Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Ferramenta MAE: Autoria Multimídia e Integração com Banco de Dados

Fernando Genta dos Santos

São Carlos-SP
Fevereiro/2003

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária/UFSCar

S237fm

Santos, Fernando Genta dos.
Ferramenta MAE: autoria multimídia e integração com banco de dados / Fernando Genta dos Santos. -- São Carlos : UFSCar, 2004.
91 p.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2003.

1. Software. 2. Autoria multimídia. 3. Base de dados multimídia. 4. SMIL. I. Título.

CDD: 005.3 (20ª)

Aos meus pais e meus avós,
pelo exemplo de amor,
trabalho e dedicação.

Agradecimentos

À minha orientadora Marina, não só pela valiosa orientação, mas também pelo apoio, paciência e compreensão;

Aos meus amigos de todas horas, Linder e Douglas, pela solidariedade e apoio;

Aos amigos do PPG-CC, especialmente ao Grupo de Banco de Dados, pelo convívio alegre e pela troca de conhecimento;

À Capes, que me auxiliou a desenvolver este projeto.

E a todos que, de alguma forma, contribuíram para essa minha vitória,

Meu muitíssimo obrigado!

Resumo

Aplicações multimídia são muito utilizadas em ambientes computacionais, pois oferecem uma quantidade muito grande de recursos e permitem uma grande interatividade com o usuário. Com as vantagens da utilização de aplicações multimídia, também vieram recursos mais complexos e a necessidade de um grande conhecimento de programação por parte do autor para construir tais aplicações. Para facilitar o trabalho de criação de aplicações multimídia, foram desenvolvidas várias ferramentas que facilitam a autoria. Este trabalho apresenta uma proposta de uma ferramenta de autoria de aplicações multimídia que visa oferecer recursos adicionais àqueles tradicionalmente encontrados nas ferramentas de autoria existentes. Essa ferramenta pode ser utilizada através da Web e também está associada a um banco de dados, onde estão disponíveis várias mídias e aplicações que podem ser recuperadas através de consultas semânticas e visam auxiliar o autor na criação da sua aplicação multimídia.

Abstract

Multimedia applications are heavily used in computing environments, since they provide a lot of resources and interaction with the user. Taking into account the advantages provided by multimedia applications, resources became more complex requiring good programming knowledge to develop them. To facilitate multimedia applications developing, many tools have been created. This work describes a proposal of a multimedia application authoring tool which aims to provide additional resources to those traditionally found in the existing authoring tools. This tool can be used on the Internet and has an associated database, where there are medias and applications available which can be selected by semantic queries in order to assist the author in the design of his own multimedia application.

SUMÁRIO

ÍNDICE DE FIGURAS

1. INTRODUÇÃO

Quando se fala em ambientes multimídia, é importante fazer uma classificação dos tipos de mídia. Mídia se refere ao tipo de informação, como dados alfanuméricos, imagens, áudio e vídeo. Existem muitas maneiras de classificar mídias. Guojun [Guojun, 1996] classifica as mídias pelo fato de existir ou não uma dimensão de tempo para a mídia. Nessa classificação, existem duas classes de mídias: estáticas e dinâmicas.

Mídias estáticas não têm uma dimensão de tempo, e seus conteúdos e significados não são dependentes do tempo de apresentação. Mídias estáticas incluem dados alfanuméricos, gráficos, e imagens.

Mídias dinâmicas têm uma dimensão de tempo, e seus significados e exatidão dependem da taxa com que são apresentadas. Mídias dinâmicas incluem animação, áudio e vídeo.

Do ponto de vista lingüístico, qualquer sistema capaz de manipular mais de um tipo de mídia deve ser chamado de sistema multimídia. Por essa definição, um sistema computacional capaz de manipular dados alfanuméricos e gráficos pode ser chamado de um sistema multimídia. Guonjun define sistemas multimídia como aqueles capazes de manipular ao menos um tipo de mídia contínua no formato digital assim como mídia estática, ou seja, informação multimídia é uma combinação de múltiplos tipos de mídias com ao menos um tipo contínuo.

Seres humanos têm múltiplos sentidos. A motivação para o uso de dados multimídia em sistemas computacionais é aprimorar a transferência de informação através do envolvimento de dois ou mais sentidos do participante. O objetivo da computação e comunicação multimídia é simular a comunicação humana e assistir os seres humanos na organização e gerenciamento de grandes quantidades de informação em vários tipos de mídias. A comunicação humana face-a-face usa uma variedade de sentidos. É efetivamente uma comunicação multimídia, envolvendo os sentidos visuais e auditivos. Com o desenvolvimento e uso da tecnologia digital, de processadores rápidos, redes de alta velocidade, dispositivos de armazenamento de grande capacidade, e novos algoritmos de processamento de sinal, os sistemas multimídias estão se tornado cada vez mais comuns e presentes em várias áreas, como medicina, entretenimento e educação. Ferramentas de uso fácil para a criação, manipulação e apresentação de conteúdo multimídia são um importante fator na utilidade de informação multimídia.

As principais ferramentas utilizadas em sistemas multimídia são as ferramentas de autoria e as ferramentas de apresentação. Como ferramentas de apresentação são consideradas aquelas que mostram para o usuário as mídias e as informações das aplicações multimídia. Já as ferramentas de autoria auxiliam o autor na criação das aplicações, pois com as vantagens da utilização de aplicações multimídia, também vieram recursos mais complexos e a necessidade de um grande conhecimento de programação por parte do autor para construir tais aplicações. Para facilitar o trabalho de criação de aplicações multimídia, foram desenvolvidas várias ferramentas que facilitam a autoria.

A principal característica das ferramentas de autoria multimídia é fornecer funcionalidades que permitam a integração de várias mídias de forma a construir uma aplicação que pode ser mostrada através de uma ferramenta de apresentação.

Quando se tratam as aplicações e suas informações como dados, que são utilizados diversas vezes e possivelmente por diversos usuários, a utilização de um SGBD é muito importante para armazenar e recuperar tais dados. Em um banco de dados multimídia é útil manter, não somente as mídias, mas também informações sobre seus conteúdos, usadas na recuperação dessas mídias. Além disso, essas mídias podem possuir informações imprecisas que podem ser úteis no processo de recuperação, permitindo satisfazer melhor os requisitos estabelecidos em uma consulta. A imprecisão nos dados armazenados pode ser melhor explorada através de mecanismos de busca nebulosa, o que permite ampliar o conjunto resposta ao usuário, visando assim melhor atender à sua consulta. As estruturas usadas para o armazenamento e manipulação dessas informações são bastante complexas, e os sistemas gerenciadores de banco de dados orientados a objetos têm sido utilizados, por apresentarem características que facilitam a manipulação de estruturas complexas.

Outro aspecto importante relacionado com a autoria de aplicações multimídia diz respeito à área de utilização da aplicação, ou seja, uma aplicação multimídia normalmente é construída especificamente para determinado domínio. Com isso, certas características, que dizem respeito somente ao domínio no qual a aplicação é utilizada, têm de ser inseridas na aplicação (no momento de sua autoria ou execução) ou nos módulos que a manipulam. Um exemplo diz respeito a aplicações multimídia voltadas para Educação a Distância [Mayes, 1994], [Schank, 1995], que devem possuir vários controles específicos, como o acompanhamento do aluno e a didática utilizada pelo professor. O uso na educação é uma poderosa aplicação de sistemas multimídia. Pessoas absorvem mais conhecimento e de forma

mais rápida quando elas podem ver, ouvir, e trabalhar com novos conceitos, o que torna multimídia um caminho natural para treinamento e educação.

Este trabalho apresenta uma ferramenta de autoria de aplicações multimídia que visa oferecer recursos adicionais àqueles tradicionalmente encontrados nas ferramentas de autoria existentes. São apresentadas as principais características da ferramenta de autoria multimídia MAE (*Multimedia Authoring Environment*) [Figueiredo 2000], para a criação e alteração de aplicações multimídia baseadas no padrão SMIL 1.0 [SMIL 1998]. O enfoque é mostrar as funcionalidades implementadas e as soluções propostas para dar mais suporte ao usuário na criação de uma aplicação multimídia. Essa ferramenta manipula um banco de dados multimídia (BDMm) que mantém aplicações criadas pela ferramenta e interage com um módulo que realiza consultas nebulosas por conteúdo nesse BDMm.

Este trabalho está organizado como segue: no capítulo 2 é apresentada a linguagem SMIL 1.0, utilizada pela ferramenta MAE, e também são comentadas algumas características de SMIL 2.0. No capítulo 3 são apresentadas algumas ferramentas de autoria multimídia, mostrando suas características e forma da utilização. No capítulo 4 é descrita a estrutura de classes do banco de dados multimídia utilizada pela MAE, que armazena as aplicações e informação semântica sobre as mídias que compõem essas aplicações. No capítulo 5 é apresentada a ferramenta MAE, mostrando a sua arquitetura, a estrutura de classes desenvolvida para a representação das aplicações multimídia utilizadas pela ferramenta MAE, a sua interface, seu funcionamento e suas principais características. Também é feita uma comparação entre a ferramenta MAE e outras ferramentas de autoria multimídia. Finalmente, no capítulo 6, são apresentadas as conclusões sobre o trabalho.

2. SMIL

2.1 - SMIL 1.0

SMIL (*Synchronized Multimedia Integration Language*) [SMIL, 1998] é uma linguagem proposta pela organização W3C (World Wide Web Consortium) e tem por objetivo proporcionar uma maneira fácil de se criar apresentações audiovisuais interativas. Uma apresentação SMIL pode ser criada usando um editor de texto simples.

O padrão SMIL foi definido a partir de XML (*Extensible Markup Language*) [XML, 2001]. SMIL permite integrar um conjunto de objetos multimídia independentes numa apresentação multimídia sincronizada. Utilizando SMIL 1.0, através de um conjunto de tags (marcações), um autor pode descrever o comportamento temporal da apresentação, descrever a disposição da apresentação na tela e associar hiperligações a objetos multimídia.

Em um documento SMIL 1.0 é possível especificar a localização de cada mídia que compõe a aplicação e também a sua duração durante a apresentação. Além disso, é possível criar associações entre elas, como por exemplo, o início de um áudio pode estar associado ao final da apresentação de uma imagem ou ainda um vídeo deve começar depois de passados 5 segundos do início de um áudio.

As principais marcações do padrão SMIL 1.0 são: <smil>, <head> e <body> (Figura 1). A marcação <smil> define um documento SMIL e todas as outras marcações no documento são suas dependentes. A marcação <head> contém informações desassociadas do comportamento temporal da apresentação, contendo informações sobre a aparência e disposição espacial do documento através das marcações <layout> e <region>. Além disso, através da marcação <meta> define metainformações sobre o documento.

A marcação <region> define, dentro da aplicação, uma região na qual pode ser definida altura, largura, posição, cor de fundo entre outras características. Cada região pode ter uma ou mais mídias associadas a ela e é através dessa região que se define o tamanho e a posição da mídia na aplicação.

A marcação <body> contém as marcações que estão associadas ao comportamento temporal e relacional do documento e define implicitamente um elemento <seq> (seqüencial). As marcações dependentes de <body> são as que tratam as mídias (<textstream>, <text>, <audio>, <animation>, <video>, e <img>), as que tratam da sincronização (<seq> e <par>) e as que tratam das hiperligações (<anchor> e <a>).

A marcação <seq> define que as mídias nela contidas devem ser apresentadas seqüencialmente, ou seja, uma mídia só começa a ser executada quando a anterior terminar. Já a marcação <par> define que as mídias devem ser apresentadas paralelamente.

Na Figura 1 é apresentado um exemplo de documento SMIL. Pode-se perceber a intenção do padrão SMIL de fornecer uma forma de manipulação, e não de criação, de mídias. As mídias estão armazenadas em arquivos externos ao documento SMIL. Na Figura 2 é mostrada a visualização desse documento SMIL através da ferramenta RealOne Player [Real, 2002].

O documento SMIL é um arquivo texto e nesse exemplo deve estar no mesmo local das mídias, para que a ferramenta de visualização possa apresentá-lo. O atributo *src* indica a mídia que está associada à tag e pode também conter uma string que indique a localização do arquivo; por exemplo, na tag <áudio> ao invés de somente "*audio01.mp3*" poderíamos ter "*c:\midias\audio01.mp3*".

Esse exemplo inicialmente executa um áudio. Terminada a execução do áudio inicia-se a execução paralela de um áudio, uma imagem e um vídeo. Entre as mídias do bloco paralelo (iniciado por <par> e finalizado por </par>), a imagem definida na marcação <img> é a única que tem uma duração explicita de 10 segundos, definida pelo documento através do atributo *dur*. As outras mídias têm o seu tempo normal de duração, definido pelo seu próprio formato. O bloco paralelo só termina quando for encerrada a exibição da mídia que tiver o maior tempo de execução.

```
<smil>
  <head>
    <layout>
      <root-layout width="500" height="300" background-color="black" />
      <region id="video" left="0" top="0" width="325" height="218" />
      <region id="imagem" left="70%" top="10%" width="133" height="88" />
    </layout>
  </head>
  <body>
    <seq>
       <audio src=”audio03.wav” id=”narracao”/>
       <par>
          <audio src="audio01.mp3" id="musica tema"/>
          <img src="img02.jpg" region="imagem" dur="10s"/>
          <video src="video01.mpeg" region="video"/>
       </par>
       <img src=”img01.jpg” region="imagem" dur="15s"/>
    </seq>
  </body>
</smil>
```

Figura 1 - Exemplo de um documento SMIL

Figura 2 - Visualização do documento SMIL

Terminada a execução da ultima mídia do bloco paralelo, inicia-se a exibição de uma imagem e que terá a duração de 15 segundos.

O documento da Figura 1 faz referência a cinco mídias, identificadas por "*audio03.wav", "audio01.mp3", "video01.mpeg", "img02.jpg"* e *"img01.jpg"*. Existem apenas duas regiões definidas nesse documento. É através da associação de uma mídia a uma região que se define a localização espacial da mídia na apresentação. Cada região possui um identificador, indicado no atributo *id*, sendo que é através desse atributo que é feita a associação com a mídia.

No exemplo dado, a mídia identificada pelo arquivo *img02.jpg* está associada à região que tem o identificador *imagem*. Uma região pode ter mais de uma mídia associada a ela, sendo que o autor deve estruturar seu documento para que essas mídias não sejam executadas em um mesmo momento. As mídias de áudio não são associadas a uma região, uma vez que não são visuais.

Atualmente existem várias ferramentas de apresentação multimídia que trabalham com o padrão SMIL 1.0, tais como QuickTime [Quick, 2002], Realplayer [Real, 2002], Grins[Grins, 2002], entre outras.

2.2 - SMIL 2.0

SMIL 2.0 [SMIL, 2001] é a segunda versão da linguagem SMIL e traz vários novos recursos que tornaram a linguagem mais poderosa. No que se refere aos atributos mantidos da versão 1.0, houve algumas alterações na sintaxe, sendo que a alteração mais visível é a retirada do hífen dos nomes de atributos; por exemplo, clipBegin é introduzido no lugar de clip-begin. Apesar dessas alterações na sintaxe, os módulos do SMIL 2.0 não exigem suporte para os antigos atributos do SMIL 1.0 que foram alterados; assim integração de aplicações não é suportada por eles. Ferramentas de apresentação de documentos SMIL devem suportar os antigos nomes de atributos do SMIL 1.0, assim como os novos nomes do SMIL 2.0, para que possam executar tanto arquivos SMIL 1.0 como SMIL 2.0. Ferramentas de apresentação que seguem somente as especificações do SMIL 2.0 não são capazes de entender os documentos SMIL 1.0.

Na versão 2.0 do SMIL foi mantido o formato declarativo XML. Além disso, foi introduzida uma estrutura baseada em módulos, sendo dividida em mais de 40 módulos que podem ser particionados em 10 grupos funcionais. Através dessa estrutura de módulos, aspectos importantes de SMIL 2.0 podem ser integrados em outras linguagens baseadas em XML, sem a necessidade de suporte para toda a especificação SMIL 2.0.

Um dos objetivos do desenvolvimento de SMIL 2.0 é estender as funcionalidades de SMIL 1.0 adicionando, assim, alguns recursos interessantes, tais como suporte para o aumento da interação, aumento da semântica temporal, aumento do controle de conteúdo e layout, e adição de novas primitivas de animação e transição.

