utilizadas na criação dos elementos que compõem o corpo de um aplicativo SMIL. Essa URI, conteúdo dessa classe, será incluída no valor do atributo *src,* localizado no respectivo *tag* da classe *presentation.* Isto será tratado na seção 6.5.3.

6.5.2 Estrutura de Apresentação

Em SMIL I-HTSPN a estrutura de apresentação é especificada pela lista de objetos *Presentation* e *Channel.* Na tradução de uma especificação SMIL I-HTSPN para SMIL estas informações serão utilizadas em dois momento: os objetos *Channel* serão

traduzidos pelas regiões definidas no *head* de um aplicativo SMIL e os objetos *Presentation* serão utilizados na criação dos elementos que compõem o corpo de um aplicativo SMIL. Isto será tratado na seção 6.5.3.

A tradução de um objeto *Channel* nos *tag Region* do SMIL se dá na forma de um mapeamento um a um dos atributos do objeto *Channel* nos parâmetros do *tag Region*. Por exemplo, os atributos do objeto *Channel*: *title , identifier, position, left, top, background-color e fit* são os parâmetros do *tag Region*, como mostra o exemplo abaixo:

<region id= "esquerda" title="Região posiconada a esquerda da tela" height="144" width="220" left="0" top="0" background-color="#000000" fit="fill"/>

6.5.3 Estrutura Conceitual

A forma de especificar a estrutura conceitual em SMIL é a composição hierárquica. O processo de tradução de uma especificação HTSPN para SMIL corresponde basicamente à tradução de uma Rede de Petri hierarquizada para uma composição hierárquica. Nesta composição hierárquica os nós internos representam as composições série e paralelo, e os nós-folhas correspondem aos elementos de mídia.

A estrutura de dados escolhida para representar a composição hierárquica foi a lista indexada. Portanto, em SMIL I-HTSPN, a estrutura conceitual é especificada através da criação de listas indexadas. Na tradução de uma especificação SMIL I-HTSPN para SMIL, primeiramente será preciso traduzir a rede raiz e sub-redes HTSPN para um conjunto de listas indexadas e depois traduzir essas listas indexadas para a estrutura textual de um arquivo SMIL.

Tradução de uma Rede HTSPN para a forma de listas indexadas

Para se fazer a tradução de qualquer Rede HTSPN para a forma de listas indexadas é utilizado o pseudocódigo abaixo. Este pseudocódigo trata-se de um procedimento de tradução visual, ou seja, a partir da representação gráfica do HTSPN. Estudos adicionais são necessários para a definição de um pseudocódigo para a implementação de um procedimento de tradução.

O nível de profundidade de uma Rede HTSPN, termo referenciado no pseudo-código da Figura 99, é dado da seguinte forma: considerando a transição com *n* arcos de

saída até a transição com *n* acros de entrada, o que estiver entre essas duas transições é considerado um nível de profundidade da Rede HTSPN. Por exemplo, a Figura 98 apresenta três níveis de profundidade. O I1 é o primeiro nível. Os elementos V1 e I2 são considerados o segundo nível de profundidade, pois como saiu duas setas da transição 1 e na transição 2 chegaram duas setas, então os elementos no interior dessas transições foram considerados um nível de profundidade. O T1 é o terceiro nível.

