MUSE - Um Ambiente para Modelagem de Aplicações Multimídia Interativas com Tradutor para E-LOTOS

Luciano Paschoal Gaspary
Maria Janilce B. Almeida

Universidade Federal do Rio Grande do Sul
Instituto de Informática
Pós-Graduação em Ciência da Computação
Campus do Vale, Bloco IV - Bento Gonçalves, 9500 - Agronomia - CEP 91591-970
Porto Alegre, RS - Brasil
E-mail: {paschoal, janilce}@inf.ufrgs.br

Resumo

Este trabalho apresenta MUSE, um ambiente gráfico para modelagem de aplicações multimídia interativas. Através de uma interface gráfica avançada e de um novo modelo de autoria de alto nível, é possível a criação de sistemas complexos de forma rápida e intuitiva. O modelo de autoria proposto neste trabalho e adotado pelo ambiente manipula componentes multimídia dispersos em uma redes de computadores, permitindo a definição de limiares aceitáveis de atraso para a apresentação destes componentes. Pela grande expressividade do modelo, no entanto, podem ser geradas especificações com inconsistências temporais. Por esta razão, o ambiente provê ainda especificações E-LOTOS que, em breve, serão utilizadas para fins de análise, permitindo a validação dos requisitos temporais das aplicações definidas pelo autor.

Abstract

This work presents MUSE, a graphical environment for modeling interactive multimedia applications. Through an advanced graphic interface and a new high-level authoring model, it is possible to create complex systems in a fast and intuitive way. The authoring model proposed in this work and adopted by the environment deals with multimedia components distributed in a computer network, allowing the definition of acceptable delay thresholds to the presentation of these components. Due to the large expressiveness of the model, however, specifications with temporal inconsistencies may be generated. For this reason, the environment also provides E-LOTOS specifications, which will soon be used for analysis purpose, allowing the validation of the temporal requirements of the applications defined by the author.

1. Introdução

Os anos 90 têm sido marcados pela utilização de aplicações multimídia em diversos setores da atividade humana tais como educação, medicina e entretenimento. Estas aplicações tem se sofisticado cada vez mais ao longo do tempo, sendo que atualmente são executadas em ambientes distribuídos, operando transparentemente em plataformas heterogêneas.

A possibilidade de os componentes multimídia que fazem parte da aplicação estarem dispersos em uma rede impacta o processo de criação e modelagem destas aplicações. Os usuários devem fornecer às ferramentas de autoria informações como restrições temporais, definindo limiares aceitáveis de atraso à apresentação dos componentes que compõem o sistema e, em alguns casos, estabelecendo a apresentação de componentes multimídia alternativos.

A definição destas restrições é realizada com base em um modelo de sincronização, que dita as regras sobre como os componentes de uma aplicação podem ser relacionados no tempo. Vários modelos de sincronização têm sido propostos [1]. Na sua maioria, eles são flexíveis e bastante expressivos. Em razão desta ampla expressividade dos modelos, no entanto, as especificações resultantes podem ser fonte de incoerências, não havendo garantia sobre a consistência temporal dos componentes multimídia envolvidos.

Uma alternativa seria utilizar diretamente uma técnica de descrição formal (TDF) para descrever as aplicações, viabilizando sua análise e, consequentemente, garantindo sua consistência. A desvantagem desta utilização direta, contudo, é a alta complexidade inerente às TDFs. Evidencia-se, assim, a necessidade de contar com um modelo estruturado de alto nível para especificar aplicações multimídia interativas. Ao mesmo tempo, deseja-se que o resultado desta especificação seja traduzido para uma TDF, sob a qual seja possível aplicar métodos de simulação e verificação.

Dentro deste contexto, considerando alguns conceitos presentes em MHEG-5 [2] e alguns modelos de sincronização, criou-se um modelo de autoria de aplicações multimídia interativas. Baseado neste modelo, desenvolveu-se o ambiente MUSE *(Interactive Networked MUltimedia Applications Specification Environment)*, que permite ao usuário definir, em alto nível e com elevado poder de expressão, uma apresentação multimídia em conformidade com o padrão MHEG-5. Alternativamente às aplicações MHEG-5, podem ser geradas aplicações Java [3]. A adoção de MHEG-5 ou Java permite o compartilhamento de informação multimídia sem se preocupar com a plataforma ou sistema operacional utilizado, viabilizando a especificação e desenvolvimento de aplicações portáveis. Outra funcionalidade do ambiente é a tradução das especificações para a TDF E-LOTOS possibilitando que, em breve, elas sejam validadas.

Este trabalho faz parte do projeto DAMD [4] (*Design de Aplicações Multimídia Distribuídas*), financiado pelo PROTEM Fase III do CNPq, que tem por objetivo fornecer uma metodologia que cubra o ciclo completo das aplicações multimídia distribuídas e que permita a um autor não especializado em métodos formais desenvolver essas aplicações naturalmente. Em linhas gerais, o projeto tem sido desenvolvido de acordo com a figura 1. Com a utilização de MUSE, as aplicações multimídia podem ser editadas e automaticamente traduzidas para a TDF E-LOTOS. Outras etapas em fase de desenvolvimento no projeto têm se preocupado em criar mecanismos para validar estas especificações e apresentar, de forma legível, os resultados ao autor. É desejável que o processo especificação-validação se repita até que as incoerências sejam eliminadas. Após esta etapa, geram-se aplicações MHEG-5 ou aplicações Java, que podem ser executadas, respectivamente, por um MHEG-5 *engine* ou por uma máquina virtual JAVA. Informações sobre o processo de geração de aplicações JAVA podem ser obtidas em [3].

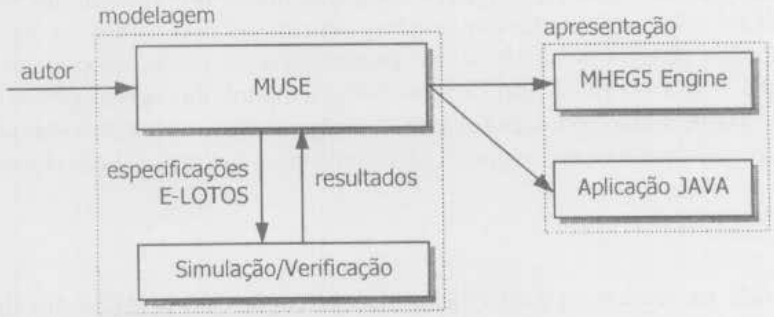


Figura 1 - Estrutura do projeto DAMD

O artigo está organizado da seguinte forma: a seção 2 apresenta um estudo de aspectos relevantes a serem considerados na autoria de aplicações, relacionando-os com alguns modelos de sincronização multimídia apontados pela literatura. A seção apresenta ainda o modelo de autoria proposto. Na seção 3 são apresentados aspectos básicos da TDF E-LOTOS e o estudo do mapeamento do modelo de autoria para esta linguagem. A seção 4 ilustra a funcionalidade do ambiente e na seção 5 são apresentadas as considerações finais.

2. Modelo de Autoria Proposto

A especificação de aplicações multimídia é realizada com base em três aspectos fundamentais: estruturação lógica da aplicação, estabelecimento de relações temporais e definição espacial entre os componentes multimídia que a compõem. A estruturação lógica se preocupa em

oferecer mecanismos de abstração, objetivando a obtenção de uma visão ampla e estrutural da aplicação [5]. A especificação do comportamento temporal, por sua vez, implica na definição de relações de sincronização entre os componentes multimídia envolvidos. A sincronização espacial, por fim, prima por ajustar o posicionamento destes mesmos componentes, de acordo com os dispositivos de saída (vídeo) [6].

O relacionamento temporal entre os componentes que constituem a aplicação está associado a um modelo de sincronização. Este impõe regras sobre como estes componentes podem se relacionar entre si. Diversos modelos têm sido propostos na literatura, obedecendo, quando possível, a alguns requisitos básicos apontados em [1]: suporte a consistência de objetos (componentes multimídia) e manutenção das especificações; componentes multimídia como uma unidade lógica da especificação; possibilidade de representar interação do usuário; facilidade na descrição das relações de sincronização; suporte à definição de QoS e hierarquização da especificação, viabilizando a manipulação de cenários grandes e complexos.