2.3 – Considerações finais

Neste capítulo foram apresentados, de uma forma bem geral, alguns conceitos básicos para o entendimento do trabalho aqui apresentado. Foram apresentadas, de maneira bem simples, a estrutura e o funcionamento de um documento SMIL. Maiores informações podem ser obtidas em [SMIL, 1998]. Também foi feita uma pequena introdução ao SMIL 2.0, mas como o trabalho aqui descrito usa exclusivamente SMIL 1.0, não foram tecidas considerações mais detalhadas sobre essa nova versão do SMIL. Outras informações podem ser encontradas em [SMIL, 2001]

3. FERRAMENTAS DE AUTORIA MULTIMÍDIA

As ferramentas de autoria existentes geralmente se concentram no processo de desenvolvimento de uma aplicação multimídia isolada, ou seja, aspectos relacionados com a utilização de mais de uma aplicação, como o reuso de partes de aplicações já desenvolvidas, não são funcionalidades normalmente disponíveis. Com isso, o autor tem um trabalho adicional no momento em que deseja reutilizar partes de uma aplicação, pois é necessário desenvolver novamente parte dessa aplicação, ou fazer uma cópia e modificá-la de forma a atender às novas necessidades. Se o número de aplicações construídas é grande, e/ou se a autoria das aplicações é feita através da cooperação de diversos autores, a reutilização de aplicações se torna necessária e seu gerenciamento tem de ser realizado de forma mais complexa. A utilização de um Sistema Gerenciador de Banco de Dados (SGBD) para armazenar e recuperar as aplicações multimídia se mostra como uma das maneiras mais adequadas para o gerenciamento e reutilização das mesmas. Tratando a aplicação e suas informações como dados que são utilizados diversas vezes e possivelmente por diversos usuários, a utilização de um SGBD é imprescindível. Além disso, o ambiente multimídia deve estar integrado ao SGBD de forma mais transparente possível, permitindo que o usuário acesse as informações sem a necessidade de conhecer linguagens de consulta e outras características internas do SGBD.

A ferramenta MAE, apresentada no capítulo 5, foi desenvolvida de forma a permitir que extensões sejam feitas de uma maneira relativamente simples, isto é, que novos componentes sejam inseridos fornecendo novas funcionalidades, sem acarretar uma grande mudança na interface e de forma que fiquem integrados aos outros módulos da ferramenta.

3.1 Classificação das ferramentas de autoria multimídia

Para iniciar um projeto de desenvolvimento de uma ferramenta de autoria multimídia é necessário conhecer e entender a tecnologia multimídia e suas particularidades. Trata-se de compreender como essa tecnologia está sendo utilizada e quais os principais paradigmas de interface com o usuário existem. Autoria é o processo de criação de aplicações multimídia, enquanto que paradigma de autoria é a metodologia para a autoria de aplicações multimídia Nesse contexto, esses paradigmas podem ser utilizados como forma de classificação das ferramentas. Marshall [Marshall, 2002] classifica os paradigmas em:

- Script Language
  - Usa uma linguagem especial para capacitar interatividades (botão, mouse, etc) e permitir saltos, condições, repetições. Por exemplo, OpenScript na Toolbook
- Slide Show
  - Uma apresentação linear. Por exemplo, PowerPoint
- Hierarchial
  - Organizado em uma estrutura de árvore, visto freqüentemente em menus de guia de aplicações
- Iconic/Flow-control
  - Ícones gráficos e um mapa de fluxo para auxiliar a autoria. Por exemplo, Authoware da Macromedia
- Card/Scripting
  - Estrutura de cartões indexados, bom para hipertexto/hipermídia. Por exemplo, SuperCard e HyperCard
- Cast/Score/Scripting
  - Com uma linguagem script; muitas "trilhas" de sincronização horizontal simultaneamente mostradas em colunas verticais. Por exemplo, Director (usa Lingo como sua linguagem script).

Esses paradigmas fornecem ao usuário uma forma de entender e visualizar o funcionamento e a manipulação das informações que estão sendo adicionadas à aplicação.

Contudo, cada ferramenta possui uma maneira de representar e armazenar essas informações para que possam ser apresentadas posteriormente. Essa maneira de armazenar as informações na maioria das vezes é feita através de um padrão proprietário. Dessa forma, para que uma ferramenta apresente e manipule uma aplicação construída em outra, é necessário entender o padrão que a outra gerou. Uma das maneiras de realizar essa atividade é converter diretamente as instruções de uma ferramenta para outra, porém é necessário que os dois formatos de representação sejam reconhecidos. Outro problema é que, idealmente, seriam necessários conversores entre cada ferramenta. A maneira mais adequada de trocar as informações é através da utilização de um padrão que deve ser conhecido pelas diversas ferramentas, independentemente de seu paradigma.

Um padrão deve ter como propriedades desejadas, ser aberto e conseguir resolver os vários aspectos relacionados com a tecnologia que o trata. Vários padrões multimídia foram especificados, entre os quais pode-se citar: PREMO, MHEG, SMIL 1.0 e SMIL 2.0. PREMO

é um padrão voltado para utilização em aplicações hipermídia, enquanto que MHEG é um padrão orientado a objetos que possui vários subprojetos (MHEG 1 a MHEG 8) voltados para os diversos aspectos da tecnologia multimídia. Já os padrões SMIL 1.0 e SMIL 2.0 são linguagens derivadas de XML, cujo intuito é permitir a integração e sincronização de mídias de diversos formatos para execução e apresentação na World Wide Web. Utilizando SMIL, através de um conjunto de tags (marcações), um autor pode descrever o comportamento temporal da apresentação, descrever a disposição da apresentação na tela e associar hiperligações a objetos multimídia. Em um documento SMIL é possível especificar a localização de cada mídia que compõe a aplicação e também a sua duração durante a apresentação. Além disso, é possível criar associações entre elas, como por exemplo, o início de um áudio pode estar associado ao final da apresentação de uma imagem, ou ainda um vídeo deve começar depois de passados 5 segundos do início de um áudio.

Através da adoção de padrões, a difusão da aplicação pode ser mais facilmente alcançada. Contudo, no que concerne à difusão de uma ferramenta, um aspecto importante diz respeito à sua portabilidade. Como a ferramenta é um software que tem de ser executado a partir de um sistema operacional (SO), sua portabilidade é importante, no sentido de possibilitar que usuários de sistemas diversos possam utilizá-la. Uma das formas mais comuns de implementação de sistemas é através das linguagens C/C++ em sua padronização ANSI, o que permite que a versão básica seja recompilada para os diversos SOs, e aspectos específicos de cada SO podem ser explorados em versões específicas do SO. Contudo, esse tipo de implementação tem de possuir uma gerência complexa, cujo controle de versões passa a ser muito trabalhoso. Nesse contexto, a linguagem Java [Java, 2002] resolve o problema da necessidade de geração de diversas versões para cada sistema operacional. Sua portabilidade permite que somente uma versão possa ser gerada pelo programador sem que seja necessário compilar o sistema para cada tipo de SO. Outro aspecto importante na linguagem Java é sua possibilidade de execução no ambiente da WWW, através de applets. Com isso, além de ser executada em vários SOs, ainda é possível utilizá-la diretamente através de um browser Web, sem que tenhamos de instalar o código binário na máquina destino.

3.2 Ferramentas de autoria

A seguir são apresentadas as características principais das ferramentas de autoria multimídia Visual Class [Tatizana, 1999], Director Shockwave Studio [Director, 2001], GRiNS [Grins, 2002] e Fluition [Fluition, 2002]. Dentre as ferramentas de autoria analisadas, essas são as que mais se aproximam das características de interesse deste projeto, porém

nenhuma delas apresenta facilidades para o reuso de aplicações, e também não estão associadas a um banco de dados.

3.2.1 – Visual Class

Visual Class [Tatizana, 1999] é um software para a criação de projetos multimídia criado através de uma parceria da Escola Politécnica com as empresas CALTECH Informática e MPR Informática, sendo Celso Tatizana o criador do software. Visual Class está disponível para a plataforma Windows (95, 98, NT 4.0). A versão 6.1 é a mais atual da Visual Class, porém aqui será analisada a versão 4.2, pois foi a única a que se teve acesso. A Figura 3 mostra a tela principal da ferramenta, com alguns editores de propriedades para imagens.

Uma aplicação criada através da ferramenta Visual Class pode ser constituída de várias cenas, onde cada cena é uma tela. Durante a criação de uma aplicação, depois de criada uma tela, pode-se inserir outra antes ou depois, criando uma seqüência de cenas. Durante a execução da aplicação, que é feita através da própria ferramenta, o usuário pode avançar ou retroceder as cenas.

A tela inicial da ferramenta é muito simples, consistindo de uma tela branca com uma barra de menus na parte superior. Através dessa barra são inseridas as mídias (imagens, filmes, áudio e texto) que farão parte da aplicação. Na área 1 da Figura 3 é mostrada uma imagem que foi inserida na tela de apresentação da Visual Class e através dela (duplo clique) é disparado o editor de propriedades (parte 2 da figura). Através desse editor é possível associar sons, funções e rótulos a ela. Pode-se também criar efeitos na imagem; a parte 3 da figura mostra o editor de efeitos onde é possível colocar molduras na imagem. Vários tipos de efeitos de apresentação podem ser criados sobre a imagem, como, por exemplo, a imagem começar a aparecer da esquerda para a direita ou de cima para baixo. A parte 4 da figura mostra a paleta de cores para a configuração da moldura.

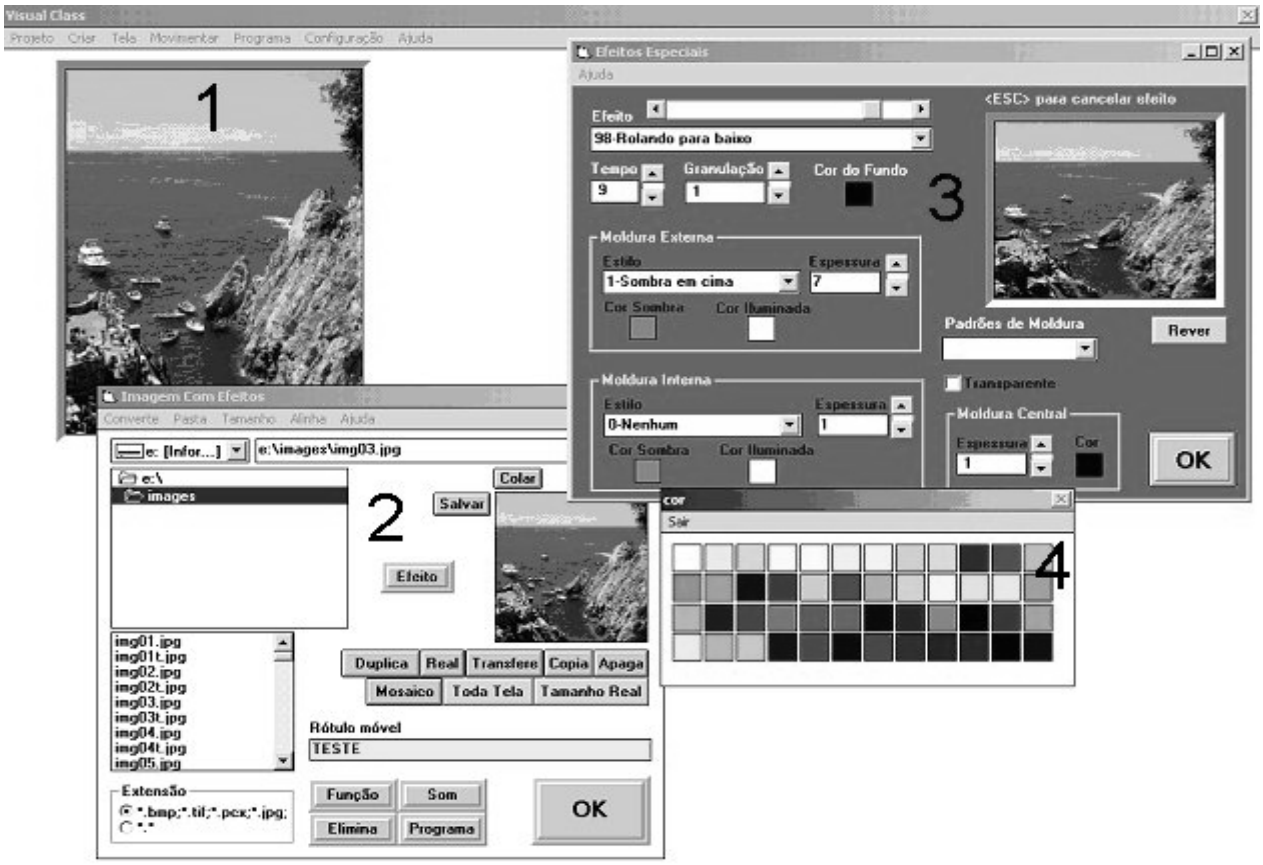


Figura 3 - Ferramenta Visual Class com alguns editores de propriedades

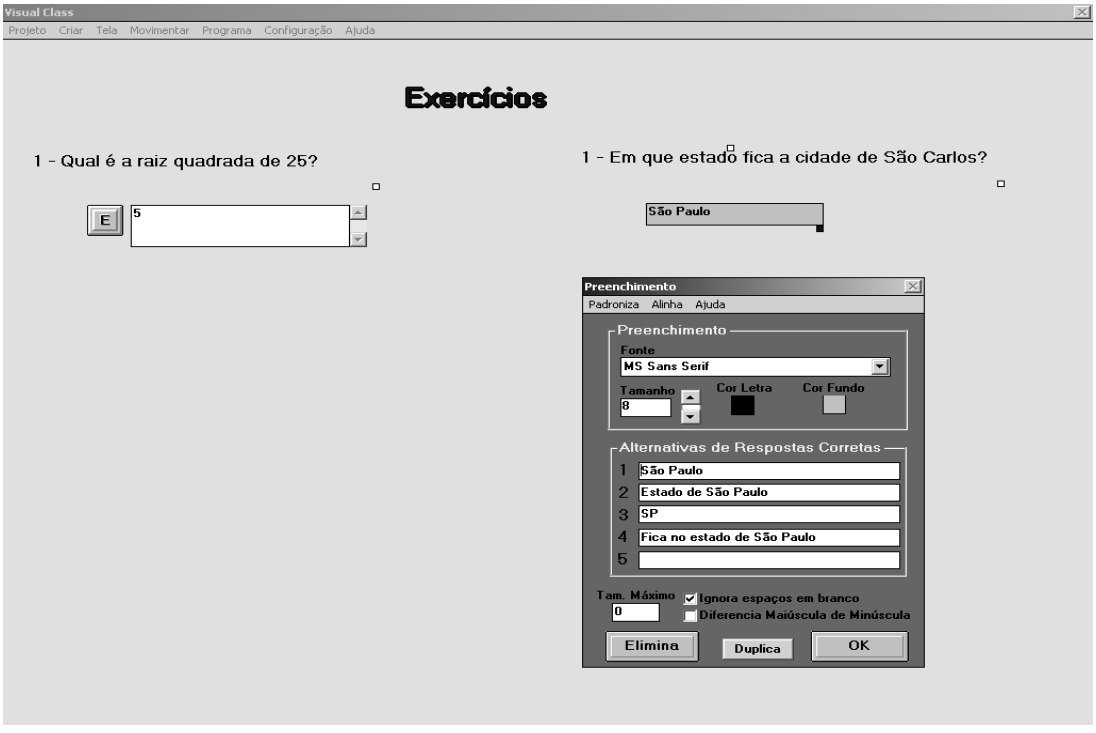


Figura 4 - Criação de exercícios na ferramenta Visual Class

Essa ferramenta possui algumas características especificas para a criação de material didático, como, por exemplo, um ambiente para facilitar a criação de testes e questões. Na Figura 4 é apresentada a forma como são criados e editados os testes e as questões, sendo que basta apenas indicar qual é a resposta correta e durante a execução a ferramenta verifica se o usuário responde corretamente. A verificação é feita quando o usuário deixa a cena que contém os exercícios. Na Figura 5 é mostrada a execução dos exercícios criados.