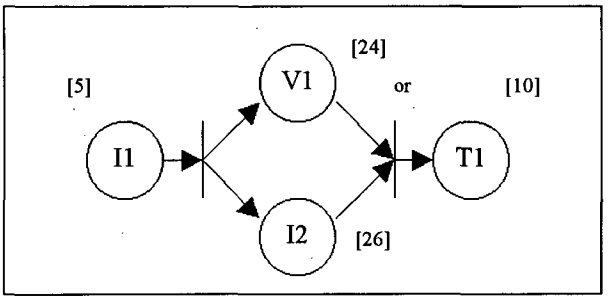


Figura 98. Ilustração dos níveis de profundidade em uma Rede HTSPN

```
1     INICIALIZA o contador de elementos PAR, SEQ
2     ENQUANTO houver níveis de profundidade da Rede HTSPN FAÇA
3          LÊ o nível da Rede HTSPN;
4          SE houver somente um elemento
5                  ENTÃO
6                     INSERE o elemento e sua duração na lista
7                  SENÃO
8                       INSERE o elemento par na lista
9                       ICREMENTA o contador de elementos PAR
10         IR para o próximo nível da Rede HTSPN
11    FIM_ENQUANTO
12
13    ENQUANTO SEQ e PAR for diferente de zero FAZER
14         SE PAR for diferente de zero
15           ENTÃO
16              LÊ o elemento PAR
17              CRIA uma nova lista e o elemento PAR aponta para ela
18              INSERE a política de disparo da transição
19              ENQUANTO houver elementos do tipo PAR FAZER
20                 SE houver elementos do tipo seq
21                   ENTÃO
22                      INSERE o elemento seq na lista
23                      INCREMENTA o contador de elementos SEQ
24                   SENÃO
25                      SE houver elementos do tipo PAR
26                        ENTÃO
27                           INSERE o elemento PAR na lista
28                           INCREMENTA o contador de elementos PAR
29                        SENÃO
30                           INSERE o elemento e sua duração na lista
```

```
31                DECREMENTA o contador de elementos PAR
32              FIM ENQUANTO
33              ENQUANTO SEQ for diferente de zero FAZER
34                LÊ o elemento SEQ da Rede HTSPN
35                CRIA uma nova lista e o elemento SEQ aponta para ela
36                ENQUANTO houver níveis de profundidade FAZER
37                  SE houver somente um elemento
38                    ENTÃO
39                      INSERE o elemento e sua duração na lista
40                    SENÃO
41                        INSERE o elemento PAR na lista
42                        INCREMENTA o contador de elementos PAR
43                FIM_ENQUANTO
44                DECREMENTA o contador de elementos SEQ
45              FIM ENQUANTO
46    FIM ENQUANTO
```

Figura 99. Pseudocódigo utilizado para se fazer a tradução de uma Rede HTSPN para a forma de listas indexadas

Para ilustrar o funcionamento de tradução de uma Rede HTSPN para a forma de listas indexadas será apresentado primeiramente um exemplo relativamente simples, que é demonstrado na Figura 100. Em seguida, será apresentado um exemplo (Figura 103) com componentes compostos.

Na Figura 100, a Rede HTSPN especifica um cenário multimídia com o seguinte comportamento: inicialmente apresenta uma imagem com a duração de 5u.t., depois ocorre uma apresentação em paralelo de um vídeo com a duração de 24u.t. e de uma outra imagem com a duração de 26u.t. Em seguida será apresentado um texto com uma duração de 10u.t. e um botão. Se esse botão for selecionado até 4u.t., será apresentado o áudio "A1".

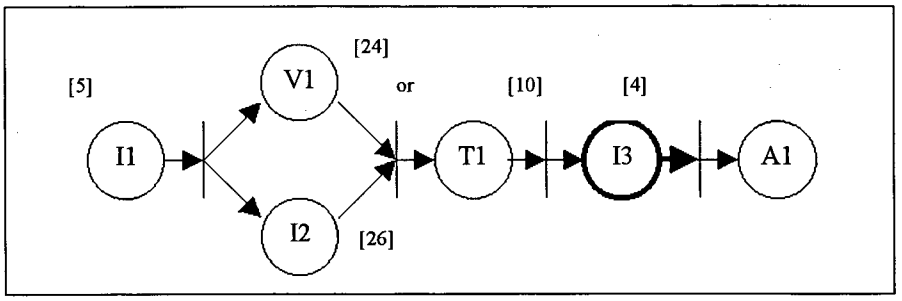


Figura 100. Exemplo utilizado para se fazer a conversão de uma Rede HTSPN para a forma de listas indexadas