Um dos modelos de sincronização mais adotados por ferramentas de autoria existentes atualmente é o baseado em linha do tempo [7]. Este, no entanto, apresenta uma série de limitações, como a dificuldade de modularizar a aplicação e estabelecer relações entre componentes multimídia de duração variável ou desconhecida (interação do usuário, por exemplo) [1].

Modelos baseados em restrições como o HTSPN (*Hierarchical Time Stream Petri Nets*) [8] e modelos orientados a objetos, por sua vez, não apresentam estes problemas. Em contrapartida, esses modelos ainda não têm sido amplamente utilizados em ferramentas comerciais pela dificuldade em se entender as especificações, resultante da complexidade destes modelos [9].

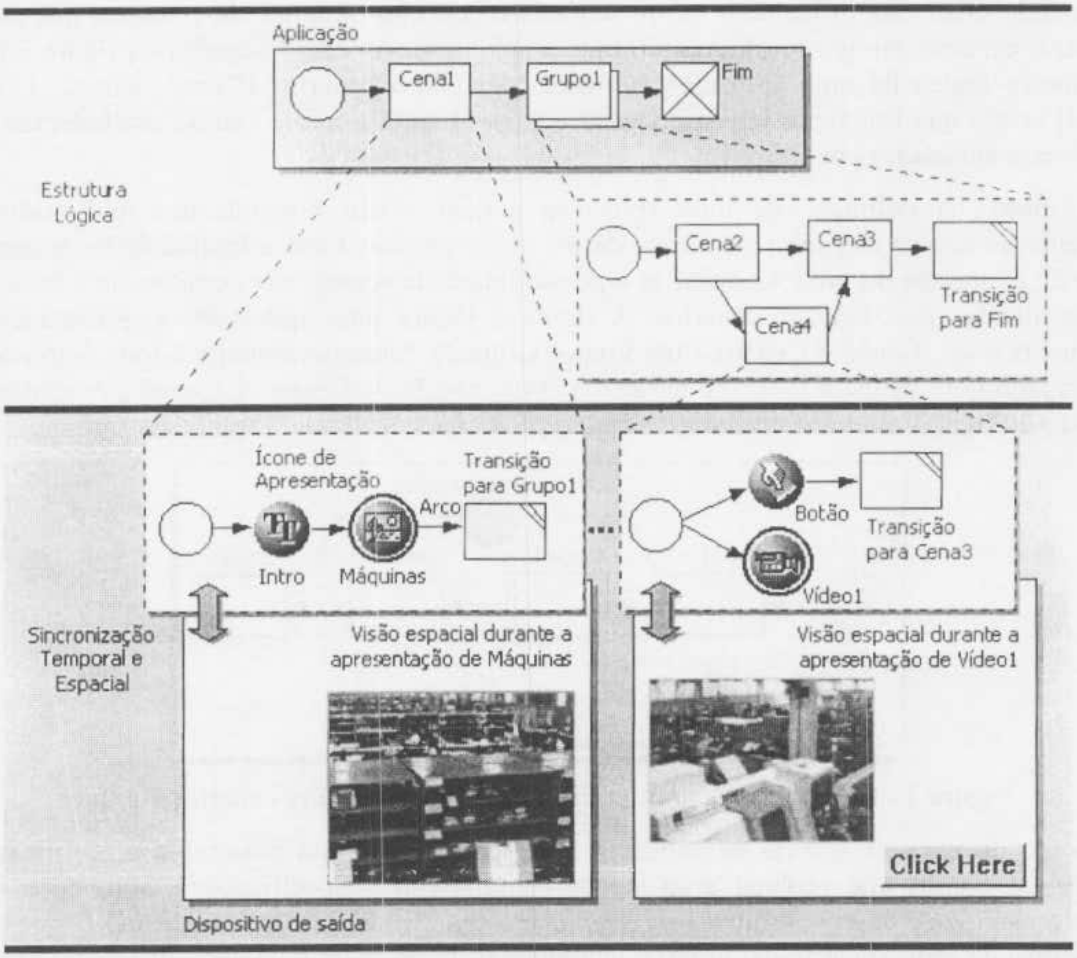


Figura 2 - Esquema geral do modelo de autoria proposto

Com o objetivo de minimizar a relação antagônica existente entre poder de expressão e facilidade de uso propõe-se, neste trabalho, um modelo de autoria que agrega mecanismos de estruturação lógica das aplicações a um modelo de sincronização similar aos modelos baseados em redes de Petri. Utilizando o conceito de ícones, criou-se uma representação gráfica simplificada que provê ao autor mecanismos facilitadores ao processo de especificação. A figura 2 apresenta um esquema geral do modelo proposto. O nível de estruturação lógica opera sobre os conceitos de cenários e grupos. A definição das sincronizações temporais é realizada em cada cenário com a utilização de uma rede simplificada. A sincronização espacial permite que os componentes multimídia sejam organizados quanto ao seu posicionamento em relação ao dispositivo de saída.

2.1. Estruturação Lógica

A complexidade das aplicações multimídia aumenta com o crescimento do número de componentes multimídia envolvidos e, por conseqüência, com os diversos relacionamentos temporais estabelecidos entre eles. Esta é a razão fundamental pela qual a especificação destas aplicações em um único plano é inadequada. Para resolver este problema, o conceito de cenas ou cenários foi incorporado ao modelo. Na realidade, este conceito foi extraído da própria norma MHEG-5, onde aplicações multimídia são organizadas como um conjunto de cenários relacionados por eventos, que provêm a navegação entre eles. O mesmo ocorre com o modelo proposto: cada cenário pode ser visto como uma *caixa preta* com um comportamento interno que, sob determinadas condições, habilita a apresentação de outros cenários.

A utilização deste conceito, contudo, não resolve de todo o problema da complexidade, uma vez que uma especificação com um número elevado de cenários será dificilmente compreendida. Procurando sanar esta dificuldade, criou-se ainda o conceito de grupo de cenários, que permite organizar cenários em grupos, hierarquizando a aplicação. A porção superior da figura 2 ilustra a estrutura lógica de uma aplicação composta de quatro cenários (Cena1, Cena2, Cena3 e Cena4) sendo que três deles (Cena2, Cena3 e Cena4), pelo grau de coesão estabelecido entre eles, foram agrupados em Grupo1.

Além disso, na definição de uma aplicação podem existir componentes multimídia cuja apresentação não se enquadra ao escopo de um único cenário. Com a finalidade de aumentar o poder de expressão do modelo, criou-se a possibilidade de representar componentes multimídia compartilhados por diversos cenários. A figura 3 ilustra uma aplicação organizada em três cenários (Cena1, Cena5 e Cena6) e um grupo (Grupo1). Simultaneamente a toda a aplicação é apresentado uma imagem (Logo) e durante a apresentação de Grupo1 e Cena5 é executado um áudio (Áudio1).

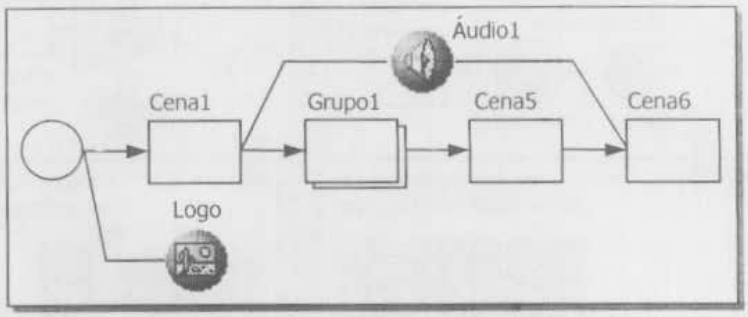


Figura 3 - Componentes multimídia compartilhados entre cenários e grupos

Do ponto de vista da autoria de aplicações, a estrutura proposta possibilita a reutilização de cenários e grupos que venham a se repetir em diferentes especificações. Somado a isto, o modelo viabiliza o desenvolvimento de *templates* - cenários básicos pré-contruídos - que tornam o processo de especificação evolutivo e incremental. Pode-se ter uma série de *templates*, sendo que o processo de especificação, neste caso, se resume a unir estes diferentes cenários, reduzindo drasticamente o tempo de desenvolvimento.