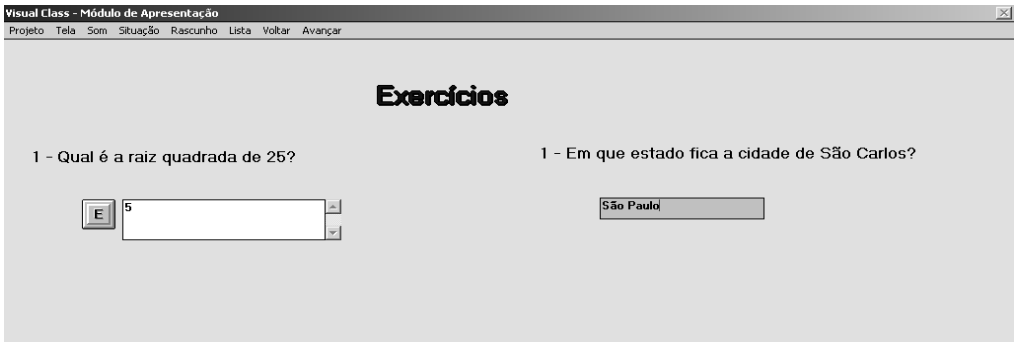


Figura 5 - Apresentação dos exercícios criados no Visual Class

3.2.2 – Director Shockwave Studio

Director Shockwave Studio [Director, 2001] é uma ferramenta de autoria de aplicações multimídia desenvolvida pela empresa Macromedia. As aplicações desenvolvidas são executadas em um browser, exigindo apenas que esteja instalado o plug-in Shockwave.

A tela principal dessa ferramenta é apresentada na Figura 6. Através da barra de ferramentas superior pode ser aberto um editor de texto, onde pode ser criado um texto que será uma mídia integrante da aplicação (nesse caso é criada uma mídia e não inserida uma pronta). Também é possível importar um arquivo externo que contenha alguma mídia que será usada na construção da aplicação, sendo que vários tipos de mídias são suportados. A área 1 da Figura 7 mostra como são visualizadas todas as mídias que estão disponíveis para a construção de uma aplicação. Essas são as mídias criadas através do próprio Director ou aquelas que foram importadas, e somente elas podem ser usadas na criação de uma aplicação. Se for necessária outra mídia, ela deve ser importada e então ficará disponível para ser utilizada.

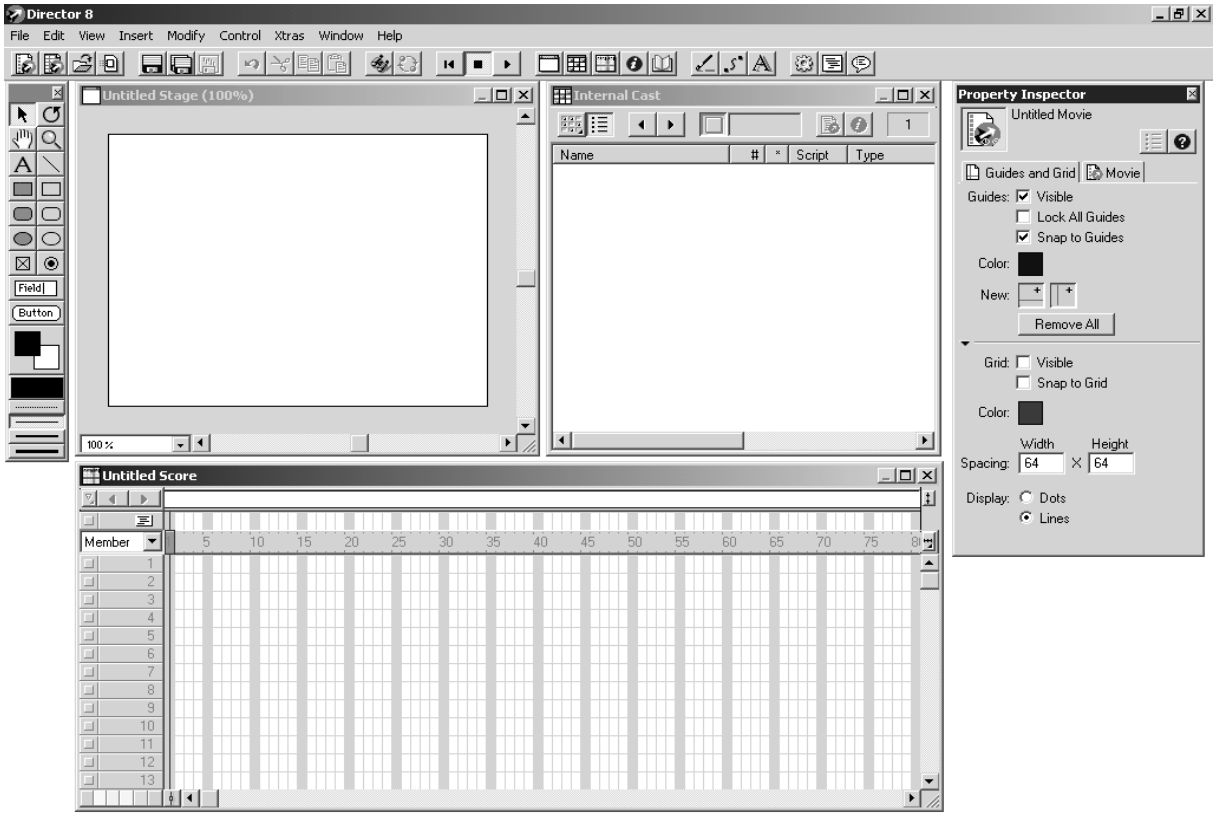


Figura 6 - Tela principal da ferramenta Director

Na área 1 da Figura 8 é mostrado o editor espacial, enquanto que a área 2 mostra o editor temporal. Para inserir uma mídia na aplicação, basta arrastá-la para o editor temporal ou espacial; as alterações realizadas em um refletem no outro.

Através do editor temporal controla-se o editor espacial. Se o marcador do editor temporal estiver no tempo 10s, no editor espacial estarão visíveis somente as mídias que estarão sendo apresentadas no tempo 10s. Se alguma mídia for inserida no editor espacial, ela será visível somente a partir do tempo que estiver indicado no marcador do Editor Temporal. Se o marcador for movido para o tempo 20s e não houver nenhuma mídia visível na aplicação nesse momento, o editor espacial ficará em branco. Dessa forma, conforme o marcador do editor temporal é movimentado, o editor espacial vai modificando e apresentando o que é visível em cada momento. Através disso, se o marcador for movimentado de forma a contemplar todas as mídias, é possível visualizar o resultado final da apresentação através do editor espacial.

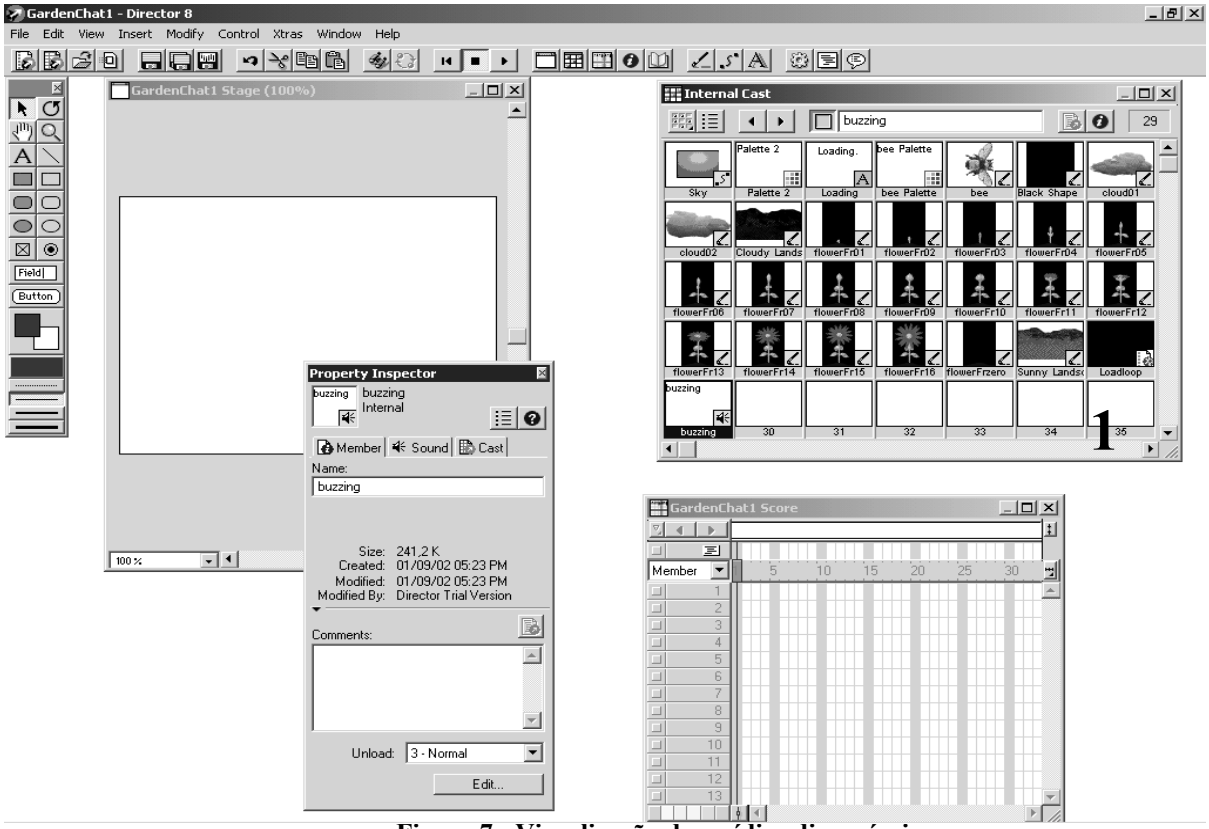


Figura 7 - Visualização das mídias disponíveis

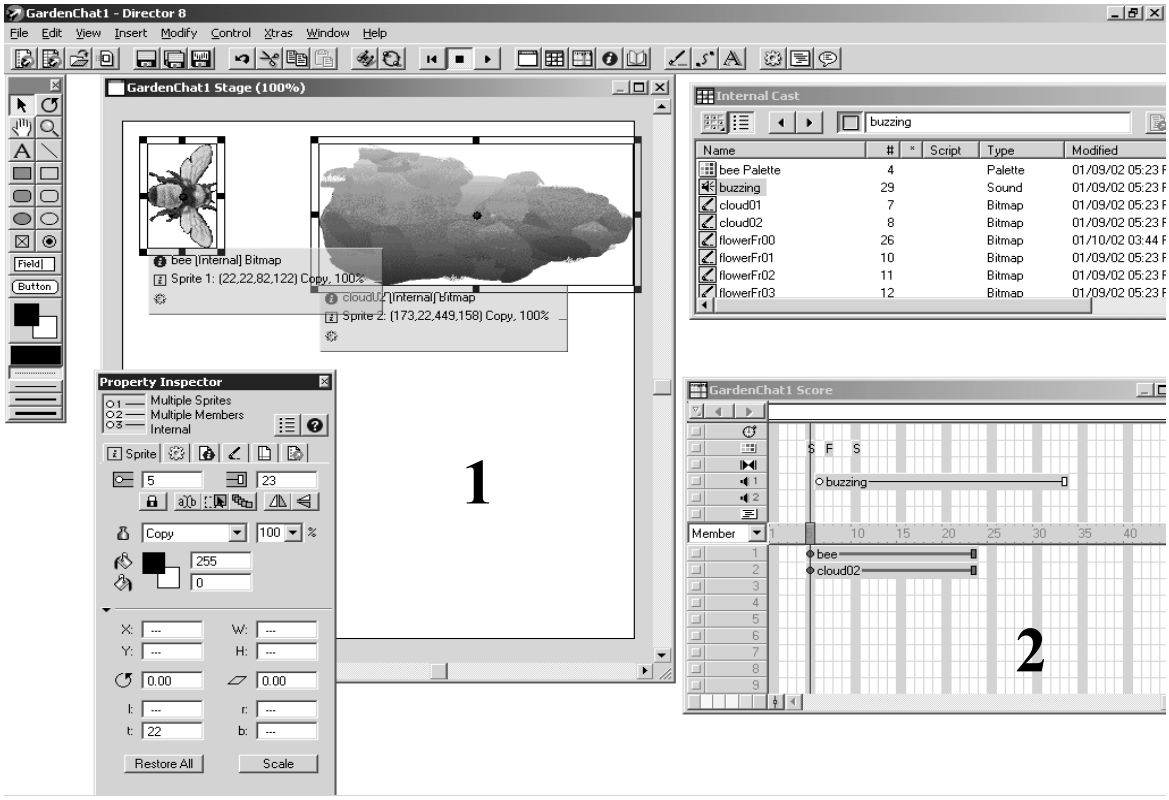


Figura 8 - Editores Temporal e Espacial da ferramenta Director

3.2.3 – GRiNS

A ferramenta GRiNS [Grins, 2002] foi desenvolvida pela companhia Oratrix Development e tem como objetivo o desenvolvimento de aplicações multimídia. Ela usa o padrão SMIL 2.0 como código fonte das apresentações criadas, as quais podem ser executadas em qualquer ferramenta de visualização que aceite documentos SMIL 2.0.

A Figura 9 mostra a tela da ferramenta com o seu Editor Temporal. A configuração temporal é feita através de blocos, sendo que os blocos verdes representam mídias que serão apresentadas em paralelo e os blocos azuis representam mídias que serão apresentadas seqüencialmente. Os blocos amarelos representam as próprias mídias. Na parte superior existe uma barra temporal com graduação em segundos. As mídias são temporalmente configuradas, dentro do seu bloco e em relação às outras mídias, relativamente a essa barra. Através da largura do bloco que representa a mídia são configurados o início e a duração da sua apresentação.

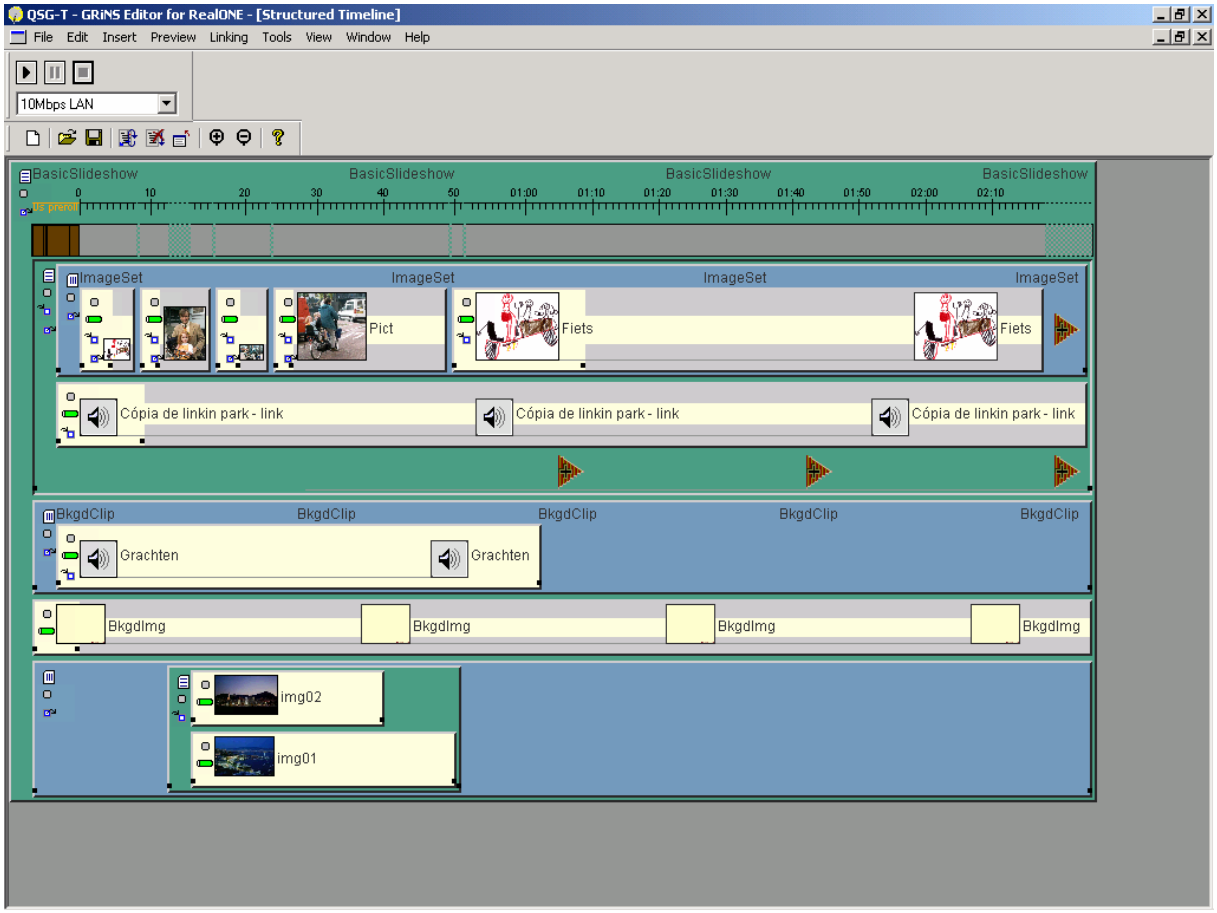


Figura 9 - Ferramenta GRiNS com seu Editor Temporal

A área 1 da Figura 10 mostra o editor de propriedades de uma mídia, que é ativado através de um clique duplo sobre o bloco (amarelo) que contém a mídia. Nesse editor podem ser configurados os atributos temporais e de layout da mídia na apresentação.