Para se decompor a estrutura de uma Rede HTSPN para um conjunto de listas, primeiramente a Rede HTSPN deverá ser analisada como uma seqüência composta de

objetos e elementos pares. Em segundo lugar, deverão se decompor todas as listas do tipo ***par*** e do tipo ***seq*** (típica de componentes compostos) até possuir somente o apontador para os objetos do tipo mídia. A figura abaixo mostra a criação das listas utilizando-se o pseudocódigo da Figura 99. A Figura 102 mostra o conjunto de listas resultantes.

```
Linhas 2 e 4 são verdadeiras então INSERE o I1 e 5 na lista
Linhas 2 e 7 são verdadeiras então INSERE o PAR na lista e PAR = 1
Linhas 2 e 4 são verdadeiras então INSERE o T1 e 10 na lista
Linhas 2 e 4 são verdadeiras então INSERE o I3 e 4 na lista
Linhas 13, 14, 17 e 18 são verdadeiras então CRIA uma nova lista e
   INSERE a política de transição "or"
Linhas 19 e 29 são verdadeiras então INSERE o V1 e 24 na lista
Linhas 19 e 29 são verdadeiras então INSERE o I2 e 26 na lista
```

Figura 101. Criação das listas utilizando-se o pseudocódigo da Figura 99

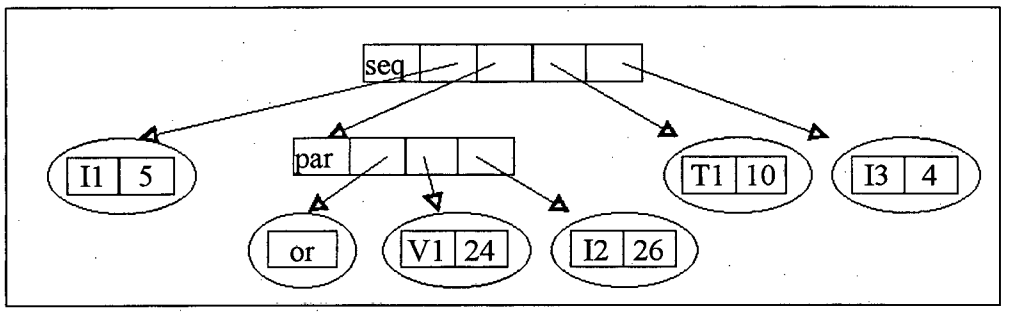


Figura 102. Tradução da Rede HTSPN da Figura 100 para a forma de listas indexadas