2.2. Sincronização Temporal

Este trabalho propõe a definição da sincronização em cada cenário individualmente, contribuindo para a diminuição da complexidade no processo de especificação. Como já mencionado anteriormente, propõe-se que as relações temporais entre os elementos que fazem parte de um cenário sejam estabelecidas a partir de uma rede de Petri simplificada. Os tópicos a seguir apresentam como a sincronização entre componentes multimídia pode ser definida.

a) Sincronização Básica

A apresentação de componentes multimídia pode ocorrer seqüencialmente ou simultaneamente. Na apresentação seqüencial, o início da apresentação de um componente depende do final da apresentação de outro. Na figura 2 aparecem os dois tipos de sincronização básica. Em Cena1 é definida a apresentação de um texto (Intro) seguido da apresentação de uma imagem (Máquinas). Em Cena4, por sua vez, tem-se a apresentação simultânea de um vídeo (Vídeo1) e um botão (Botão).

b) Duração da Apresentação de Componentes e Limiares Aceitáveis de Atraso

A cada componente multimídia está associada uma duração mínima e uma duração máxima de apresentação. No caso de uma imagem ou de um texto estes valores são equivalentes por se tratarem de componentes multimídia independentes do tempo. Observe na figura 4 que o tempo de apresentação da imagem *Slide1* é 10 segundos.

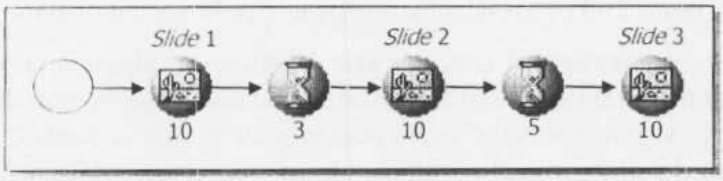


Figura 4 - Modelagem de atrasos intencionais

Quando se depara com componentes como áudio e vídeo, no entanto, é importante determinar uma duração mínima e máxima de apresentação, uma vez que dificilmente eles serão apresentados à taxa nominal em decorrência de problemas como tráfego de rede. A representação destas durações é dada por um intervalo. Na figura 7, por exemplo, o fragmento de animação ANI_P1 (Cena3) possui uma duração mínima prevista de 2 segundos e uma duração máxima de 3 segundos.

Para possibilitar a modelagem de um atraso intencional entre a apresentação de dois componentes multimídia consecutivos, pode-se utilizar um ícone especial; este não possui um componente associado e apenas representa a passagem do tempo. A figura 4 ilustra três *slides* (*Slide1*, *Slide2* e *Slide3*) sendo apresentados seqüencialmente, sendo que entre o primeiro e o segundo é modelado um atraso de 3 segundos e entre o segundo e o terceiro, um atraso de 5 segundos.

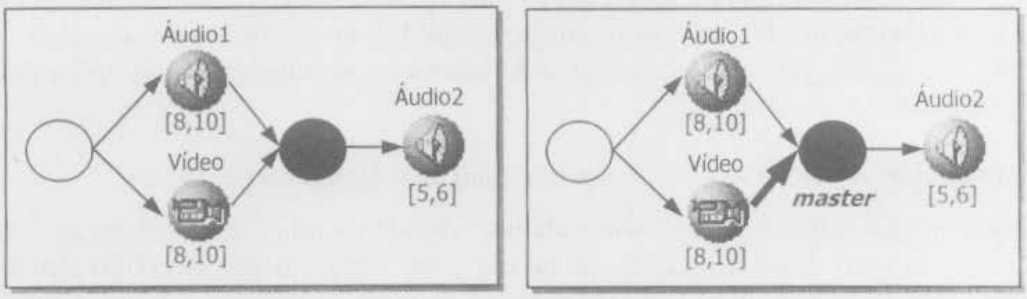
c) Interação do Usuário e Transição de Cenários

A interação do usuário corresponde, por exemplo, a um clique em um botão ou a seleção de um objeto. Ela é representada neste modelo como um ícone cujo tempo de apresentação é indeterminado, variando entre os valores mínimo e máximo associados. Quando o limiar máximo é atingido, o cenário prossegue com sua apresentação. É possível ainda especificar um botão sem tempo máximo; neste caso, a aplicação só evoluirá após a interação. Cena3 (figura 7) apresenta um botão (Interação) com duração de [0,20]. Isto significa que o usuário terá 20 segundos para selecioná-lo. Passado este tempo, o componente P4 será apresentado.

Normalmente, a interferência do usuário está associada a uma transição entre cenas. Transição é o ícone que efetiva a possibilidade de navegação entre elas. Sua execução implica na imediata suspensão da apresentação de todos os componentes multimídia associados ao cenário corrente e no início da apresentação de um novo cenário. Em Cena1 (figura 2), observa-se a transição para Grupo1. Uma transição para o final da aplicação pode ser observada em Cena4, na mesma figura. Estas transições são modeladas de forma consistente com a estrutura lógica estabelecida.

d) Políticas de Disparo de Pontos de Sincronização

Os pontos de sincronização permitem associar o início da apresentação de um ou mais componentes multimídia a diferentes políticas relacionadas à finalização da apresentação de outros componentes que convergem para estes pontos. A figura 5a ilustra um cenário onde são apresentados em paralelo um vídeo (Vídeo) e um áudio (Áudio1), seguidos de outro áudio (Áudio2). Esta representação implica na obrigatoriedade de ambos terminarem sua apresentação ou expirar seu tempo de duração para que Áudio2 passe a ser executado.



(a) Ponto de sincronização

(b) Política *master* de disparo do ponto de sincronização

Figura 5 - Pontos de sincronização e políticas de disparo

Para permitir um maior poder de especificação, adotou-se algumas políticas amplamente comentadas na literatura. Elas permitem a associação de comportamentos diferentes aos pontos de sincronização [10]. As políticas suportadas pelo modelo são as seguintes:

- *Master*: um ponto de sincronização é disparado quando a apresentação de um componente mestre é encerrada, interrompendo todos os outros. Esta seria a política a ser utilizada no exemplo da figura 5a acima no caso de se desejar que o final da apresentação do vídeo (mestre) interrompesse a execução de Áudio1, iniciando Áudio2 (vide figura 5b). O componente mestre é identificado pela presença do caracter *m* ou pela palavra *master* próxima a ele.
- *Earliest*: o ponto de sincronização é disparado quando a apresentação do primeiro componente for encerrada, interrompendo todos os componentes que estão executando simultaneamente. Graficamente, esta política é representada pela presença do caracter *e* ou pela palavra *earliest* próxima ao ponto de sincronização.
- *Latest*: a ausência de indicação próxima ao componente ou ao ponto de sincronização indica que todos os componentes que antecedem este ponto serão executados (ou finalizarão pelo esgotamento de sua duração máxima de apresentação) antes de ele ser disparado (figura 5a).

e) Instantes de Sincronização

Em MUSE, a sincronização entre componentes em outros instantes que não o início ou fim de suas apresentações requer a divisão dos mesmos em partes, criando um conjunto de segmentos. A granularidade desta divisão está diretamente associada ao grau de precisão desejado para a sincronização. A figura 6 ilustra a sincronização de duas legendas com um vídeo, onde este é dividido em quatro segmentos denotados respectivamente pela apresentação do quadro 1 ao 29, do 30 ao 44, do 45 ao 89 e, por fim, do quadro 90 ao 120. A primeira legenda é apresentada simultaneamente ao segundo segmento do vídeo e a segunda legenda ao mesmo tempo que o terceiro.