É possível visualizar o código SMIL 2.0 referente à apresentação criada. Na Figura 11 é apresentado o código gerado pela ferramenta, o qual pode ser alterado pelo autor.

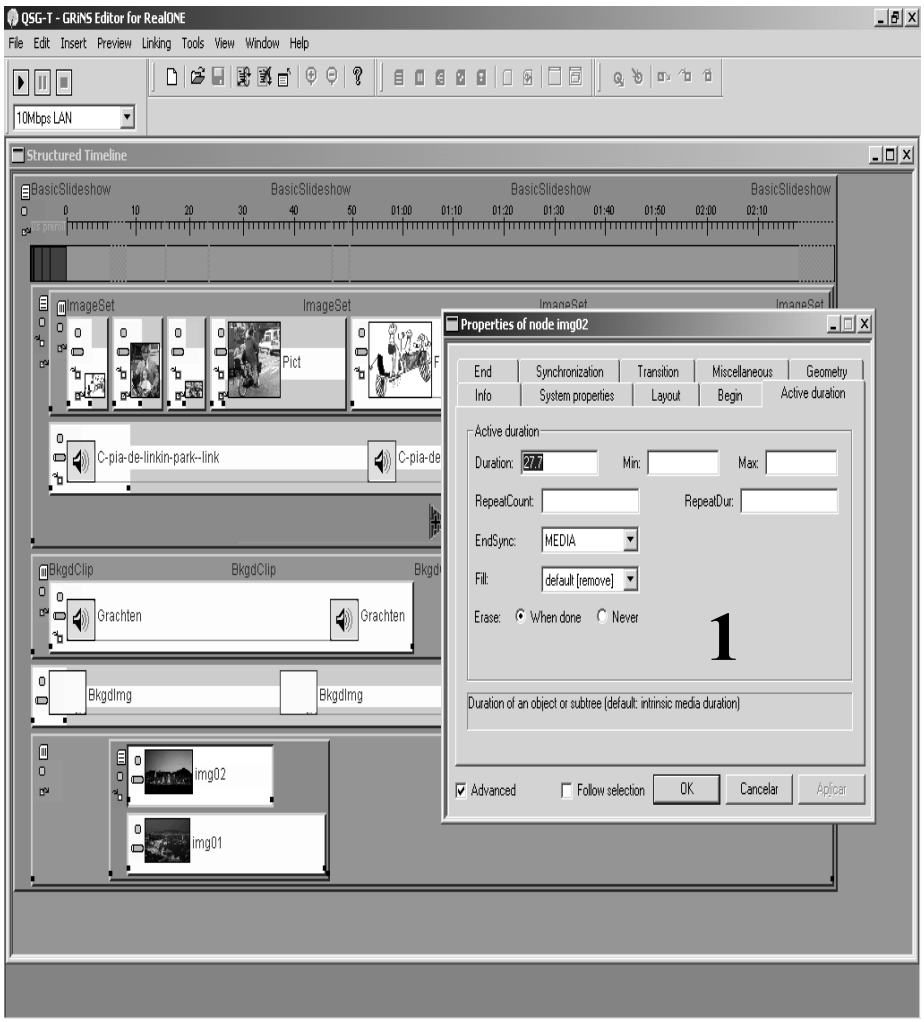


Figura 10 - Ferramenta GRiNS com seu Editor de Propriedades

Na Figura 12 é apresentado o Editor Espacial. A área 1 indica a árvore de objetos que é usada para navegar entre os objetos criados na apresentação; a área 2 mostra a região que representa a aplicação e a área 3 mostra o editor de propriedades de layout da mídia selecionada. Ao ser selecionado um objeto na árvore de objetos, ele se torna visível no editor espacial e pode ser movido para qualquer posição da apresentação. Somente um objeto fica visível no editor temporal. É possível criar regiões e associá-las com as mídias, sendo que as

regiões aqui criadas ficam todas visíveis no editor espacial, possibilitando uma melhor visualização da disposição espacial dos objetos na aplicação.

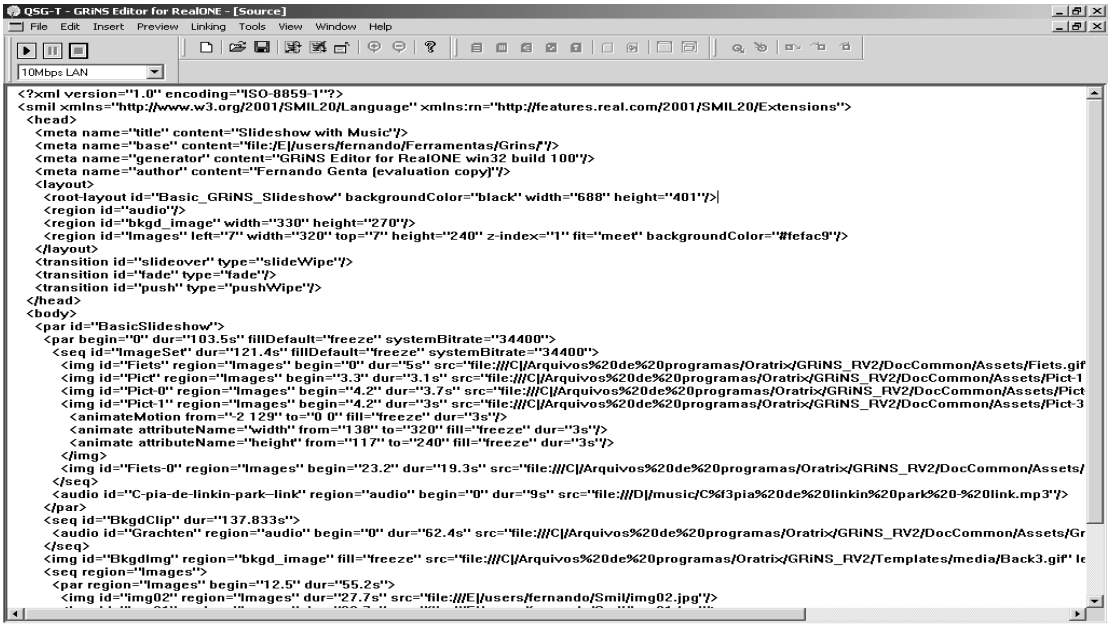


Figura 11 - Código SMIL gerado pela ferramenta GRiNS

Através da ferramenta é possível executar a apresentação criada. A Figura 13 mostra a execução de uma apresentação.

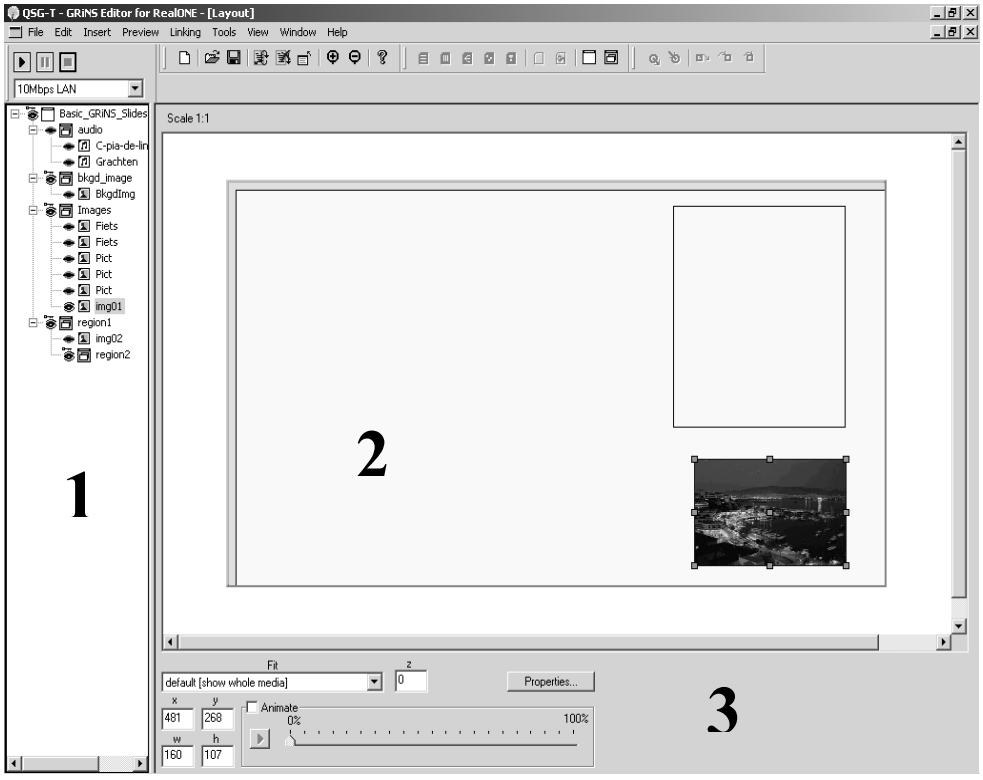


Figura 12 - Editor Espacial da ferramenta GRiNS

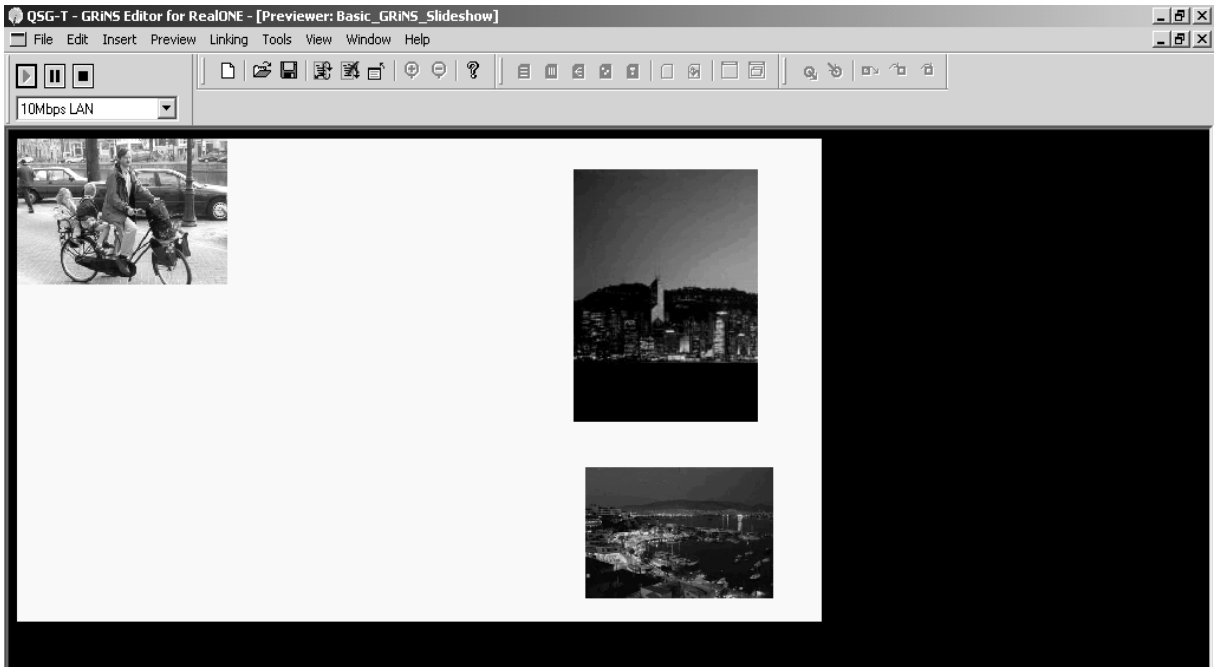


Figura 13 - Visualização através da ferramenta GRiNs de uma apresentação criada

3.2.3 – Fluition

A ferramenta Fluition [Fluition, 2002] foi desenvolvida pela empresa Confluent Technologies de Toronto, Canadá. Ela é uma ferramenta de criação de aplicações multimídia que trabalha com o padrão SMIL 1.0 para gerar o código das aplicações desenvolvidas.

A Figura 14 mostra a tela principal da ferramenta com os seus principais componentes. A área 1 mostra o componente que é usado para manipular o editor espacial, sendo que através dele é possível criar regiões dentro da aplicação, definir a cor de fundo da região e executar a aplicação em desenvolvimento. As mídias serão associadas às regiões criadas, sendo que uma região pode ter mais de uma mídia associada a ela e é através da manipulação das regiões que se determinam a localização e o tamanho da mídia na apresentação.

A ferramenta possui alguns formatos de layout pré-definidos, chamados de templates, que podem ser selecionados no início de uma nova aplicação.

A área 2 da Figura 14 indica o editor espacial, onde é possível trabalhar com as regiões criadas, movimentando-as e redimensionando-as. Através dele se tem uma visão espacial de todas as regiões criadas.

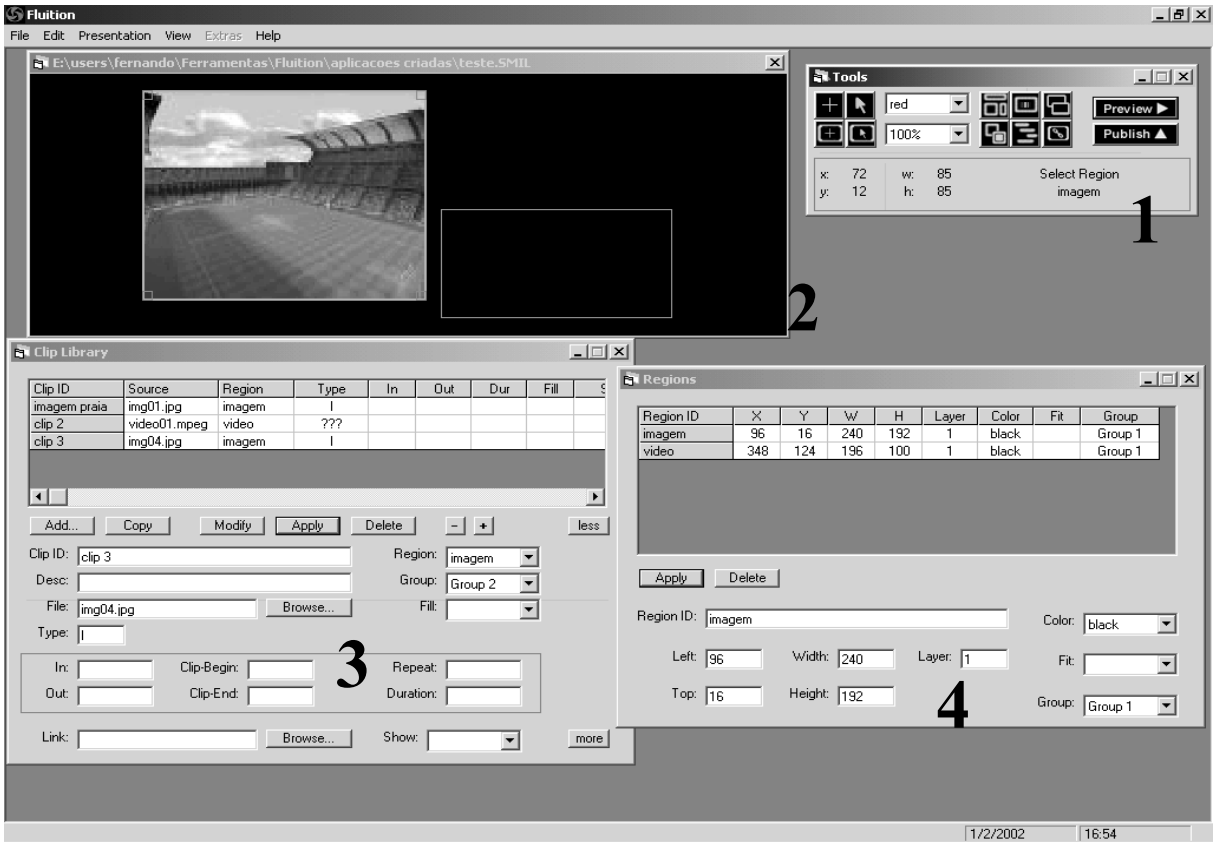


Figura 14 - Tela principal da ferramenta Fluition

A área 3 da Figura 14 mostra o editor de propriedades das mídias e através dele são inseridas as mídias na aplicação e associadas a uma região, além de oferecer todas as configurações temporais e de layout das mídias. A área 4 indica o editor de propriedades das regiões criadas e contém todos os atributos de layout.