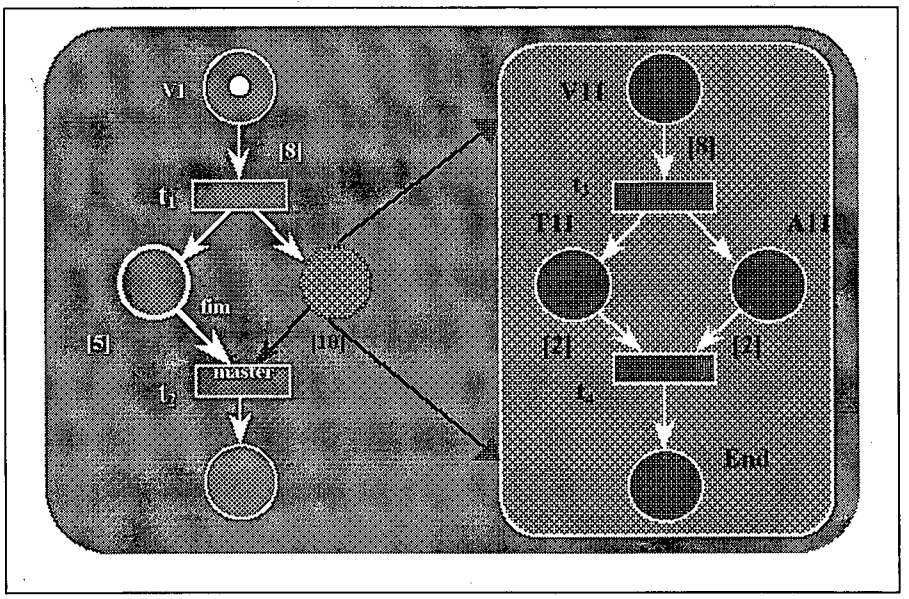


Figura 103. Exemplo utilizado para se fazer a conversão de uma Rede HTSPN com componentes compostos para a forma de listas indexadas

```
Linhas 2 e 4 são verdadeiras então INSERE o V1 e 8 na lista
Linhas 2 e 7 são verdadeiras então INSERE o PAR na lista e PAR = 1
Linhas 13, 14, 17 e 18 são verdadeiras então CRIA uma nova lista e
   INSERE a política de transição "id-ref"
Linhas 19 e 29 são verdadeiras então INSERE o I1 e 5 na lista
Linhas 19, 20 e 21 são verdadeiras então INSERE o PAR na lista e SEQ=1
Linhas 33, 36 e 38 são verdadeiras então INSERE o V11 e 8 na lista
Linhas 36, 38 e 40 são verdadeiras então INSERE o PAR na lista e PAR=1
Linhas 14, 17 e 18 são verdadeiras então CRIA uma nova lista e INSERE
   a política de transição "last"
Linhas 19 e 29 são verdadeiras então INSERE o T11 e 2 na lista
Linhas 19 e 29 são verdadeiras então INSERE o A11 e 2 na lista
```

Figura 104. Criação das listas referentes ao exemplo da Figura 103, utilizando-se o pseudocódigo da Figura 99

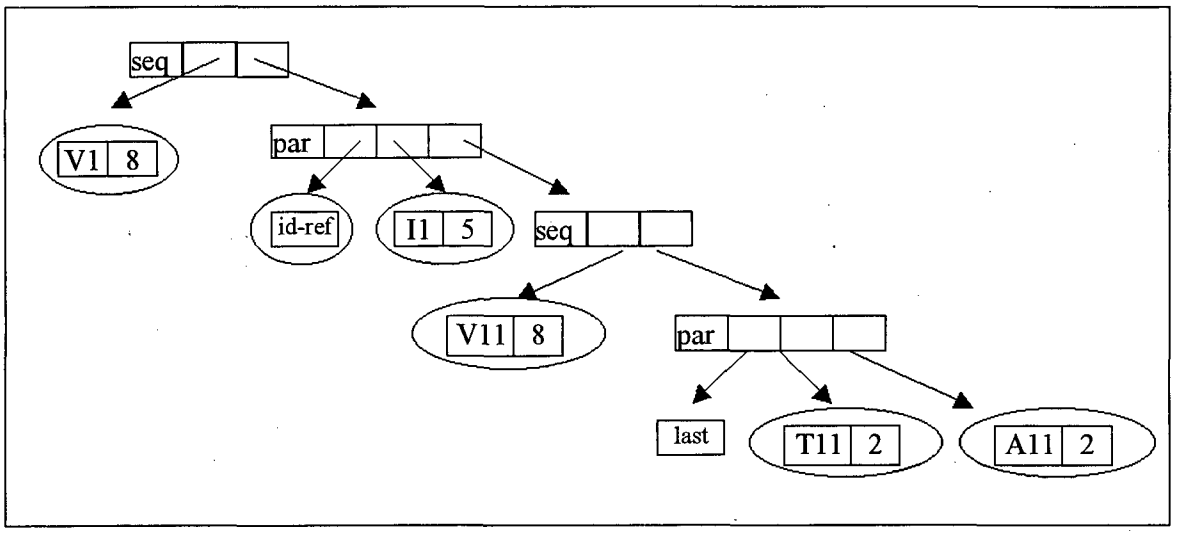


Figura 105. Tradução da Rede HTSPN da Figura 103 para a forma de listas indexadas

Tradução de um conjunto de listas indexadas para um arquivo SMIL

Toda tradução de um conjunto de listas indexadas para um arquivo SMIL inicia com uma composição "seq", pois o corpo de um arquivo SMIL é uma composição seqüencial.

Uma lista indexada pode possuir objetos de mídia (áudio, vídeo, etc.) ou outras listas do tipo "par" e "seq".

Quando um ponteiro da lista apontar para um objeto de mídia, ele deverá chamar um método que escreverá o *tag* no código SMIL correspondente ao objeto de mídia armazenado naquela posição da lista.