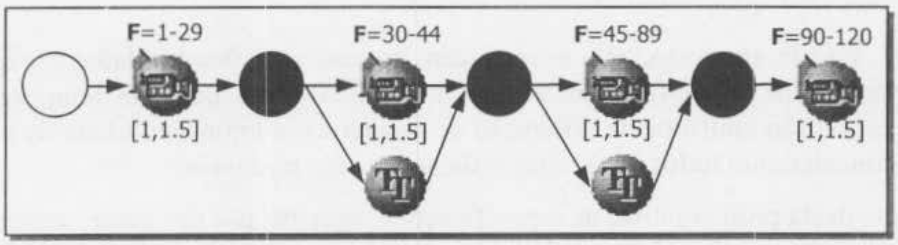


Figura 6 - Sincronização de um vídeo com legendas

2.3. Sincronização Espacial

A sincronização espacial permite ao autor organizar o posicionamento dos componentes visíveis de um cenário. Não é possível realizar a sincronização espacial considerando um tempo determinado transcorrido após o início da apresentação da cena. Exemplificando, não é possível visualizar Cena2 (figura 7) considerando 4 segundos após o início de sua execução. Idealmente, neste instante deveria estar sendo apresentado o componente P1; nada impede, devido às variações temporais admissíveis, que em alguma execução o componente RI esteja sendo apresentado, já que este possui uma duração que pode variar entre 3 e 4.5 segundos. Por esta razão, a sincronização espacial é realizada sempre com base na apresentação de um componente. A disposição espacial dos componentes de Cena2 (figura 7) durante a apresentação de P1, por exemplo, permitirá organizar somente o componente P1. Se na definição deste cenário houvesse outros componentes definidos para serem apresentados simultaneamente a P1, estes componentes também apareceriam nesta visão.

2.4. Um Exemplo de Utilização do Modelo

O exemplo ilustrado na figura 7 modela uma aplicação apresentada em [1], onde inicialmente um vídeo (VD) e um áudio (AU) são apresentados simultaneamente. A seguir, é apresentada a repetição de uma interação gravada do usuário (RI), uma seqüência de três *slides* (P1-P3) e uma animação (ANI) que é parcialmente comentada por uma seqüência de áudio (Áudio2). Após o seu início, uma questão de múltipla escolha é apresentada ao usuário (Interação). Caso o usuário faça a seleção, uma imagem final (P4) é apresentada. Naturalmente, esta é apenas uma das diversas formas de representar essa aplicação. A facilidade no entendimento da mesma é obtida principalmente pelo bom senso do autor no momento da especificação. Na figura a seguir, as durações associadas aos ícones são fictícias; o exemplo original [1] é apresentado sem a definição das mesmas.

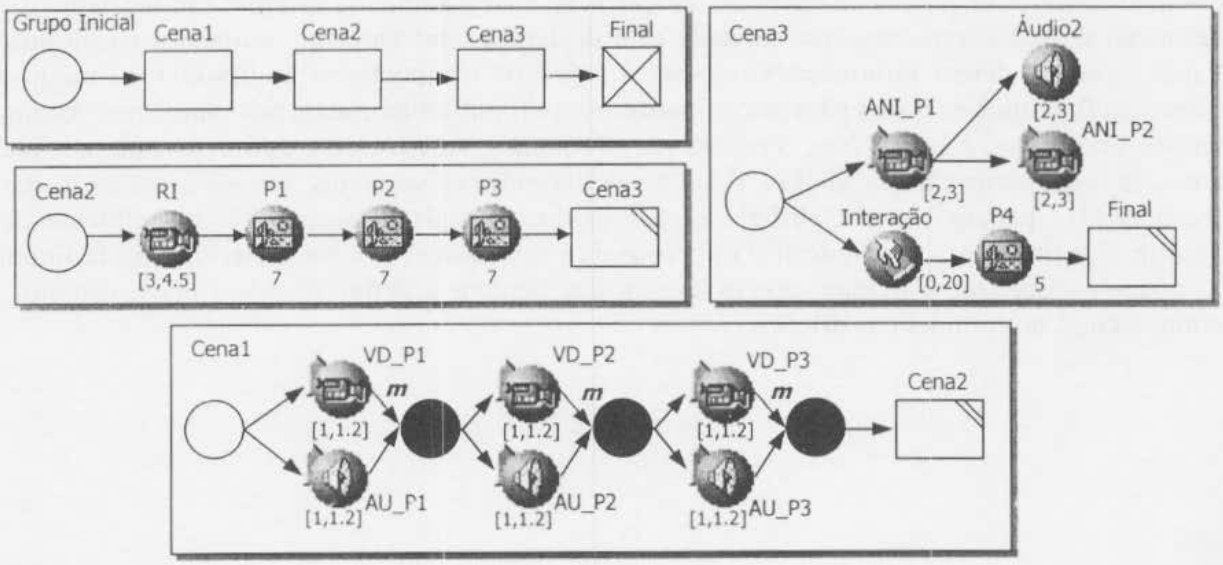


Figura 7 - Representação de um exemplo simples

3. Mapeamento das Aplicações para E-LOTOS

O modelo de autoria proposto, em decorrência de sua alta flexibilidade e expressividade, permite a definição de especificações incoerentes sob o ponto de vista temporal. Exemplos destas incoerências são conflitos na utilização de recursos e a impossibilidade de atingir o final da aplicação considerando todos os caminhos de navegação possíveis.

Em decorrência desta problemática, as especificações descritas por um autor, segundo o modelo apresentado na seção anterior, são traduzidas para uma representação formal. A partir desta representação, métodos de análise podem ser aplicados às mesmas. Como resultado, as incoerências detectadas na validação devem ser apresentadas ao usuário de forma legível, para que ele possa fazer os ajustes necessários. Neste trabalho, aborda-se o processo de geração automática de código E-LOTOS. A validação e apresentação dos resultados são trabalhos que estão em andamento no projeto DAMD [4].

A técnica de descrição formal E-LOTOS (*Enhancements to LOTOS*) [11] é uma versão aprimorada de LOTOS e está em fase final de desenvolvimento junto a ISO, sendo que o padrão para esta nova TDF está previsto para o início de 1999. A principal inovação da linguagem é a incorporação da noção de tempo quantitativo, tornando possível a definição de instantes nos quais ações ou eventos poderão ocorrer. Esta é uma característica fundamental para a representação de aplicações multimídia interativas que, como já mencionado, são descritas de modo geral como um conjunto de relações e restrições temporais entre seus elementos. Além das características mencionadas acima, outros fatores foram determinantes para a sua adoção: possibilidade de modularizar as especificações, de definir e tratar exceções e de representar mais facilmente tipos abstratos de dados.

Com base nas características e funcionalidades da TDF escolhida e considerando o modelo de autoria oferecido ao usuário, estudou-se como representar as aplicações multimídia interativas de maneira sistemática em E-LOTOS. A solução encontrada aproveita a estrutura hierárquica provida pelo modelo de autoria ao considerar suas entidades essenciais: aplicação, grupos, cenas, componentes multimídia e restrições. Estas entidades são descritas como processos que evoluem de acordo com as relações de sincronização previamente estabelecidas entre elas. A forma de representar formalmente as aplicações, apresentada a seguir, é baseada na abordagem proposta em [12].

3.1. Estruturação do Processo Raiz e Declaração dos Dados

A estruturação da aplicação se dá a partir da instanciação do processo raiz. A figura 8 apresenta o esqueleto, em E-LOTOS, da aplicação apresentada anteriormente na figura 7. Em (1), é possível observar os pontos de interação da aplicação com o ambiente externo. Em (2), tem-se a definição de seu comportamento, iniciada pela declaração das variáveis utilizadas na mesma. Estas variáveis detém informações relevantes sobre os componentes multimídia. O módulo *classes* define tipos de dados para os componentes multimídia suportados pelo ambiente. Assim, tem-se tipos como *BitmapClass*, *StreamClass*, *SwitchButtonClass*, cuja definição está baseada em suas respectivas classes MHEG-5. Uma vez definidas as variáveis, tem-se a iniciação das mesmas (3), quando lhe são atribuídos seus valores iniciais, provenientes do ambiente de autoria. Por fim, o processo inicial (GrupoInicial) é instanciado (4). Na parte inferior da figura 8, pode-se visualizar o módulo *processos* (5), que conterá a definição dos grupos, cenários, componentes multimídia e restrições.