A ferramenta também disponibiliza o código fonte para que possa ser alterado, sendo que qualquer alteração é refletida nos outros componentes. Ela possibilita também que o código seja exportado para um arquivo externo, gerando um arquivo SMIL que contém toda a apresentação. A Figura 15 mostra em destaque o código SMIL gerado pela ferramenta.

A ferramenta utiliza grupos para realizar a edição temporal da aplicação. Existem dois tipos de grupos, paralelos e seqüenciais, sendo que quando uma região é criada ela deve ser associada a um grupo. As regiões que estiverem associadas a um mesmo grupo seqüencial serão exibidas uma após a outra, enquanto que as que estiverem associadas a um mesmo grupo paralelo serão executadas em paralelo. Os grupos são seqüenciais entre si, ou seja, primeiro é executado o grupo 1, depois o grupo 2 e assim por diante. As regiões também são seqüenciais; dessa forma se mais de uma mídia for associada à mesma região elas serão

executadas seqüencialmente. A área 2 da Figura 16 mostra o Editor de Grupos, onde são criados os grupos e configurados como seqüenciais ou paralelos.

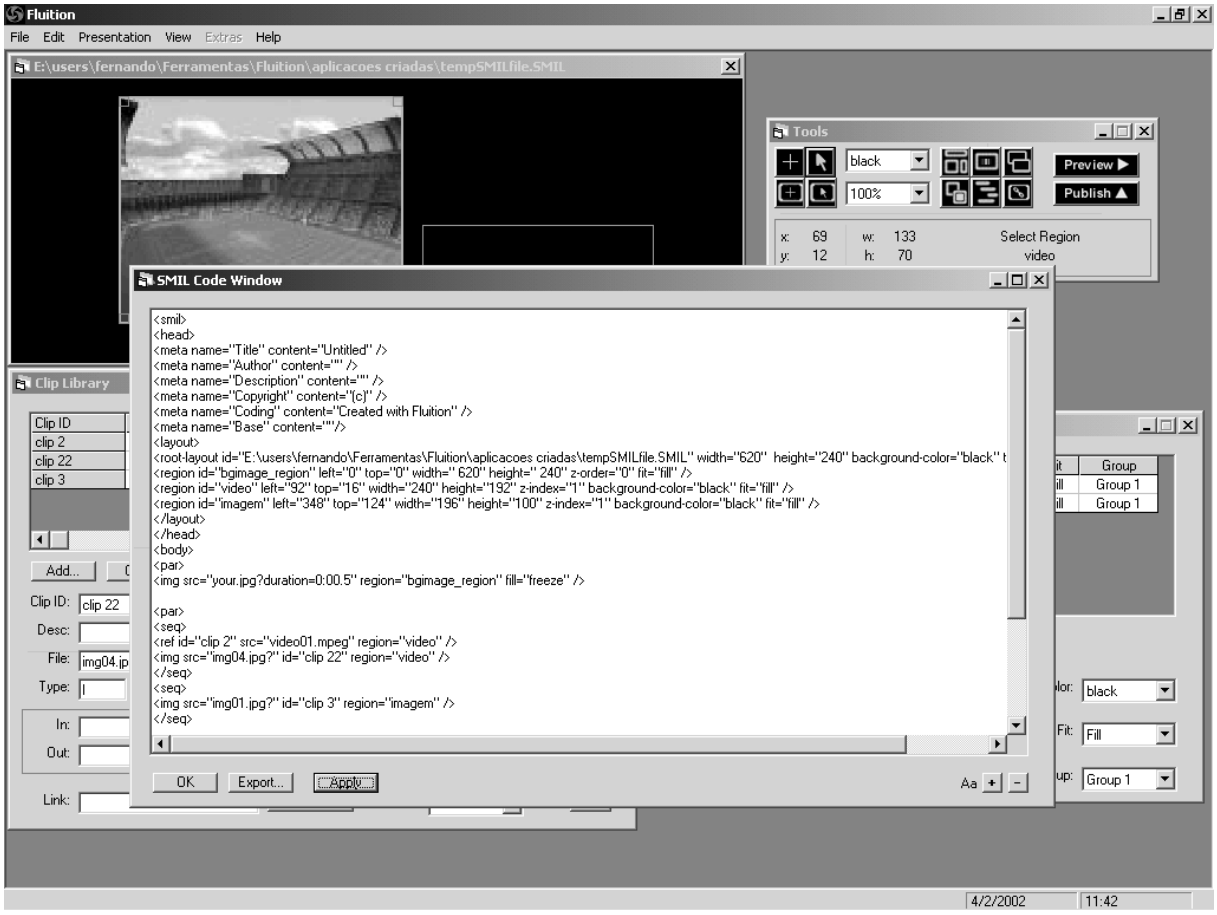


Figura 15 - Código gerado pela ferramenta Fluition

Além de grupos e regiões, a ferramenta também tem um editor temporal, que é usado para configurar temporalmente as mídias dentro das regiões. A área 1 da Figura 16 mostra o Editor Temporal, que só apresenta as mídias referentes à região sobre a qual se está trabalhando. Nesse editor é possível determinar a duração e o tempo de início da exibição de uma mídia, sendo que essas configurações são referentes ao início da execução da região e não da aplicação. É possível alterar a unidade de tempo da apresentação indicada através da barra temporal, sendo que pode ser definida em 1 segundo, 6 segundos ou 1 minuto, ou seja, cada intervalo de tempo da barra temporal pode representar um tempo maior ou menor, dependendo da configuração escolhida.

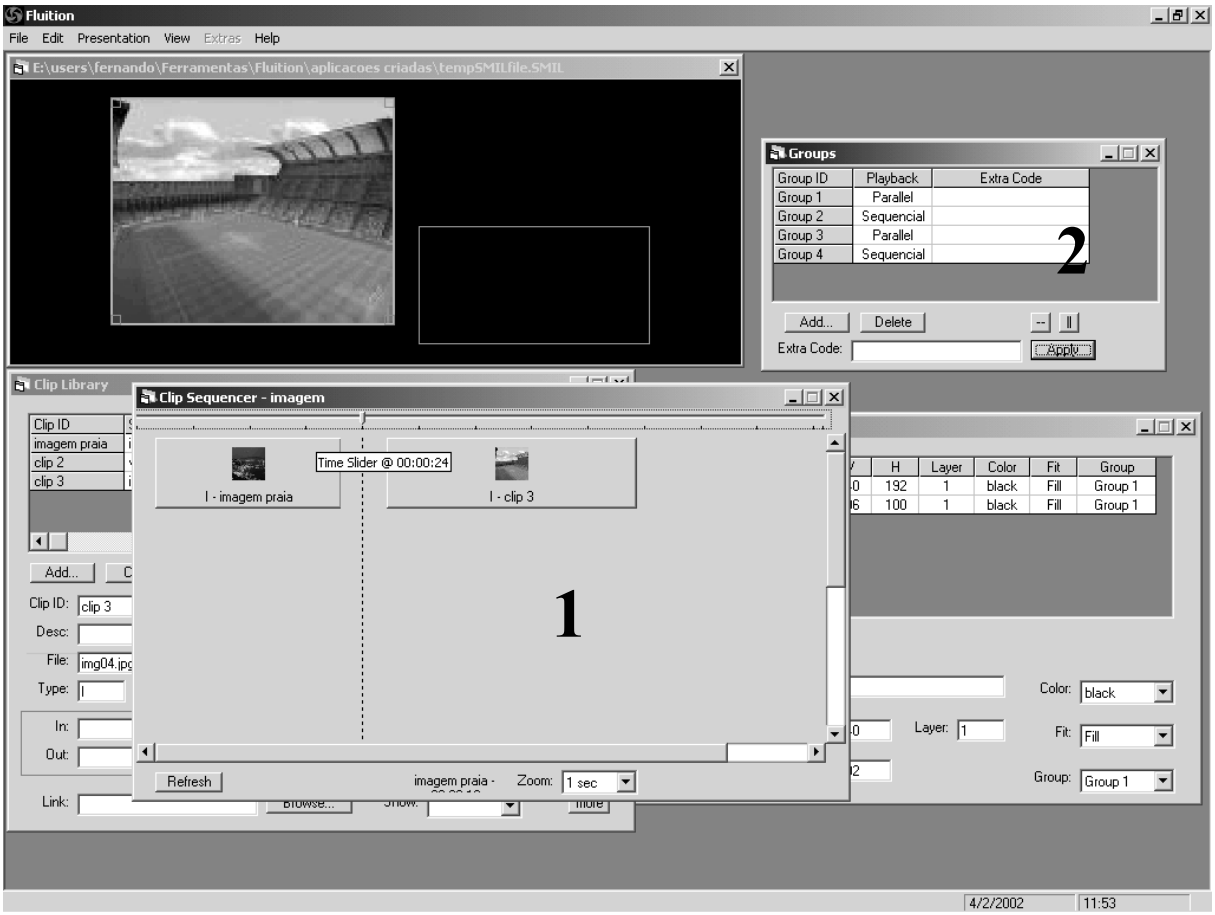


Figura 16 - Modelo de edição temporal da ferramenta Fluition

3.3 Considerações finais

Neste capítulo foi apresentado o conceito de ferramenta de autoria, sendo apresentadas algumas ferramentas. Sobre essas ferramentas foi mostrando o seu funcionamento e suas características. No próximo capítulo é apresentado o ambiente no qual a ferramenta MAE está inserida, sendo discutidas as estruturas de classes criadas para o armazenamento das aplicações.

4. AMBIENTE DE AUTORIA

A ferramenta MAE [Figueiredo, 2000], apresentada no capítulo 5, está inserida no projeto AMMO (*Authoring and Manipulation of Multimedia Objects*) que é desenvolvido pelo Grupo de Banco de Dados do Departamento de Computação da Universidade Federal de São Carlos. A ferramenta MAE tem por objetivo auxiliar, via Web, a construção, manipulação, armazenamento e recuperação de aplicações multimídia baseadas no padrão SMIL, sendo que ela oferece um ambiente visual para a realização dessas operações.

A seguir será dada uma breve explicação sobre o projeto AMMO para contextualizar a ferramenta MAE.

4.1 - AMMO

O objetivo do projeto AMMO é prover um ambiente para suporte à autoria, armazenamento e manipulação de aplicações multimídia. Esse projeto está sendo desenvolvido de forma a viabilizar a sua utilização na Web.

A arquitetura geral do AMMO é ilustrada na Figura 17, a qual possui os seguintes componentes:

Multimedia Objects Server (MmOS), responsável por gerenciar as aplicações multimídia armazenadas em um banco de dados multimídia;

Multimedia Application WebBuilder (MAW), provê o suporte à autoria de aplicações multimídia na Web. Os módulos do MAW foram implementados em Java, com o objetivo serem executdos em qualquer Web browser, além de prover uma grande portabilidade. A ferramenta MAE está inserida neste módulo

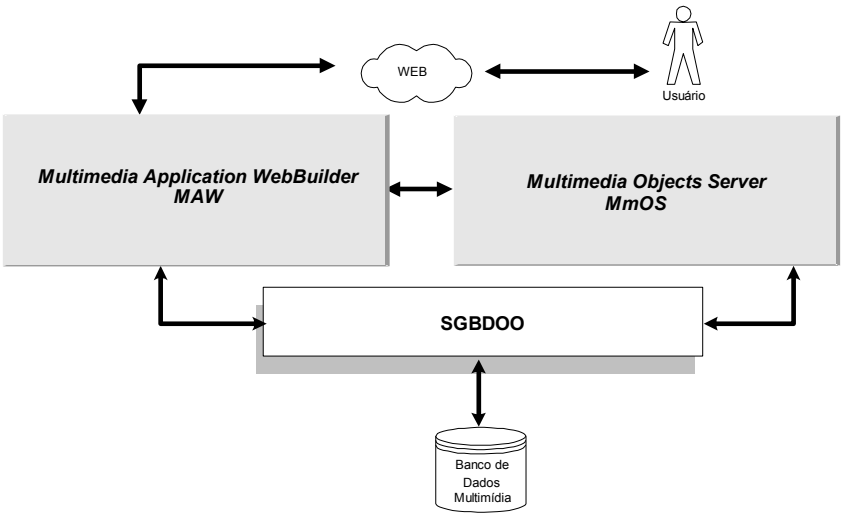


Figura 17 - Estrutura do Projeto AMMO

Interface WEB, provê uma interface para realizar consultas ao banco de dados multimídia e construir aplicações através da Web;

Sistema Gerenciador de Banco de Dados Orientado a Objetos (SGBDOO), gerencia os objetos mantidos no banco de dados multimídia.

Para armazenar as aplicações criadas através da ferramenta MAE, foi desenvolvida uma estrutura de classes baseada no padrão SMIL. No ambiente do AMMO existe um módulo de consultas inserido no MmOS, onde é possível realizar buscas baseadas em informações sobre o conteúdo das mídias [Borges, 2001], [Fornazari, 1999], [Fornazari et al., 1999], [Vieira & Santos, 1997], [Vieira et al. 1999a], [Vieira et al. 1999b], [Santos et al., 2000], [Vieira et al, 2002]. Para viabilizar essas buscas, foi criada uma estrutura de classes para armazenar as informações semânticas relativas às mídias que fazem parte das aplicações multimídia armazenadas no ambiente. Essas estruturas de classe são apresentadas a seguir.

4.2 - Estrutura de Classes do AMMO

4.2.1 - Armazenamento de Aplicações

A Figura 18 apresenta a estrutura de classes que foi construída, no projeto AMMO, para armazenar aplicações segundo o padrão SMIL. Mais adiante será apresentada a estrutura usada para armazenar as informações semânticas utilizadas nas consultas. Para a modelagem foi utilizada a linguagem UML (Unified Modeling Language).

No contexto deste projeto, uma aplicação multimídia é constituída de um conjunto de documentos SMIL, sendo que um documento SMIL é considerado uma cena dentro da aplicação.

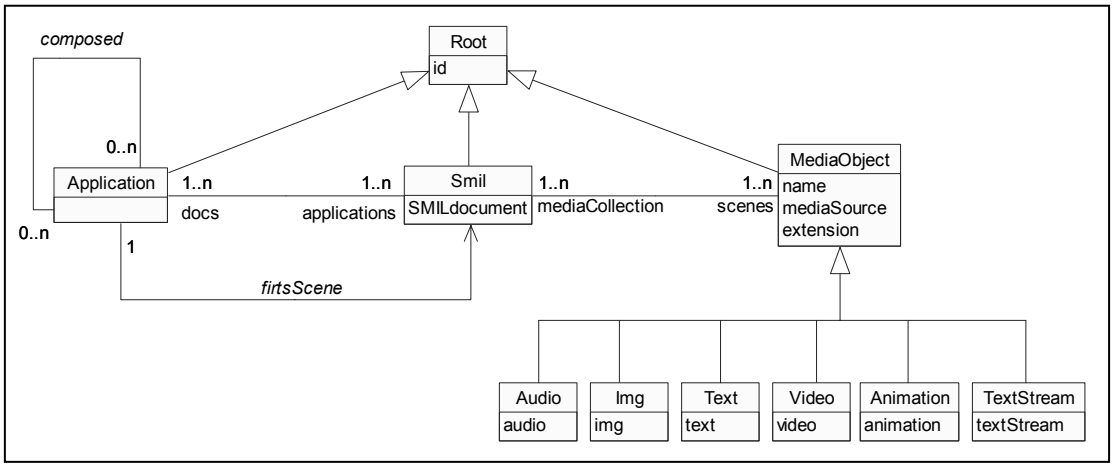


Figura 18 - Estrutura de classes do projeto AMMO para armazenamento de aplicações

A classe *Root* generaliza todas as classes da estrutura. Ela possui o atributo *id,* o qual é herdado por todas as classes que fazem parte de sua hierarquia para representar um identificador único de cada objeto pertencente à aplicação.

A classe *Application* agrega os documentos SMIL que compõem uma aplicação e através da agregação *docs* são mantidos todos os documentos (cenas) que fazem parte da aplicação. A associação *firstScene* indica qual é a primeira cena da aplicação e a seqüência de apresentação é definida nos próprios documentos SMIL através de marcações (*tags*) que representam ligações (*links)* para outros documentos. Através dessas ligações o autor define a navegação entre as várias cenas que compõem a aplicação, sendo que apenas a primeira cena deve ser indicada para dar início à execução da aplicação.

Na classe SMIL são armazenados os documentos SMIL, que pertencem a uma ou mais aplicações. O atributo *SMILdocument* armazena o código SMIL do documento e a associação *mediaCollection* faz referência a todas as mídias que são utilizadas por esse documento. A associação a*pplications* indica a quais aplicações pertence esse documento.