Quando existir um ponteiro para uma lista "par" deverá ser escrito os elementos <par> e </par> acrescentando, no interior desses elementos, as mídias que estão incluídas nessa lista. Além disso, será armazenado na primeira posição dessa lista o valor do atributo *endsync* (*first*, *id-ref* e *last*), que determinará o término da composição em relação ao término dos componentes. Se esse valor não for especificado na Rede HTSPN, assume-se que o *endsync* é igual a *last*, que é o valor *default* de uma composição paralela em um arquivo SMIL, não precisando ser especificado.

Na primeira posição do vetor, antes de ser chamado o método para escrever o *tag* correspondente do código SMIL, deverá ser implementado um método que escreva os elementos <body> e </ body>.

A tradução de uma lista SMIL I-HTSPN para um arquivo SMIL é realizada no mapeamento um a um dos objetos de mídia (resultantes da decomposição de um objeto de mídia e das listas do tipo *par*), como mostra o exemplo abaixo, resultante da decomposição da Figura 102.

```
<body>
          <img id= "I1"  />

          <par endsync = "first">
            <video id= "V1" />
            <img id= "I2"  />
          </par>

          <text id= "T1" />
          <a>
            <img id= "I3"  />
          </a>
</body>
```

Figura 106. Tradução do conjunto de listas indexadas da Figura 102 para um arquivo SMIL

Quando existir um ponteiro para uma lista "seq" deverá ser escrito os elementos <seq> e </seq> acrescentando, no interior desses elementos, as mídias que estão incluídas nessa lista.

6.6 Conclusão

Apesar de SMIL ser dita uma linguagem simples, é difícil manipular a função do tempo na forma textual. Além disso, a linguagem SMIL não permite aplicar técnicas de análise para se verificar a consistência temporal da aplicação.

Neste sentido, este capítulo propôs um modelo formal gráfico de mais alto nível, que foi obtido graças a uma nova interpretação do modelo I-HTSPN voltado para a especificação, análise e geração de código SMIL Boston. Esta nova interpretação, chamada SMIL I-HTSPN, herdará todas as técnicas de análise do HTSPN. Esse capítulo apresentou também o procedimento de tradução de uma especificação SMIL I-HTSPN para um código SMIL.

Com relação aos outros modelos interpretados do HTSPN, o modelo SMIL I-HTSPN apresenta algumas limitações, que são:

- o modelo SMIL I-HTSPN não permite definir uma tríplice IVT [x, n, y], onde x, y e z definem valores distintos para a duração mínima, nominal e máxima admissível do tratamento associado ao componente atômico. No SMIL I-HTSPN, o valor de x, n e z são iguais;
- na linguagem SMIL é possível descrever somente *loops* de reinicialização utilizando-se o atributo *a*. Essa linguagem não permite descrever *loops* que levam a um outro ponto interno do documento;
- as únicas transições tipadas do modelo I-HTSPN que tem correspondentes na linguagem SMIL são o *or*, o *master* e o *and*. As outras estratégias de sincronização do modelo I-HTSPN, *weak-and, strong-or, or-master, and-master, strong-master e weak-master,* não possuem equivalentes na linguagem SMIL.

Uma das vantagens do modelo SMIL I-HTSPN em relação ao modelo I-HTSPN é o reaproveitamento das características das regiões onde serão posicionados os elementos. Caso um elemento tenha criado uma região para seu uso, outro elemento poderá utilizar a mesma região. Além disso, uma mesma região pode ser utilizada para apresentar diferentes componentes.

Esta nova interpretação do modelo I-HTSPN também propiciou incluir a definição de componentes compostos, que não são definidos na linguagem SMIL Boston e que facilita a autoria de documentos multimídia. Segundo Hardman (1995), uma das vantagens dos componentes compostos é a possibilidade de agrupar itens em mini-apresentações que podem ser manipuladas como um todo.

No próximo capítulo será apresentado as conclusões dessa dissertação e será proposto alguns trabalhos futuros.