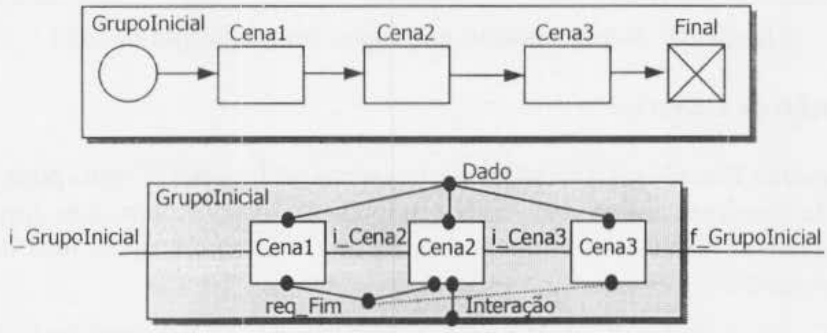
```
specification Aplicação
  import processos, classes, midias
  [i_Aplicação,f_Aplicação,Interação, Dado:class]                    (1)
  behavior
  local var
    RI:StreamClass,d1:Time,d2:Time,                                  (2)
    P1:BitmapClass,dP1:Time,
    P2:BitmapClass,dP2:Time,
    P3:BitmapClass,dP3:Time
    ...
  init
    ...
    ?P1:=((Teste,6),                                                 (3)
          content⇒(content_reference⇒http://www.inf.ufrgs.br/test.gif),
          original_box_size⇒(1000,750),
          original_position⇒(50,50))
    ...
  in
    GrupoInicial[i_Aplicação,f_Aplicação,Interação,Dado]             (4)
      (...,RI,d1,d2,P1,dP1,P2,dP2,P3,dP3,...)
endspec

module processos is                                                  (5)
...
endmod
```

Figura 8 - Esqueleto da estrutura principal de uma aplicação em E-LOTOS

3.2. Representação de Grupos

Grupos são formados por cenários, outros grupos e até mesmo componentes multimídia (componentes compartilhados). Na figura 9 é apresentada a definição do grupo raiz (GrupoInicial), instanciado na figura anterior. Em (1), pode-se observar a utilização do operador de ocultação. Alguns eventos como o início de Cena2 (i_Cena2) e Cena3 (i_Cena3) não são visíveis externamente ao processo. Sua utilidade é permitir a sincronização entre os grupos e cenários envolvidos, modelada com a utilização do operador de sincronização *par* (2). Exemplificando, o início de Cena2 está associado ao evento de final de Cena1 (i_Cena2) (3 e 4). Da mesma forma, o início de Cena3 está sincronizado com o final de Cena2 (i_Cena3) (4 e 5).



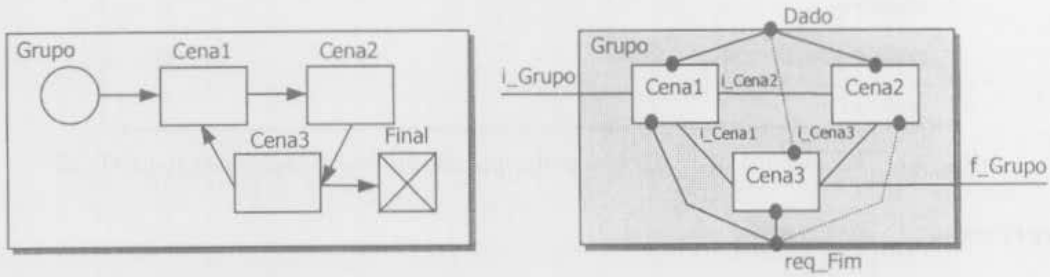

```
process GrupoInicial[i_GrupoInicial,f_GrupoInicial,Interação,Dado]
  (...,RI:StreamClass,d1:Time,d2:Time,P1:BitmapClass,
   dP1:Time, P2:BitmapClass,dP2:Time,P3:BitmapClass,
   dP3:Time,...):exit is
  hide i_Cena,i_Cena2,i_Cena3, req_Fim in                                        (1)
    i_GrupoInicial;
    par i_Cena2#2,i_Cena3#2                                                      (2)
      [i_Cena2]→Cena1[i_Cena,i_Cena2,Dado,req_Fim](...)                          (3)
      [i_Cena2,i_Cena3]→Cena2[i_Cena2,i_Cena3,Interação,Dado,req_Fim](RI,d1RI,d2RI,P1,dP1,P2,dP2,P3,dP3)  (4)
      [i_Cena3]→Cena3[i_Cena3,f_GrupoInicial,Dado,req_Fim](...)                  (5)
    endpar
    [>req_Fim;exit                                                               (6)
  endhide
endproc
```

Figura 9 - Modelagem de GrupoInicial em E-LOTOS

Em (6), observa-se a utilização do evento req_Fim. Ele é utilizado em conjunto com o operador de desabilitação para modelar a finalização da aplicação. Ao ser gerado (por uma transição para o final da aplicação), os processos que modelam grupos, cenários, componentes e restrições são terminados com sucesso.

O grupo no exemplo anterior modela uma estrutura linear. A figura 10 ilustra um grupo hipotético composto de três cenários (Cena1, 2 e 3) estruturados em forma de rede. Como pode ser observado, Cena1 pode ser ativada por dois eventos distintos (1): i_Cena, quando o grupo é iniciado pela primeira vez ou i_Cena1, gerado por Cena3. O efeito da existência de dois pontos de entrada é que, na definição do cenário, ele será iniciado por *i_Cena [] i_Cena1*, indicando que evoluirá quando qualquer um dos eventos ocorrer.

Do mesmo modo, Cena3 apresenta dois pontos de saída (2): o primeiro (f_Grupo) encerra a execução do grupo e o segundo (i_Cena1), como já foi mencionado, habilita a reapresentação da Cena1. Neste caso, também é impossível estabelecer *a priori* como se dará a evolução da aplicação.




```
process Grupo[i_Grupo,f_Grupo,Dado,req_Fim]:exit is
  ...
  hide i_Cena,i_Cena1,i_Cena2,i_Cena3 in
    i_Grupo;
    par i_Cena1#2,i_Cena2#2,i_Cena3#2
      [i_Cena1,i_Cena2 ]->Cena1[i_Cena,i_Cena1,i_Cena2,Dado, req_Fim](...)        (1)
      [i_Cena2,i_Cena3 ]->Cena2[i_Cena2,i_Cena3,Dado, req_Fim](...)
      [i_Cena3,i_Cena1,i_Cena]->Cena3[i_Cena3,f_Grupo,i_Cena1,Dado, req_Fim](...)  (2)
    endpar
    [> req_Fim;exit
  endhide
endproc
```

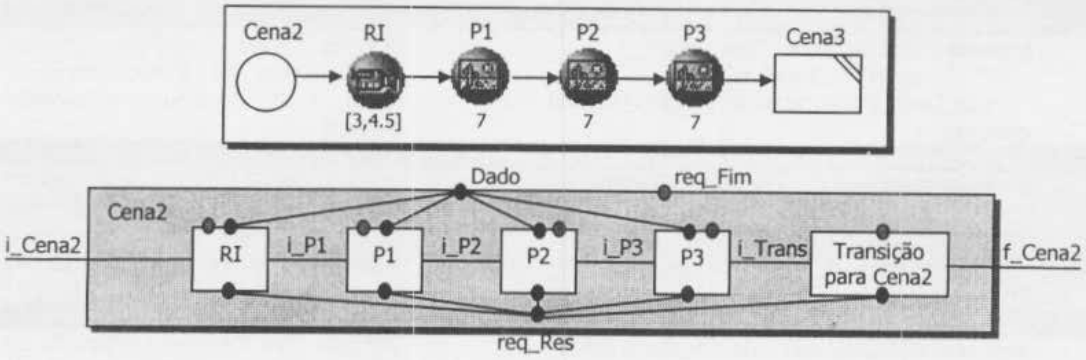
Figura 10 - Representação de grupos com estrutura de rede

3.3. Representação de Cenários

A figura 11 apresenta Cena2, instanciado anteriormente na figura 9. Como pode ser observado, a modelagem de cenários difere em vários pontos da modelagem dos grupos. Uma das diferenças é que não se instanciam mais processos de cenários e grupos, mas de componentes multimídia e restrições.