Com relação ao tratamento do armazenamento das mídias, existem duas possibilidades: a mídia pode ser armazenada no próprio banco de dados, ou ela está em outro local, sendo que no banco de dados existe apenas uma referência a esse local.

A classe MediaObject generaliza todas as classes que armazenam as mídias, possuindo atributos comuns a todas elas. Em *mediaSource* é armazenada a referência ao local em que está armazenada a mídia, caso esta não esteja armazenada no banco de dados. O atributo *extension* indica qual a extensão do arquivo da mídia e em *name* está o nome da mídia. A associação s*cenes* indica a quais documentos SMIL a mídia pertence.

As especializações da classe MediaObject são as classes Video, Image, Audio, TextStream, Text e Animation. Elas contêm as mídias que estão armazenadas no banco de dados, no caso dessa abordagem estar sendo usada. Os atributos *video, img, audio, textStream, text* e *animation* guardam as mídias.

A Figura 19 apresenta um exemplo de documento SMIL. Esse exemplo exibe primeiro uma imagem durante 25 segundos e, terminada a exibição da imagem, inicia-se a exibição paralela de um vídeo e um áudio. Na mídia "*video02.avi*" foi criada uma ligação, através da marcação <a>, para o arquivo "segundaCena.smi".

A Figura 20 apresenta o arquivo "segundaCena.smi", outro documento SMIL, onde é exibido um vídeo e em seguida uma imagem durante 5 segundos.

Esses documentos SMIL serão usados para exemplificar como é o armazenamento de uma aplicação na estrutura da Figura 18.

O documento da Figura 19 faz referência a três mídias, identificadas por *”http://www.programguiden.se/honda/brasil.JPG”*, *“video02.avi”* e “*audio01.wav”*. Quando esse arquivo SMIL for recuperado para ser apresentado, é preciso que as mídias, as quais ele faz referência direta ao arquivo, também sejam recuperadas. Dessa forma, no armazenamento desse documento deve existir uma indicação de que essas mídias estão associadas a ele, para que seja possível a recuperação delas sempre que esse documento for requisitado. O mesmo acontece para o documento da Figura 20. Com relação à imagem, que tem como origem o endereço *”http://www.programguiden.se/honda/brasil.JPG”*, o tratamento é diferente, uma vez que a mídia não está armazenada no banco de dados; durante a execução da cena, a própria ferramenta de apresentação procura pela mídia no endereço especificado e a apresenta segundo as configurações definidas no documento SMIL.

```
<smil>
 <head>
  <layout>
   <root-layout height="189" width="483" background-color="black"/>
   <region id="video" left="0" top="0" height="189" width="283"/>
   <region id="image" left="285" top="0" height="189" width="200"/>
  </layout>
 </head>
 <body>
  <seq>
   <img src=”http://www.programguiden.se/honda/brasil.JPG” region="image" dur="25s"/>
   <par>
       <a id="link_segundaCena" show="replace" href="segundaCena.smi">
           <video src="video02.avi" region="video"/>
       </a>
      <audio src="audio01.wav"/>
   </par>
  </seq>
 </body>
</smil>
```

Figura 19 - Exemplo de um documento SMIL (primeira cena)

```
<smil>
 <head>
    <layout>
      <root-layout height="250" width="500" background- color="black"/>
      <region id="video" left="0" top="0" height="189" width="283"/>
      <region id="image" left="285" top="0" height="189" width="200"/>
    </layout>
  </head>
  <body>
    <seq>
      <video src="video03.mpg" region="video"/>
      <img src="img02.jpg" region="image" dur="5s"/>
    </seq>
  </body>
</smil>
```

Figura 20 - Exemplo de outro documento SMIL (segunda cena)

Nesse exemplo a aplicação é constituída de duas cenas, apenas para servir de ilustração, sendo que o documento da Figura 19 é a primeira cena da aplicação. Como foi mostrado, a navegação entre as cenas é definida através do próprio documento SMIL.

Para armazenar essa aplicação, primeiro é criado um objeto, AP1 (nome dado ao objeto pelo autor da aplicação), da classe *Application*, que vai receber todas as cenas da aplicação.Também são criados dois objetos, S1 e S2, da classe *SMIL*, sendo que:

S1.*SMILdocument* - armazena o código SMIL da Figura 19
S1.a*pplications* - faz referência a AP1
S2.*SMILdocument* - armazena o código SMIL da Figura 20
S2.a*pplications* - faz referência a AP1

Como AP1 contém todas as cenas que compõem a aplicação, então:

AP1.docs - faz referência a S1 e a S2

Como a cena inicial da aplicação é S1, então temos:

AP1.firstScene – faz referência a S1

Será usado o documento da Figura 19 para exemplificar como seria o armazenamento de uma cena. O armazenamento é feito de forma análoga para o documento da Figura 20. Considerando que as mídias estarão armazenadas no banco de dados, então é criado um objeto, V1, da classe *Vídeo*, um objeto, I1, da classe Image e um objeto, A1, da classe *Audio*. Esses objetos armazenam as mídias que fazem parte da cena e são instanciados da seguinte maneira:

V1.video – armazena o arquivo "video02.avi"

V1.scenes – faz referência a S1
V1.name – armazena a string "montagem de micro"
V1.mediaSource – não é instanciado
V1.extension – armazena a string “avi”

I1.img – não é instanciado
I1.scenes – faz referência a S1
I1.name – armazena a string "placa de som"
I1.mediaSource – armazena a string “http://www.programguiden.se/honda/brasil.JPG”
I1.extension – armazena a string “jpg”

A1 audio – armazena o arquivo "audio01.wav"
A1.scenes – faz referência a S1
A1.name – armazena a string "narração"
A1.extension – armazena a string “wav”
A1.mediaSource – não é instanciado

O documento SMIL também deve referenciar todas as mídias que fazem parte dele, e é através da associação mediaCollection que isso é feito. Dessa forma:

S1.mediaCollection – faz referência a V1, I1 e A1

Esse exemplo ilustrou como ficaria o banco de dados com a inserção de uma aplicação. Na próxima seção será discutida a inserção de informação semântica referente às mídias que compõem as aplicações.

4.2.1 - Armazenamento de Informação Semântica

A estrutura da Figura 21 mostra como estão integradas as estruturas de armazenamento das informações semânticas (área 1) e das aplicações (área 2). Através dela é possível verificar como as mídias estão associadas às suas informações semânticas.

A área 1 da Figura 21 mostra as classes que foram criadas para armazenar a informação semântica das mídias e que dão suporte à busca por conteúdo e busca nebulosa no ambiente do AMMO. Vários projetos de pesquisa (concluídos e em andamento) têm contribuído na construção e refinamento dessas classes [Borges, 2001], [Fornazari, 1999], [Fornazari et al., 1999], [Vieira & Santos, 1997], [Vieira et al. 1999a], [Vieira et al. 1999b], [Santos et al., 2000], [Vieira et al, 2002]. Para cada mídia, a informação semântica que pode ser considerada envolve temas, seus qualificadores e associações existentes entre os temas. As informações semânticas de uma mídia referem-se ao seu conteúdo.

A classe *Semantic_information* armazena informações semânticas sobre as mídias (áudio, vídeo, imagem etc) e é formada por temas, identificados na estrutura por *subjects*, e por associações, identificadas por *subjectAssoc,* entre esse temas. A classe *Subject* possibilita

guardar informações sobre os temas que compõem uma mídia. Por exemplo, na Figura 22 podem ser identificados os temas *lago*, *morro alto e sem vegetação*, *prédio*, *pássaro branco*. Nesse caso foram identificados quatro temas para uma imagem. Os temas são compostos por um substantivo (*morro*, *lago, prédio*, *pássaro*), identificado na estrutura pela associação *identifier,* e por um ou mais adjetivos para esse substantivo (*alto*, *sem vegetação*, *branco),* identificados através de *qualifiers.*

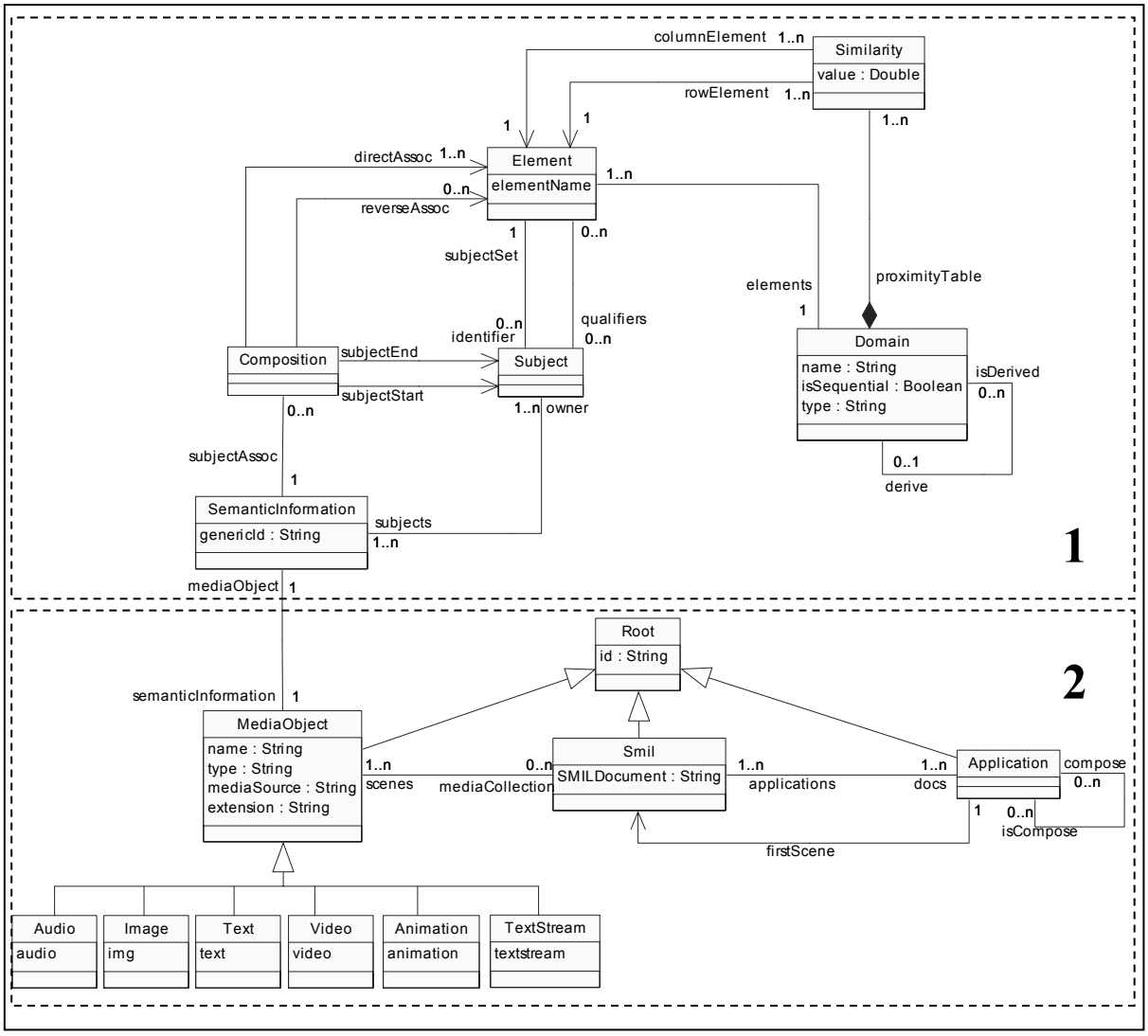


Figura 21 - Estrutura do AMMO de armazenamento das aplicações e informação semântica

A classe *Composition* é destinada a armazenar a composição de associações entre os temas existentes em uma mídia, como, por exemplo, *morro* [*alto e sem vegetação*] ***próximo à*** *lago*. O termo ***próximo à*** faz a associação entre os dois temas, sendo que *morro* [*alto e sem vegetação*] é o tema inicial (identificado na estrutura por *subjectStart*) e *lago* é o tema final (identificado por *subjectEnd*) da associação. Pode-se armazenar também a associação reversa entre dois temas, como por exemplo, no caso de *lago* ***com*** *pássaro* [*branco*] (associação

direta, sendo que o termo ***com*** é identificado na estrutura por *directAssoc*) e *pássaro* [*branco*] ***à margem do*** *lago* (associação reversa, sendo que o termo ***à margem do*** é identificado na estrutura por *reverseAssoc*).

Figura 22 - Imagem armazenada no BDMm do AMMO

A classe *Element* destina-se a armazenar todos os termos que compõem a informação semântica, quer eles representem temas (*morro*, *lago, prédio*, *pássaro*), qualificadores (*alto*, *sem vegetação*, *branco*) ou associações (*na, com*, *próximo à*). Esses termos são armazenados no atributo *elementName.* Um objeto da classe *Element* pode conter mais de um termo no seu atributo *elementName* (implementado como uma coleção de *strings*), no caso deles serem semanticamente iguais, como, por exemplo, *guri* e *garoto*, ou *belo* e *bonito*, ou ainda *perto de* e *próximo de*.

A classe *Domain* é usada para guardar informações sobre os domínios dos elementos considerados (temas, qualificadores e associações). Um exemplo de domínio seria *Animais* (armazenado no atributo *name*), que seria composto pelos elementos *pássaro*, *cachorro*, *gato*, *cavalo*, etc (indicados pela associação *elements*).

A classe *Similarity* armazena informações sobre a similaridade existente entre elementos de um mesmo domínio. Essa classe foi criada para dar suporte a busca nebulosa desenvolvida no projeto AMMO. Para cada domínio instanciado é armazenada uma tabela de proximidade *(proxmityTable*), a qual é uma representação da relação de proximidade entre os elementos de um mesmo domínio. Essa relação de proximidade associa um valor, compreendido no intervalo [0,1], a cada par de elementos do domínio, os quais são mantidos na classe *Similarit*y, através do elemento linha *(rowElemen*t), coluna *(columnElemen*t) e o grau de similaridade entre eles *(valu*e).

4.3 Considerações finais

Neste capítulo foi apresentado o ambiente AMMO, no qual a ferramenta MAE está inserida, sendo apresentadas as estruturas de classes criadas para o armazenamento das aplicações e informação semântica. O próximo capítulo mostra a ferramenta MAE, apresentando a sua arquitetura, a estrutura de classes utilizada na sua construção, seus módulos e suas funcionalidades. Também é feita uma comparação com as ferramentas apresentadas no capítulo 3.

5. FERRAMENTA MAE

O processo de desenvolvimento de uma ferramenta de autoria, como qualquer outro software, tende a seguir as etapas de especificação, modelagem e implementação do problema. Na MAE [Santos et al., 2002a], [Santos et al., 2002b] optou-se por obedecer ao padrão SMIL em todos os aspectos, isto é, a conformância ao padrão determina desde os aspectos funcionais até aspectos de interface da ferramenta. Essas questões ficarão mais claras quando forem aqui discutidas.

Na definição do funcionamento da ferramenta, existem muitos aspectos que devem ser tratados para que a ferramenta seja amigável para o autor, de forma a tornar fácil o processo de desenvolvimento de uma aplicação multimídia. As soluções apresentadas devem levar em consideração não só o desenvolvimento de uma interface amigável, mas também as características da linguagem usada para a construção da aplicação. Dependendo da linguagem de implementação utilizada, diferentes soluções podem ser apresentadas.

No caso da ferramenta MAE, as soluções desenvolvidas levam em conta as características da linguagem SMIL 1.0, que tentou deixar isso o mais transparente possível para o autor, de forma que ele não precise ter um grande conhecimento dessa linguagem para desenvolver suas aplicações.

Neste capitulo é apresentada a ferramenta MAE, mostrando a sua arquitetura e explicando cada um dos módulos que a compõem. Também é mostrada a interface da ferramenta explicando o seu funcionamento. Por fim é feita uma comparação da MAE com as ferramentas que foram apresentadas no capítulo 3.

5.1 Arquitetura do ambiente de autoria e manipulação de aplicações multimídia

A ferramenta MAE foi implementada na linguagem Java [Java, 2000] e é executada via World Wide Web (Web) para a criação e alteração de aplicações multimídia baseadas no padrão SMIL 1.0 [SMIL 1998]. Ela interage com o Sistema Gerenciador de Banco de Dados (SGBD) Caché [Caché, 2002] para o armazenamento e recuperação de aplicações multimídia.