Capítulo 7. Conclusão

Para garantir que as páginas na Internet apresentem suporte para definição de relacionamentos de sincronização entre mídias, de forma que não apresente uma alta complexidade para autores que não possuem conhecimentos em programação, foi proposto pelo W3C a linguagem SMIL, na qual seus arquivos fontes podem ser escritos utilizando-se somente um editor de texto. A desvantagem é que a manipulação da noção de tempo na forma textual é difícil de manipular e modificar.

Por outro lado, os modelos gráficos têm a vantagem de ilustrar de maneira gráfica a semântica das relações temporais. Este tipo de modelo simplifica a especificação das restrições temporais e reduzem o tempo de criação do documento.

Levando-se em conta essa motivação, essa dissertação propôs um modelo gráfico de mais alto nível, que permite a especificação gráfica e análise do documento, para posterior tradução para o código SMIL Boston.

A especificação gráfica de documentos SMIL foi obtida graças a uma nova interpretação do modelo I-HTSPN voltada para a especificação, análise e geração de código SMIL Boston. Esta nova interpretação foi chamada de SMIL I-HTSPN e herdará todas as técnicas de análise do HTSPN.

Outra vantagem do uso do modelo I-HTSPN em relação ao uso direto da linguagem SMIL é a capacidade de se definir componentes compostos, que não são definidos na linguagem SMIL Boston e que facilita a autoria de documentos multimídia. Segundo Hardman (1995), uma das vantagens dos componentes compostos é a possibilidade de agrupar itens em mini-apresentações que podem ser manipuladas como um todo.

Devido ao pouco poder de expressão da linguagem SMIL Boston, o modelo SMIL I-HTSPN terá algumas limitações em relação ao modelo I-HTSPN, entre elas a não possibilidade de definir: um IVT relacionado a inclusão da mídia no documento,

loops internos e algumas regras de transições do modelo I-HTSPN. Além disso, o componente composto reutilizado implica na replicação de código.

O modelo SMIL I-HTSPN não tem mecanismos explícitos para especificar a validade temporal dos componentes de uma aplicação. Ele permite especificar apenas uma duração ideal através do atributo *dur*, que fixa uma duração exata para a apresentação da mídia. Não será possível definir uma tríplice IVT [x, n, y], onde x, y e z definem valores distintos para a duração mínima, nominal e máxima admissível do tratamento associado ao componente atômico. No SMIL I-HTSPN, o valor de x, n e z são iguais. Portanto, na representação gráfica do modelo, aos arcos de saída dos lugares será associada apenas a duração nominal do componente.

Na linguagem SMIL é possível descrever somente *loops* de reinicialização (Figura 90b), utilizando-se o atributo *a*. Essa linguagem não permite descrever *loops* que levam a um outro ponto interno do documento (Figura 90a).

As únicas transições tipadas do modelo I-HTSPN que tem correspondentes na linguagem SMIL são o *or*, o *master* e o *and*. As outras estratégias de sincronização do modelo I-HTSPN, *weak-and, strong-or, or-master, and-master, strong-master e weak-master,* não possuem equivalentes na linguagem SMIL, pois ela não leva em consideração o não-determinismo temporal dos sistemas distribuídos.

Na tradução do componente composto do modelo SMIL I-HTSPN para o código SMIL Boston, o componente composto seria uma forma de "alias", que no momento da tradução seria substituído pelo seu conteúdo. Caso o componente composto seja reutilizado, ocorrerá a replicação de código. Isto, pois a reutilização do código não é possível em SMIL Boston.

Esse trabalho foi elaborado seguindo a 4ª revisão da especificação da linguagem SMIL Boston. Como essa especificação ainda passará por outras revisões, o trabalho deverá ser revisado quando sair a especificação final da linguagem.

Com base nos resultados obtidos neste trabalho pode-se recomendar os seguintes trabalhos futuros:

- implementar um editor SMIL I-HTSPN que herdaria as facilidades do modelo I-HTSPN acrescentando novos recursos a linguagem SMIL Boston, entre eles a

inclusão de componentes compostos. Com um editor, seria mais fácil manipular e modificar a noção de tempo, por ser um sistema gráfico. Além disso, por esse modelo ser baseado em Redes de Petri, que é um modelo formal, permite a construção de uma semântica precisa e não ambígua de documentos e permite a utilização de técnicas de análise sofisticados, como a simulação, validação e verificação do comportamento do documento;

- estender a linguagem SMIL Boston, a fim de acrescentar outras funcionalidades do I-HTSPN, entre elas, a definição de mecanismos de tolerância de sincronização, *loops* internos e incluir outras estratégias de sincronização, além do *or*, *and* e *master*. Isso é possível devido ao fato da linguagem SMIL Boston seguir o padrão XML, que é uma linguagem extensível (metalinguagem).