Outra diferença importante é a utilização do operador *loop* (1) na sua representação. Ele aparece na definição de todos os cenários, permitindo modelar que sua apresentação ocorra mais de uma vez durante a execução da aplicação (estrutura em rede). Assim, o operador de desabilitação (evento req_Fim) também aparece na modelagem dos cenários (3). Na ocorrência deste evento, o processo que modela o cenário é encerrado com sucesso (*exit*). A sincronização entre os componentes multimídia e as restrições é realizada de forma análoga ao que ocorre na sincronização entre grupos e cenas (2).

Na representação de cenários aparece também o evento req_Res. Antes de ocorrer uma transição para outro cenário, é necessário reiniciar o estado de todos os componentes do cenário que está sendo apresentado. Eles passam a aguardar pela solicitação de reapresentação deste cenário, quando então estarão aptos a evoluir novamente.




```
process Cena2 [i_Cena2,f_Cena2,Dado,req_Fim]
  (RI:StreamClass,d1:Time,d2:Time,P1:BitmapClass,dP1:Time,
   P2:BitmapClass,dP2:Time,P3:BitmapClass,dP3:Time)
  hide i_Cena,i_P1,i_P2,i_P3,req_Res in
    loop forever in                                              (1)
      i_Cena2;
      par i_P1#2,i_P2#2,i_P3#2, i_Trans#2                        (2)
        [i_P1]→RI [i_Cena, i_P1,Dado,req_Fim,req_Res] (RI,d1,d2)
        [i_P1,i_P2]→P1[i_P1,i_P2,Dado,req_Fim,req_Res](P1,dP1))
        [i_P2,i_P3]→P2[i_P2, i_P3,Dado,req_Fim,req_Res](P2,dP2))
        [i_P3,i_Trans]→P3[i_P3,i_Trans ,Dado,req_Fim,req_Res](P3,dP3))
        [i_Trans]→Transição[i_Trans,f_Cena2,req_Fim,req_Res]()
      endpar
    endloop
    [> req_Fim;exit                                              (3)
  endhide
endproc
```

Figura 11 - Representação de cenários

Assim, o código E-LOTOS que modela uma transição para outro cenário é composto por três eventos: i_Trans, que denota a ocorrência da transição, req_Res, que provoca o reinício das mídias do cenário correspondente e f_Cena, indicando o final da apresentação do cenário (vide figura 12a). Como a transição também é um processo que está sempre em execução, ela é desabilitada pela ocorrência do evento req_Fim. Na transição para o final da aplicação, por sua vez, o evento req_Res é substituído pelo evento req_Fim (figura 12b).

```
process Transição [i_Trans,f_Cena,req_Fim,req_Res]:exit is
  loop forever in
    i_Trans;req_Res;f_Cena
  endloop
  [> req_Fim;exit
endproc
```

(a) Transição de Cenário

```
process Transição [i_Trans,f_Cena,req_Fim]:exit is
  i_Trans,req_Fim;f_Cena;exit
endproc
```

(b) Transição para o final da aplicação

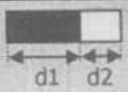
Figura 12 - Representação das transições

3.4. Representação de Componentes Multimídia e Restrições

Objetos básicos ou monolíticos foram definidos por [7] e modelam a apresentação de componentes multimídia simples. Estes componentes são definidos pela ocorrência de eventos síncronos (início e fim) e assíncronos (interação do usuário). Diversas combinações destes eventos podem ser formuladas, mas apenas oito são apontadas como relevantes na definição de cenários multimídia interativos.

Este trabalho utiliza três destas combinações (vide tabela 1). O quarto objeto apresentado nesta tabela (pSsSe-*Synchronous start Synchronous end*) não é apresentado em [7]. Ele permite modelar componentes multimídia dependentes do tempo, com durações mínima e máxima de apresentação. Na definição dos processos, a porta Dado foi utilizada para representar a apresentação do componente.

Tabela 1 - Objetos monolíticos utilizados para representar componentes multimídia

Objeto	Código E-LOTOS	Descrição
	Synchronous start Synchronous end	
	**process** pSsSe[Início,Fim,Dado:class] (Mídia:class,d:time)**:exit is** Início;Dado(!Mídia);**wait**(d);Fim@t[t=0];**exit** **endproc**	Utilizado para modelar mídias independentes do tempo (imagem, texto) com duração previamente definida.
	Synchronous start Asynchronous maximum end	
	**process** pSsAme[Início,Fim,Usuário,Dado:class] (Mídia:class,d1,d2:time)**:exit is** Início; Dado(!Mídia);**wait**(d1) ; (Usuário@t[t<=d2][]**wait**(d2);**exit**);Fim@t[t=0];**exit** **endproc**	Utilizado para modelar a interação do usuário; mesmo que a interação não ocorra no intervalo [d1,d2] o processo é terminado após o tempo máximo (d2).
	Synchronous start Asynchronous end	
	**process** pSsAe[Início,Fim,Usuário,Dado:class] (Mídia:class,d:time)**:exit is** Início;Dado(!Mídia);**wait**(d);Usuário;Fim@t[t=0];**exit** **endproc**	Utilizado para modelar a interação do usuário, sem um tempo máximo de espera definido; o processo só termina após a interação.
	Synchronous start Synchronous maximum end	
	**process** pSsSme[Início,Fim,Dado:class] (Mídia:class,d1,d2:time)**:exit is** Início;Dado(!Mídia);**wait**(d1);Fim@t[t≤d2];**exit** **endproc**	Utilizado para modelar mídias dependentes do tempo como vídeo e áudio que possuem uma duração mínima e máxima.

A figura 13 ilustra a representação do componente P2, presente na definição de Cena2, na figura 11. O evento req_Fim (3) pode novamente ser observado, porque os componentes estão sempre sendo executados (1); se há um laço na definição do cenário, alguns componentes podem ser executados mais de uma vez durante a execução da cena. Ainda nesta figura, é possível observar o efeito da ocorrência do evento req_Res (2): reinício do componente ao estado inicial de apresentação.

```
process P2 [i_P2,f_P2, Dado,req_Fim,req_Res]:exit is
  (P2:BitmapClass,dP2:Time)
  loop forever in                                    (1)
    pSsSe[i_P2, f_P2, Dado] (P2,dP2)
    [>req_Res                                        (2)
  endloop
  [> req_Fim;exit                                    (3)
endproc
```

Figura 13 - Representação de componentes multimídia simples

O modelo de autoria provê a definição de três tipos restrições temporais distintas: *WaitMaster*, *WaitLatest* e *WaitEarliest*. Sua representação em E-LOTOS controla o final da apresentação dos componentes multimídia que convergem para o ponto de sincronização em questão. As restrições, ao contrário dos componentes multimídia, não são implementadas em bibliotecas, pois o comportamento das mesmas é diferente dependendo do número de componentes que convergem para o ponto de sincronização. A tabela 2 apresenta o código E-LOTOS para estes três tipos de restrição.