A Figura 23 apresenta os principais componentes da ferramenta MAE e a interação com outros módulos do ambiente AMMO. A área 1 indica os módulos pertencentes à ferramenta, enquanto que a área 2 indica os módulos pertencentes ao AMMO. Uma primeira versão dessa ferramenta foi desenvolvida por Figueiredo [Figueiredo, 2000], se concentrando na construção dos módulos Editor de Propriedades, Editor Textual, Árvore de Objetos e

Editor Espacial. Também foi definido um modelo para o Editor Temporal, que não chegou a ser implementado.

Neste trabalho foi construída uma nova versão da ferramenta MAE. Foi desenvolvida uma nova versão do Editor Espacial, dando mais recursos a ele, como também foi desenvolvido e implementado um novo modelo para o Editor Temporal, diferente do modelo proposto na primeira versão.

Foi feita também a integração da ferramenta com o banco de dados, que não havia na primeira versão. Os módulos Acesso à Base de Dados, Inserção/Recuperação de Aplicações e Inserção/Recuperação de Informação Semântica foram desenvolvidos nesta nova versão.

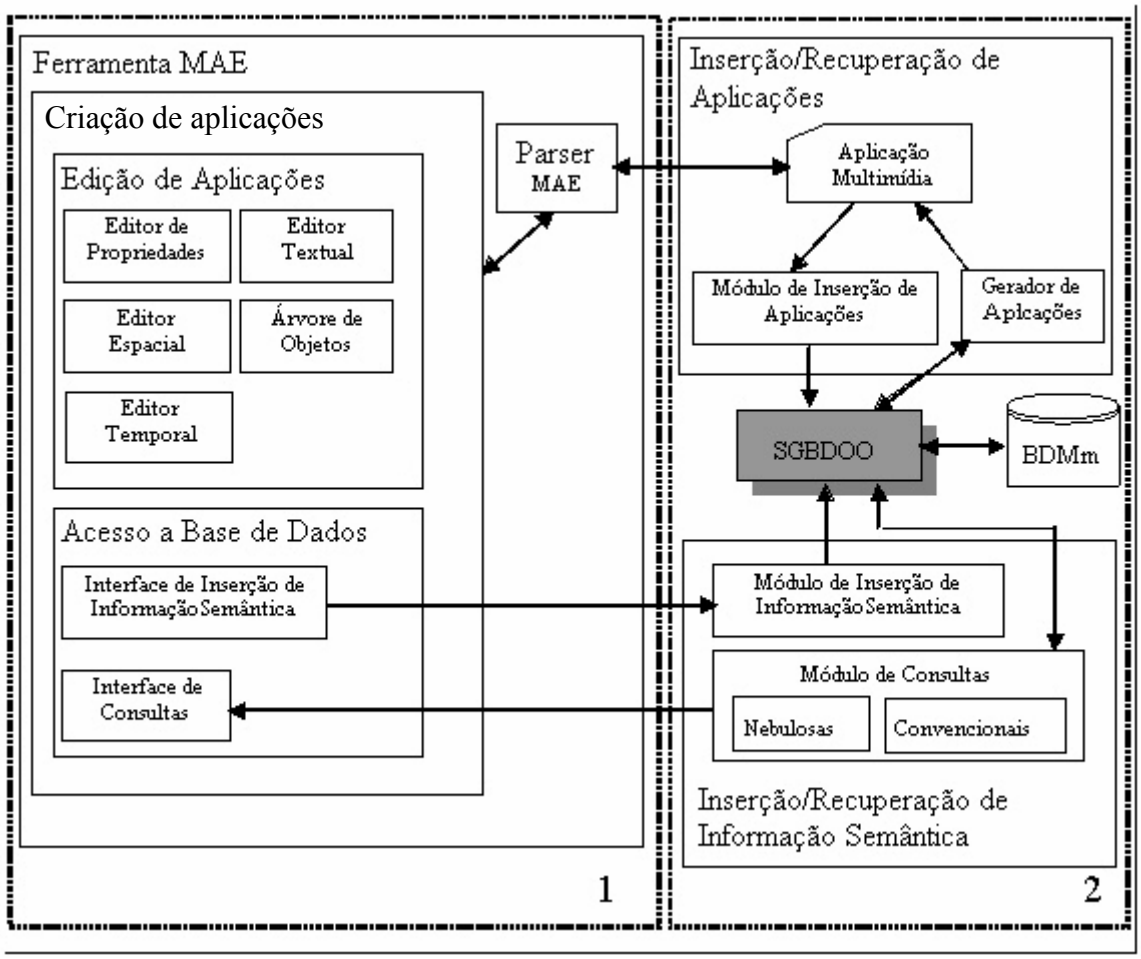


Figura 23 - Arquitetura do ambiente de autoria e manipulação de aplicações multimídia

O módulo Edição de Aplicações possui as funcionalidades oferecidas ao usuário para a criação/edição de uma aplicação. Já o módulo Acesso à Base de Dados oferece os recursos necessários para que o autor possa recuperar mídias e cenas armazenadas no banco de dados multimídia (BDMm). Além disso, é através dele que o autor insere a informação semântica no BDMm. Esse módulo interage com o módulo Inserção/Recuperação de informação semântica,

que possui os métodos necessários para a manipulação da informação semântica mantida no BDMm. O módulo Inserção/Recuperação de Aplicações manipula a estrutura de armazenamento das aplicações do BDMm, armazenando as aplicações criadas através da MAE e também possibilitando a recuperação e edição, através da MAE, das aplicações armazenadas. Maiores detalhes sobre os módulos componentes da arquitetura apresentada na Figura 23 são apresentados nas próximas seções.

5.2 Estrutura de classes da MAE

A modelagem das informações que serão tratadas pela ferramenta tem como intuito determinar de que forma a ferramenta representa e manipula cada uma dessas informações. Nesse sentido, foi definido que seria utilizada uma representação através de classes de objetos, utilizando o paradigma da Orientação a Objetos.

A Figura 24 apresenta as principais classes criadas no desenvolvimento da ferramenta MAE e as associações entre elas. As classes da área 1 são referentes ao padrão SMIL, enquanto que as classes da área 2 são aquelas principais que compõem a ferramenta em si. A classe MAE agrega todas as classes que compõem a ferramenta, fornecendo uma interface única para a utilização de todos os recursos oferecidos. Além disso, também está associada com a estrutura do padrão SMIL, uma vez que todas as configurações feitas pela ferramenta no desenvolvimento de uma aplicação ficam armazenadas nessa estrutura SMIL.

A estrutura de classes completa do padrão SMIL é apresentada na Figura 25 descrita ainda nesta seção, enquanto que as classes que compõem a ferramenta são descritas nas próximas seções.

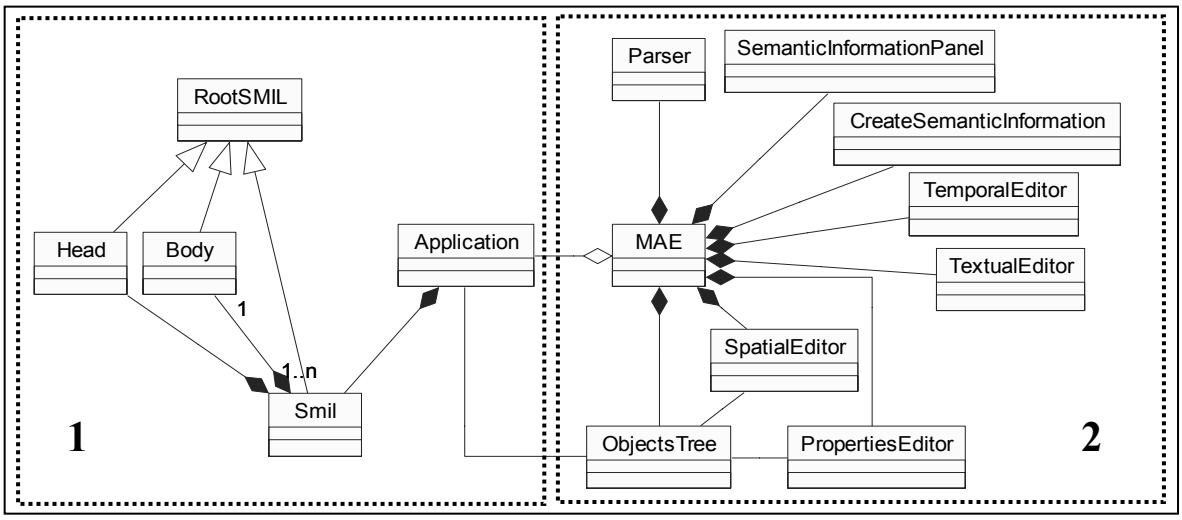


Figura 24 - Classes da ferramenta MAE

Para manipular as informações referentes ao padrão SMIL, Figueiredo [Figueiredo, 2000] criou uma classe para cada tag e cada elemento definidos na DTD [SMIL, 1998] do SMIL 1.0, conforme mostrado na Figura 25. Foi introduzida uma classe adicional, a classe Application (área 1 da figura) não prevista no padrão. Seu objetivo é permitir representar uma aplicação multimídia como constituída de vários documentos SMIL, onde cada documento é uma cena da aplicação. O padrão SMIL considera que uma aplicação multimídia real é representada em um único documento SMIL, independente de sua complexidade. Considerou-se que toda complexidade de uma aplicação real tende a ficar mais fácil de ser gerenciada se partes específicas da aplicação forem divididas em documentos SMIL separados, pois um conjunto de documentos simples é mais adequado para a manipulação e compreensão do que um único documento complexo.

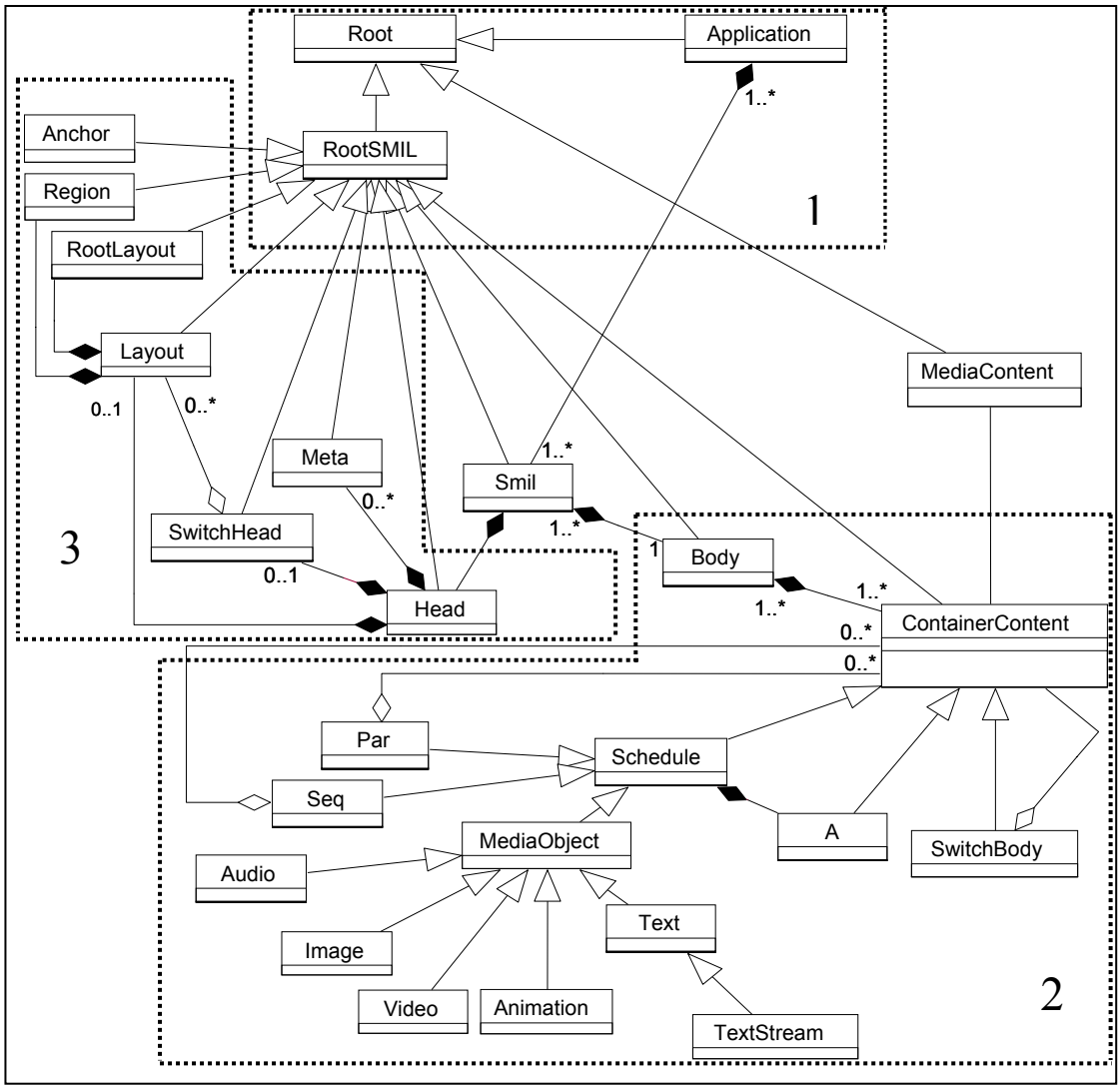


Figura 25 - Estrutura de classes SMIL

Além da classe Application, tem-se também, na área 1, a classe Root, que generaliza todas as classes da estrutura, possuindo atributos e métodos comuns a todas elas, e a classe RootSMIL que generaliza todas as classes que fazem parte do padrão SMIL.

Um objeto Smil é composto de um objeto do tipo Body e um do tipo Head. Na área 2 da Figura 25 são destacadas as classes que compõem a classe Body, que são aquelas que influenciam no comportamento temporal da aplicação. As classes Par e Seq determinam, respectivamente, o comportamento paralelo e seqüencial das mídias. As classes Audio, Image, Video, Animation, Text e TextStream tratam dos atributos das mídias, como por exemplo a que região está associada e a sua duração. A classe A trata das ligações feitas entre as cenas de uma aplicação. As outras classes da área 2 tratam de questões bem específicas do padrão SMIL, que aqui não são relevantes.

Na área 3 da Figura 25 são destacadas as classes que compõem a classe Head, que agrega as classes que contém informações sobre o layout da apresentação. As classes mais importantes são: Layout, que contém algumas definições sobre a aparência da apresentação; Region que contém informação sobre as regiões; e Anchor que também trata de ligações entre as cenas.

A classe MediaContent não está relacionada ao padrão SMIL, porém mantém informações sobre as mídias, tais como nome e extensão.

5.3 Descrição dos Módulos Componentes da Ferramenta MAE

A Figura 26 mostra a tela principal da ferramenta, onde na área 1 tem-se a Árvore de Objetos, que mostra, através de uma hierarquia, os objetos que compõem uma aplicação multimídia. A área 2 indica o Editor de Propriedades, que mostra as propriedades do objeto selecionado através da Árvore de Objetos, do Editor Espacial ou do Editor Temporal. A área 3 mostra o Editor Espacial, onde é possível criar regiões e configurar o tamanho e o posicionamento delas. A área 4 mostra as paletas que são usadas para acessar todos os componentes da ferramenta. Nas próximas seções são apresentados todos esses componentes e os módulos da ferramenta associados a eles.

5.3.1 Árvore de Objetos

Um primeiro módulo implementado com o intuito de facilitar a interface com o usuário é a Árvore de Objetos. A principal característica da Árvore é fornecer uma visão geral da aplicação que está sendo desenvolvida, permitindo que todos os objetos manipulados possam ser selecionados e alterados.

A área 1 da Figura 26 mostra a Árvore de Objetos adotada na ferramenta MAE. Nela, através de uma hierarquia, são apresentados os objetos que compõem uma aplicação multimídia. Essa representação foi utilizada porque apresenta ao autor uma visão geral da aplicação, mostrando as várias cenas (documentos SMIL) que a compõem. Além disso, para cada documento, é possível visualizar quais mídias foram incluídas e quantas regiões foram criadas. Com isso, ele permite que o usuário navegue facilmente entre as várias cenas que fazem parte da aplicação e entre os vários objetos que compõem as cenas, permitindo a alteração de suas propriedades.