Capítulo 8. Referências Bibliográficas

[Ackermann, 94]P. Ackermann. Direct Manipulation of Temporal Structures in a Multimedia Application Framework. In proc. of the ACM Multimedia 94.

[Allen, 83] J.F. Allen. Maintaining Knowledge about Temporal Intervals. Communications of the ACM 26(11):832-843, 1983.

[Apple, 91] Apple Computer Inc. QuickTime Developer's Guide. Developer technical publications, 1991.

[Apple, 94] Multimedia Demystified - A Guide to the World of Multimedia from Apple Computer, Inc. Random House/NewMedia Series, 1994.

[Authorware, 89]Inc. Authorware Professional Manual. Redwood City, CA, 1989.

[Blakowski, 92] G. Blakowski, J. Jens Hübel, U. Langrehr, M. Mühlhäuser. Tool Support for the Synchronization and Presentation of Distributed Multimedia. Computer Communication, 15(10):611-618. December, 1992.

[Botafogo, 95] R. Botafogo, D. Mossé. The MORENA Model for Hypermedia Authoring and Browsing. In proc. of the IEEE Int. Conf. on Multimedia Computing and Systems, pp. 42-49, Washington, DC, May 1995.

[Brown, 89] H. Brown. Standards for structured documents. The Computer Jornal, vol. 32, nº 6, junho 1989, pp. 505-514.

[Buford, 94] Multimedia Systems. John F. Koegel Buford. ACM Press - SIGGRAPH Series - New York, New York, 1994.

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

[Drapeau, 91] G.D. Drapeau, H. Greengiled, H. MAEstro - A Distributed Multimedia Authoring Environment. In proc. of the USENIX Summer'91 Conference, pp. 315-328. Nashville TN, 1991.

[Eyes, 92] Eyes 2.0 Reference Manual. Center for Productivity Enhancement. University of Massachusetts-Lowell. 1992.

[FAQ, 97] Multimedia Authoring Systems FAQ. Version 2.13. http://www.tiac.net/users/jalisglar/MMASFAQ.HTML. June 1997.

[Fluckiger, 95] François Fluckiger. Understanding Networked Multimedia: Applications and Technology. Prentice Hall International (UK) Limited, 1995.

[Gibbs, 91] S. Gibbs. Composite Multimedia and Active Objects. In proc. of the OOPSLA'91. Phoenix, Arizona. October, 1991.

[Ginige, 95] A. Ginige, D.B. Lowe, J. Robertson. Hypermedia Authoring. IEEE Multimedia 2(4):24-35, 1995.

[Haindl, 96] M. Haindl. Multimedia Synchronization. Computer Science/Department of Interactive Systems. IEEE Journal on Selected Areas in Communication 14(1):73-83, 1996.

[Hamblin, 72] C.L. Hamblin. Instants and Intervals. In proc. of 1st Conf. Int. Soc. For the Study of Time, J.T. Fraser et l. (Eds.), Springer-Verlag, pp. 324-331, New York, 1972.

[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.

[Hardman, 95] L. Hardman, D.C.A Bulterman. Authoring Support for Durable Interactive Multimedia Presentations. STAR Report in Eurographics'95.

[Herman, 94] I. Herman, G.J. Reynolds. MADE: A Multimedia Application Development Environment. In proc. of the Int. Conf. on Multimedia Communication and Systems, pp. 184-193, 1994.

[Hirzalla, 95] N. Hirzalla, B. Falchuk and A. Karmouch. *A Temporal Model for Interactive Multimedia Scenarios*. IEEE Multimedia, pp. 24-31, Fall 1995.