Tabela 2 - Restrições descritas em E-LOTOS

WaitMaster

```
process WaitMaster[f_A,f_B,f_M,f_Restrição,req_Fim,ctl_Master,req_Res]:exit is
  loop forever in
    (f_A|||f_B)[>f_M;ctl_Master;f_Restrição@t[t=0]
    [>req_Res
  endloop
  [>req_Fim;exit
endproc
```

WaitLatest

```
process WaitLatest[f_A,f_B,f_C,f_Restrição,req_Fim,req_Res]:exit is
  loop forever in
    (f_A|||f_B|||f_C);f_Restrição @t[t=0]
    [>req_Res
  endloop
  [>req_Fim;exit
endproc
```

WaitEarliest

```
process WaitEarliest[f_A,f_B,f_C, f_Restrição,req_Fim,req_Res]:exit is
  loop forever in
    (f_A[]f_B[]f_C);f_Restrição@t[t=0]
    [>req_Res
  endloop
  [>req_Fim;exit
endproc
```

4. O Ambiente de Autoria

O ambiente de criação das aplicações multimídia interativas está dividido basicamente em duas unidades: *repositório de componentes multimídia* e *área de especificação*. A qualquer momento o usuário pode adicionar componentes ao repositório. Isto é feito a partir da localização de um componente multimídia local ou da criação de uma referência a um componente remoto. A figura 14 apresenta as duas janelas que permitem, respectivamente, incorporar novos componentes (*New Media*) à aplicação e manipular (*Medias Palette*) os já existentes.

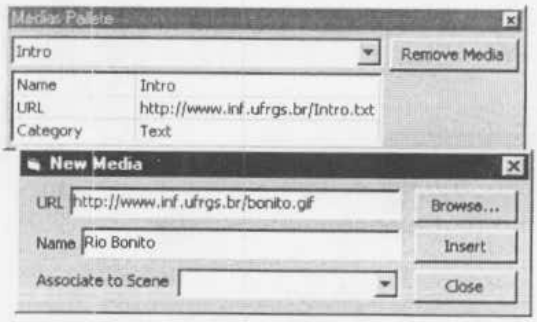


Figura 14 - Gerenciamento de componentes multimídia

A área de especificação, por sua vez, constitui-se dos diversos cenários e grupos que compõem a aplicação. Cada cenário é representado por duas visões distintas: a temporal e a espacial. A visão temporal do cenário permite ao usuário adicionar ícones e pontos de sincronização, bem como estabelecer seu relacionamento com a utilização de arcos. Os componentes visíveis, utilizados na sincronização temporal, podem ser ajustados a partir da visão espacial do cenário.

A figura 15 apresenta a funcionalidade básica do ambiente de autoria. Na barra superior pode-se visualizar as opções disponíveis ao usuário para especificar sua aplicação (1). Pode-se verificar ainda a presença de duas janelas de apoio: hierarquia da especificação *(Specification Hierarchy)* (2) e palheta de ícones *(Icons Palette)* (3).

A hierarquia da especificação provê ao usuário a visão geral da aplicação apresentando todos os cenários e grupos na forma de uma árvore, provendo uma visão clara do relacionamento entre os

mesmos. Neste caso, a aplicação modelada foi o exemplo apresentado na seção 2.4 (figura 7) e conta, portanto, com três cenários: Cena1, Cena2 e Cena3 (4).

A palheta de ícones viabiliza a visualização e edição das propriedades de cada um dos ícones presentes nos diversos cenários da aplicação. Na mesma figura, o ícone de *bitmap* (P1) do cenário Cena2 está selecionado e as suas propriedades específicas são apresentadas na janela mencionada (3). Os ícones que possuem um componente multimídia (áudio, texto, imagem, vídeo) associado apresentam uma propriedade denominada *media*. Esta propriedade precisa ser preenchida com um componente previamente existente no repositório. Neste exemplo, o ícone P1 está associado ao componente Rio Bonito, que aparece sendo adicionado ao repositório na figura anterior.

Ainda na figura 15 verifica-se a especificação do cenário Cena2 (5). Este é composto de um vídeo (RI), seguido seqüencialmente pela apresentação de três imagens (P1,P2 e P3). Ao final da apresentação da última imagem, ocorre uma transição para Cena3. Estas informações são apresentadas pela visão temporal. Ao mesmo tempo, tem-se a visão espacial considerando o componente P1 (6). Nesta visão aparece apenas o próprio componente, já que não existem outros definidos para serem apresentados simultaneamente a ele, nem componentes compartilhados com o cenário a que pertence. O componente visível pode ser movido ou redimensionado, sendo que suas propriedades relativas às coordenadas e dimensões são atualizadas automaticamente na palheta de ícones.

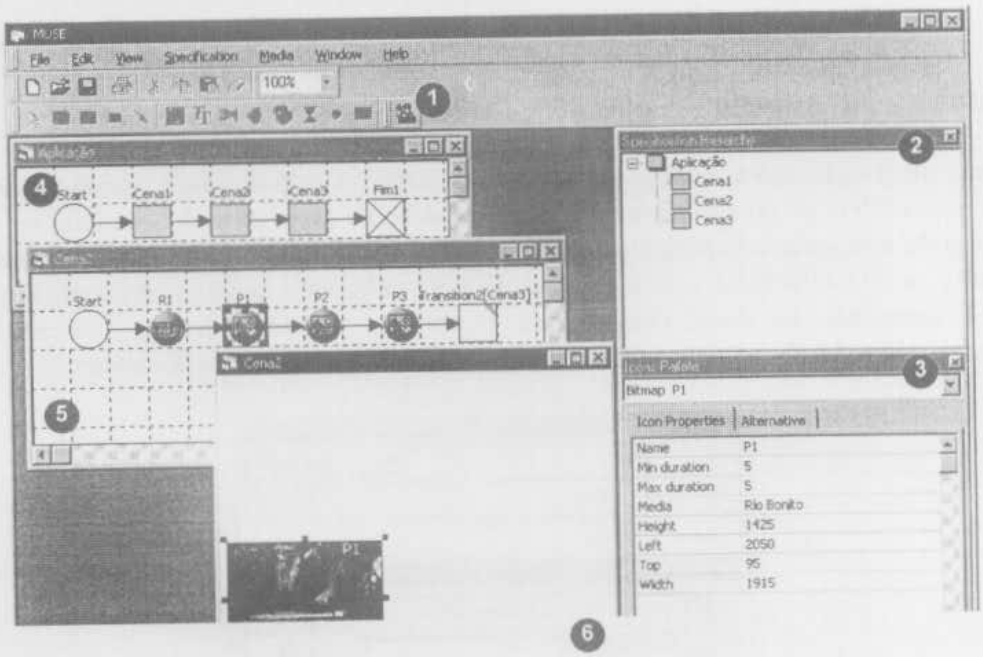


Figura 15 - Interface gráfica do ambiente

Componentes multimídia dependentes do tempo como o vídeo podem ser divididos em segmentos menores, permitindo a sincronização de outros elementos com pontos bem específicos dos mesmos. O ambiente provê mecanismos que facilitam o processo de fragmentação destes componentes. A figura 16 apresenta a janela que disponibiliza esta funcionalidade, que pode ser feita de duas maneiras: a cada *n* quadros ou de acordo com pontos de referência. Abaixo, optou-se pela segunda maneira, sendo criados quatro ícones que representarão quatro partes distintas do vídeo (quadros 1 a 44, 45 a 89, 90 a 134 e 135 a 180).