[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 86] ISSO/IEC 8879 Information Processing – Text and Office Systems – Standard Generalized Markup Language. Outubro 1986.

[ISO 8613] ISO 8613 Information Processing - Text and Office Systems - Office Document Architecture (ODA) and Interchange Format (ODIF), 1998.

[Klas, 90] W. Klas. E.J. Neuhold, M. Schrefl. Using an Object-Oriented Approach to Model Multimedia Data. Computer Communication 13(4):204-216, 1990.

[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.

[Newcomb, 91] S. Newcomb et al. The HyTime Hypermedia/Time-based Document Structuring Language. Communication of the ACM 34(11):159-166, 1991.

[Rodrigues, 99] L. M. Rodrigues; R. F. Rodrigues; D. Muchaluat-Saade; L. F. G. Soares. Acrescendo facilidades NCM a documentos SMIL. Anais do Simpósio Brasileiro de Sistemas Multimídia e Hipermídia 1999.

[Schloss, 94] G.A. Schloss, M.J. Wynblatt. Building Temporal Streuctures in a Layered Multimedia Data Model. In proc. ACM Multimedia'94, pp. 271-278, 1994.

[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.

[Shackelford, 93] D.E. Shackelford, J.B. Smith, F.D. Smith. The Architecture and Implementation of a Distributed Hypermedia Storage System. Technical Report 93-013, Department of Computer Science, The University of North Carolina at Chapel Hill, 1993.

[Stanek, 99] W. R. Stanek. SMIL: o novo formato Web para multimídia. Disponível por www em http://www.zdnet.com.br/revistas/pcmag/materias/pctech/ pcte0399-1.cfm, agosto 1999.

[Stotts, 90] P.D. Stotts, R. Furata. Temporal Hyperprogramming. Journal of Visual Languages and Computing 1:237-253, 1990.

[Vazirgiannis, 93] M. Vazirgiannis, M. Hatzopoulos. A Script Based Approach for Interative Multimedia Applications. In proc. of Multimedia Modelling'93, pp. 129-143, Singapore, November, 1993.

[XML 98] "Extensible Markup Language (XML) 1.0" T. Bray, J. Paoli and C.M. Sperberg-McQueen. W3C Recommendation 10 February 1998. Disponível por www em http://www.w3.org/TR/REC-xml, fevereiro 1998.

[W3C 98] Synchronized Multimedia Working Group of the World Wide Web Consortium. Synchronized Multimedia Integration Language (SMIL) 1.0 Specification. W3C Recommendation. Disponível por www em http://www.w3.org/TR/REC-smil/, junho 1998.

[W3C 00] Synchronized Multimedia Working Group of the World Wide Web Consortium. Synchronized Multimedia Integration Language (SMIL) Boston Specification. W3C Recommendation. Disponível por www em http://www.w3.org/TR/smil-boston, junho de 2000.

[Wahl, 94] T. Wahl, K. Rothermel. Representing Time in Multimedia Systems. In proc. of the International Conference on Multimedia Computing and Systems (ICMCS'94), pp. 538-543, Boston, 1994.

[Wang, 95] H.K. Wang, J.L.C. Wu. Interactive Hypermedia Applications: A Model and Its Implementation. Software-Practice and Experience 25(9):1045-1063, 95.

[Willrich, 96] 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, 96a] 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.

[Willrich, 96b] 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, 97] R. Willrich, P. de Saqui-Sannes. *Concepção Formal de Aplicações Multimídia Java*. Anais do XIV Simpósio Brasileiro de Redes de Computadores, pp 135-149, São Carlos (SP), 1997.

[Wynblatt, 95] M. Wynblatt. Position Statement on Multimedia Synchronization. In proc. of the IEEE Workshop on Multimedia Synchronization (Sync'95). In URL: http://spiderman.bu.edu/sync95/sync95.html. Tysons Corner, Virginia, 95.

[Yang, 96] C.C. Yang, J.H. Huang. A Multimedia Synchronization Model and Its Implementation in Transport Protocols. IEEE Journal on Selected Areas in Communications 14(1):212-255, 1996.