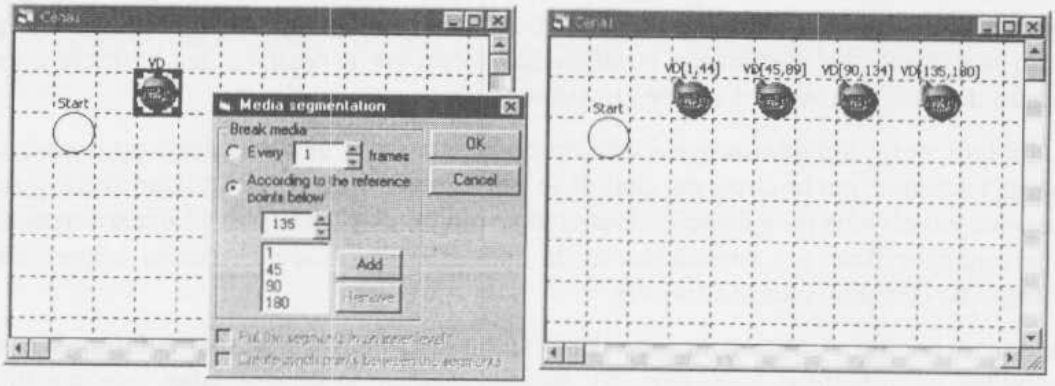


Figura 16 - Segmentação de componentes multimídia

O ambiente permite o reuso de cenários e grupos que pode ser realizado a partir da localização do grupo ou cenário a ser recuperado e reutilizado na nova aplicação. A figura 17 apresenta a reutilização de Cena2, apresentada anteriormente, na definição de uma nova aplicação. Torna-se necessário apenas redefinir a transição, para onde a aplicação deverá evoluir após sua apresentação. Por fim, vale ressaltar ainda a funcionalidade de geração automática de código E-LOTOS das aplicações. Esta é obtida através da opção especial de salvamento *Save As E-LOTOS.*

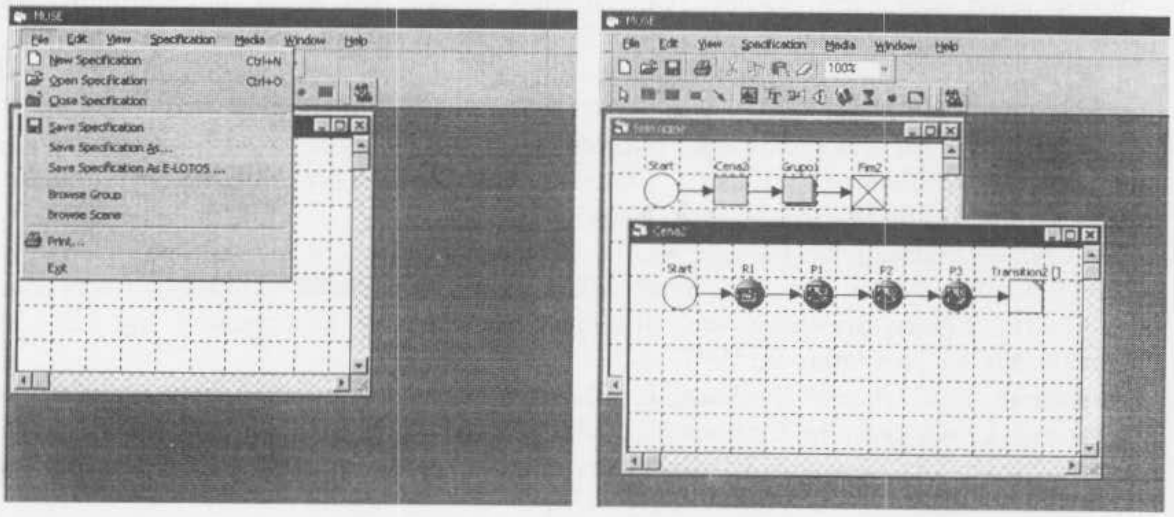


Figura 17 - Reuso de cenários e grupos

5. Conclusões e Trabalhos Futuros

Este trabalho apresentou inicialmente a proposta de um novo modelo para a especificação de aplicações multimídia interativas que leva em consideração restrições temporais de apresentação dos componentes que as compõem. Além disso, apresentou-se mecanismos de mapeamento deste modelo para a linguagem E-LOTOS e o ambiente desenvolvido. A maior contribuição deste trabalho é, portanto, a construção de um ambiente de especificação destinado ao usuário, oferecendo-lhe um alto poder de expressão e disponibilizando a representação formal de suas especificações com fins de análise da consistência temporal da aplicação.

O modelo proposto neste trabalho distingue propositadamente os conceitos de estruturação lógica e sincronização temporal. A estrutura lógica das aplicações possibilita aos projetistas organizar estas aplicações em capítulos, seções ou em qualquer outra unidade. Assim, a aplicação é representada de forma modular, o que contribui para diminuir a complexidade dos cenários e para evitar o problema da explosão de estados.

Com relação ao mapeamento das especificações para E-LOTOS, vale salientar a importância da solução proposta neste trabalho para a geração de código de forma automática. Possivelmente, o

código gerado não seja o mais elegante em todos os casos. Importante, contudo, é que a forma de representação adotada, além de sistemática, oferece os subsídios necessários para a realização dos passos de verificação e validação, ao prover especificações bastante completas sob o ponto de vista dos eventos e ações tratadas.

A continuidade deste trabalho prevê a criação de mecanismos que permitam ao usuário definir no próprio ambiente parâmetros de qualidade de serviço, que serão utilizados na busca dos componentes envolvidos na aplicação. Pretende-se ainda, disponibilizar ao usuário mecanismos que o permitam indicar o comportamento da aplicação diante de variações imprevisíveis na apresentação de componentes multimídia.

Vale ressaltar, por fim, que a utilização deste ambiente integrado às outras ferramentas sendo desenvolvidas no projeto possibilitará ao usuário realizar todas as etapas que constituem o processo de desenvolvimento de aplicações multimídia interativas: especificação, verificação e apresentação. A facilidade do modelo de autoria apresentado ao usuário e a utilização de uma técnica de descrição formal para validar as aplicações por ele criadas tornam o ambiente atrativo e de fácil utilização sem perder a alta expressividade e, principalmente, capaz de apresentar resultados confiáveis sobre a coerência das aplicações criadas.

Referências Bibliográficas

[1] G. Blakowski and R. Steinmetz. *A Media Synchronization Survey: Reference Model, Specification, and Case Studies*. IEEE Journal on Selected Areas in Communications, 14(1):5-35, January, 1996.

[2] ISO/IEC DIS 13522-5. *Information Technology - Coding of Multimedia and Hypermedia Information, Part 5: Support for Base-Level Interactive Applications*, 1995.

[3] L. P. Gaspary, M. J. Almeida e R. Willrich. *MUSE: Um Ambiente para a Concepção de Aplicações Multimídia JAVA*. IV Simpósio Brasileiro sobre Sistemas Multimídia e Hipermídia, Rio de Janeiro. Maio de 1998.

[4] W.Lopes de Souza, J. M. Farines, M.J.B. Janilce, L.F. Pires, M.S. Camargo, R. Willrich, R.J.C. Costa. *Design de Aplicacoes Multimidia Distribuidas (DAMD)*. Anais do II Seminario Franco-Brasileiro em Sistemas Informaticos Distribuídos. pp. 271-281, Fortaleza (CE). Novembro de 1997.

[5] Bulterman, D. C. A. and Hardman, L. *Multimedia Authoring Tools: State of the Art and Research Challenges*. In Computer Science Today: Recent Trends and Developments, edited by Jan van Leeuwen, Lecture Notes in Computer Science 1000, 575-591, Springer 1995.

[6] L. Soares e R. Rodrigues. *Autoria e Formatação Estruturada de Documentos Hipermídia com Restrições Temporais*. III Workshop sobre Sistemas Multimídia e Hipermídia, São Carlos. Maio de 1997.

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

[8] P. Sénac, R. Willrich, P. de Saqui-Sannes. *Hierarchical Time Stream Petri Nets: A Model for Hypermedia Systems*. In Application and Theory of Petri Nets, 1995.

[9] Willrich, R.; de Saqui-Sannes, P. *Concepção Formal de Aplicações Multimídia Java*. Anais do XV Simpósio Brasileiro de Redes de Computadores, São Carlos. Maio de 1996.

[10] P. Sénac, M. Diaz and P. de Saqui-Sannes. *Toward a formal specification of multimedia synchronization scenarios*. Ann. Telécommun. no. 49, pp 297-314.

[11] ISO/IEC JTC1/SC21/WG7. *Enhancements to LOTOS*. Revised Working Drafts on Enhancements to LOTOS (V4), Project WI 1.21.20.2.3, January, 1997.

[12] J. P. Courtiat and R.C. de Oliveira. *Proving Temporal Consistency in a New Multimedia Synchronization Model*. ACM Multimedia, Boston, 1996.