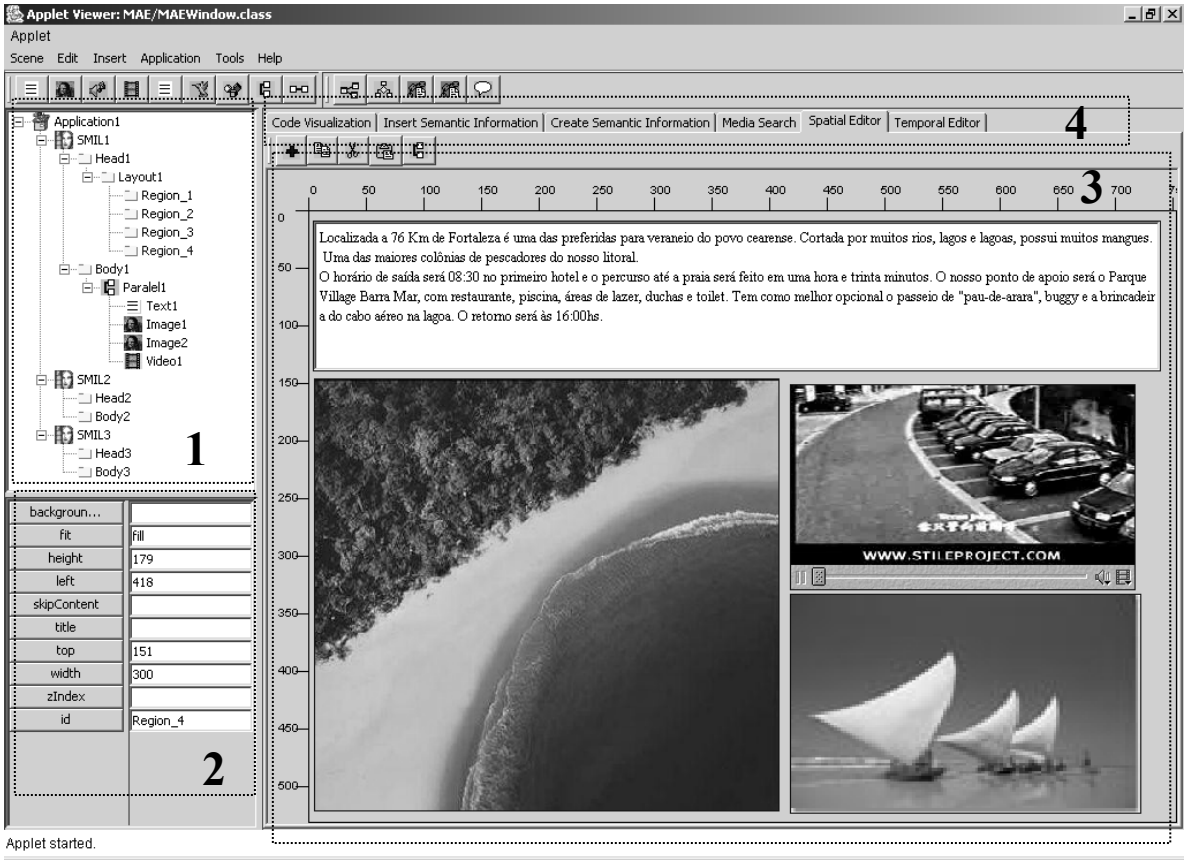


Figura 26 - Interface da Ferramenta MAE

No exemplo apresentado existem três cenas (documentos SMIL) que fazem parte da aplicação, identificadas por SMIL1, SMIL2 e SMIL3. A cena SMIL1 contém quatro regiões (“Region_1”, “Region_2”, “Region_3” e “Region_4”), que são referentes à configuração do layout da cena e são informações contidas no cabeçalho (“Head1”). No corpo (“Body1”) foram inseridas quatro mídias (“Text1”, “Image1”, “Image2”, “Video1”) dentro de um bloco paralelo (“Paralel1”). Os ícones são usados para diferenciar os tipos de objetos. Por exemplo, o ícone representa uma mídia do tipo vídeo.

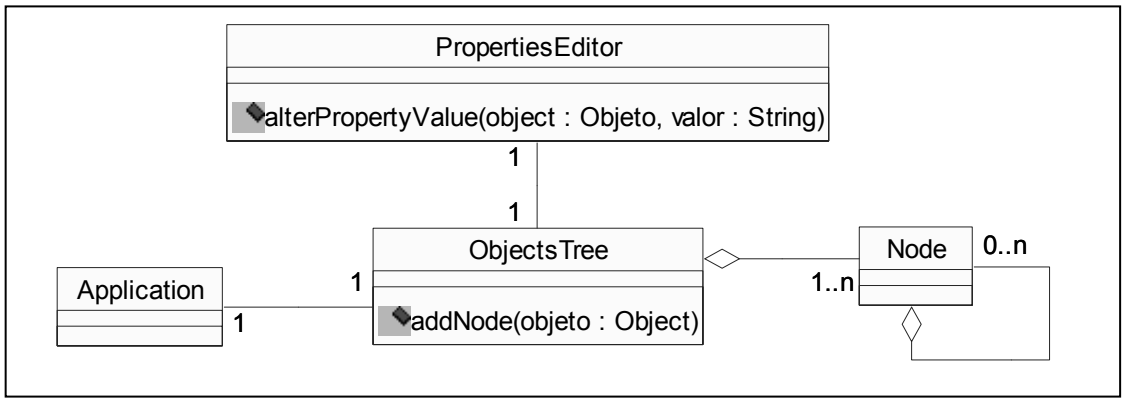


Figura 27 - Classes da Árvore de Objetos e Editor de Propriedades

As principais classes relacionadas à Árvore de Objetos e o Editor de Propriedades são mostradas na Figura 27. A árvore é formada por vários nós, objetos da classe Node, sendo que um nó pode possuir outros nós. O objeto raiz da árvore é sempre um objeto Application. O método addNode(objeto: Object) adiciona os nós na árvore, sendo que é passado o objeto que vai ser adicionado.

5.3.2 Editor de Propriedades

O Editor de Propriedades (área 2 da Figura 26) da ferramenta MAE mostra as propriedades do objeto selecionado através da Árvore de Objetos, do Editor Espacial ou do Editor Temporal. Os valores dessas propriedades também são mostrados e podem ser alterados. As alterações realizadas são refletidas nos outros editores. As propriedades que não são referentes à configuração espacial ou temporal só podem ter seus valores alterados através desse editor ou diretamente no código gerado. O Editor de Propriedades está sempre visível, mostrando o comportamento de cada atributo quando alguma alteração é feita em qualquer um dos módulos.

A figura 27 mostra a associação do Editor de Propriedades com a Árvore de Objetos. O método alterPropertyValue(objeto: Objeto, valor String) é responsável por alterar o valor de um dos atributos de configuração da aplicação. O parâmetro “objeto” recebe o objeto relacionado ao atributo a ser modificado enquanto que “valor” recebe o valor que esse atributo deve receber.

5.3.3 Editor Espacial

As relações espaciais são geralmente conhecidas como os relacionamentos de layout que definem o espaço usado para a apresentação de um objeto de mídia em um dispositivo de

saída, em um certo tempo em uma apresentação multimídia. Se o dispositivo de saída é bidimensional (por exemplo, monitor ou papel), o layout especifica a área bidimensional a ser usada.

A área 3 (da Figura 26) mostra o Editor Espacial, onde é possível realizar, sobre as mídias que fazem parte da aplicação, algumas operações, tais como incluir, apagar, recortar, colar, copiar e alinhar. Nesse editor podem ser criadas várias regiões, e cada região deve ser associada a uma mídia. É através dessas regiões que se determina a localização da mídia na aplicação. Qualquer mídia, exceto áudio, que pertença à aplicação, deve estar associada a uma região e ela será executada na região a qual está associada.

Outro aspecto interessante em relação ao Editor Espacial é a questão da visualização da mídia que será apresentada em cada região, isto é, para o usuário é mais adequado se ao associar uma mídia a uma região ele já possa visualizá-la nessa região. É necessário lembrar que mídias de formatos diferentes podem ser utilizadas. Assim, a visualização de uma imagem é diferente da visualização de um vídeo, e um áudio não possui visualização. Nesse aspecto, para diferenciar entre um vídeo e um áudio associado à uma região basta que o usuário utilize a Árvore de Objetos, já que optou-se por apresentar o vídeo congelado na primeira cena. Além disso, quando se está trabalhando com vídeos, existe uma barra com comandos para que possa ser executado esse vídeo no próprio Editor Espacial.

Para entender como funciona o gerenciamento de regiões na MAE, considere o exemplo da figura 26, onde foram criadas quatro regiões, identificadas por “Region_1”, “Region_2”, “Region_3” e “Region_4” na Árvore de Objetos. Para cada uma delas foram determinados o seu tamanho e posicionamento na cena da aplicação. No momento em que é criada uma região, o usuário deve informar a mídia associada à região criada de forma a tornar possível a sua visualização nessa região. Dessa maneira, depois de feita a associação da mídia com a região, o usuário trabalha diretamente com a visualização da mídia na área do editor. Em todas as regiões onde existirem mídias a elas associadas, se tem a visualização dessas mídias ao mesmo tempo, fornecendo ao autor uma boa idéia de como ficará o *layout* da apresentação.

A Figura 28 apresenta as classes utilizadas pelo Editor Espacial, que é composto por uma régua (Ruler), por uma área de desenho (DrawArea), onde são inseridas as regiões, e por uma barra de ferramentas (TollBar) que é composta por vários botões (Button). A área de desenho pode possuir várias regiões espaciais (SpatialRegion). A área 1 da figura mostra as classes SMIL que estão associadas ao Editor Espacial.

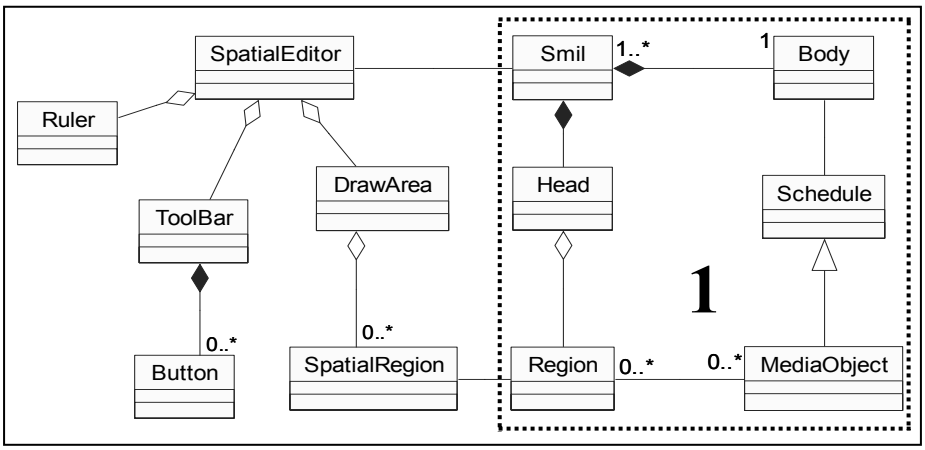


Figura 28 - Classes do Editor Espacial

5.3.4 Editor Textual

Um Editor Textual é usado em ferramentas que trabalham com uma linguagem conhecida para a construção da aplicação. Muitas ferramentas possuem uma linguagem própria e não disponibilizam o código, outras mostram o código mas não permitem alterações e outras apresentam apenas partes do código que podem ser alteradas pelo usuário. Na MAE optou-se por permitir a edição do código por ser útil para usuários que tenham um bom conhecimento da linguagem utilizada.

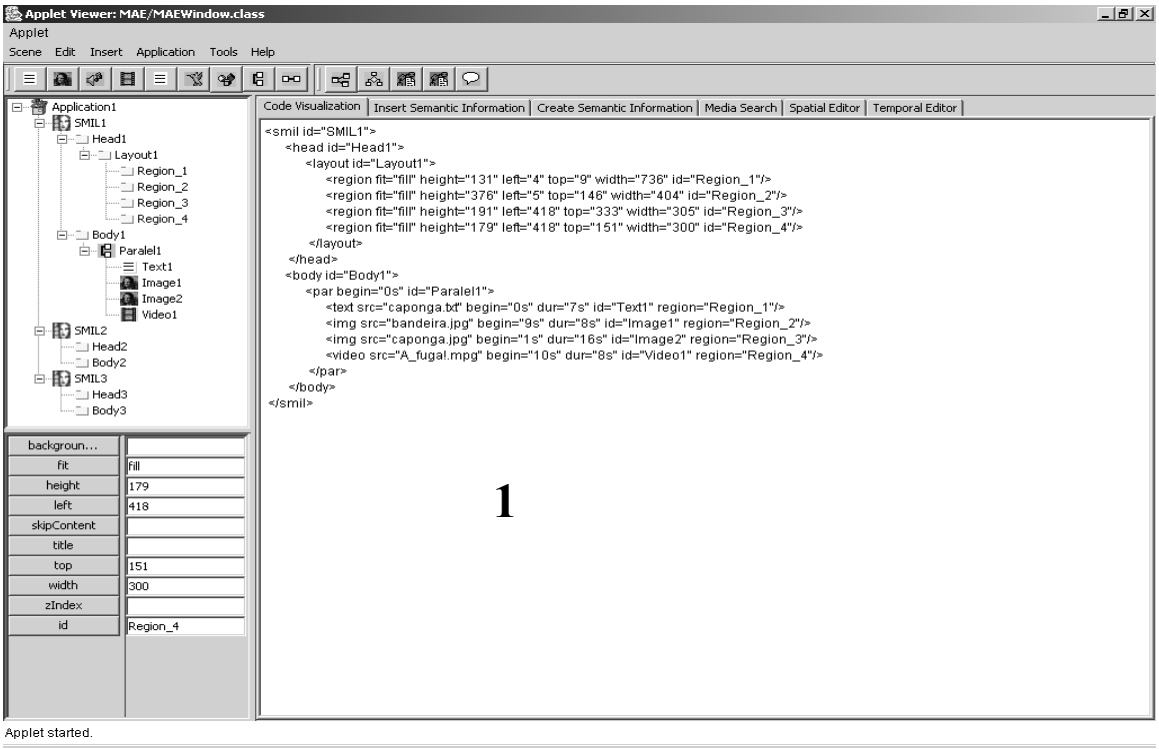


Figura 29 - Editor Textual da ferramenta MAE

A Figura 29 mostra o Editor Textual, indicado pela área 1, o qual apresenta o código SMIL gerado para a aplicação desenvolvida. Através dele é possível visualizar e alterar o código. Esse editor é útil para os autores que tenham familiaridade com a linguagem SMIL, pois permite acesso total ao código e qualquer parte pode ser alterada. A verificação sobre a correção das alterações realizadas é de responsabilidade do usuário.

5.3.5 Editor Temporal

A edição temporal, ou seja, a sincronização das mídias na aplicação, é uma função imprescindível em uma ferramenta de autoria multimídia. Sincronização multimídia tem sido identificada como umas das questões mais importantes em sistemas multimídia [Herzner, 1994], [Steinmetz & Nahrsted, 1995], [Little & Ghafoor, 1995]. Sincronização pode ser definida como a correta ou desejada aparência (apresentação) temporal das mídias. O objetivo da especificação da sincronização temporal é representar, de maneira expressiva e fácil, os relacionamentos temporais entre os objetos de mídias em um sistema multimídia. Um esquema de sincronização define os mecanismos usados para alcançar a sincronização requerida/desejada.

A adoção de um Editor Temporal visual é feita por muitas ferramentas, outras utilizam somente atributos, algumas baseiam as ações em eventos, e algumas utilizam uma combinação de tudo isso. O paradigma utilizado na ferramenta influencia em muito a solução adotada por uma ferramenta para a realização da configuração temporal das mídias.

A linguagem SMIL foi determinante na definição do modelo de edição temporal da MAE. Como SMIL utiliza conceitos de blocos paralelos e seqüenciais, eles tiveram que ser adotados na edição temporal da ferramenta, sendo que se procurou realizar isso sem tornar complicada a edição. Para que o autor tenha uma visão geral do comportamento das mídias na aplicação, foi adotado um editor onde todas as mídias são visíveis ao mesmo tempo, sendo possível perceber, de forma clara e rápida, o comportamento da aplicação.

Buford [Buford, 1994] descreve os possíveis relacionamentos temporais entre dois objetos de mídia. A abordagem utilizada pela ferramenta MAE para a modelagem dessa sincronização cobre todos o possíveis relacionamentos apresentados por Buford e toda a modelagem definida na linguagem SMIL.

A Figura 30 apresenta a notação utilizada no Editor Temporal e os possíveis tipos de sincronização entre mídias. Uma mídia é representada por um retângulo e a sua largura equivale ao tempo de apresentação, enquanto que um bloco também é representado por um retângulo, mas com as bordas mais grossas. No Editor Temporal uma mídia sempre está

dentro de um bloco, sendo que ela pode estar em vários blocos ao mesmo tempo, ou seja, uma mesma mídia pode ser apresentada em momentos diferentes da cena.

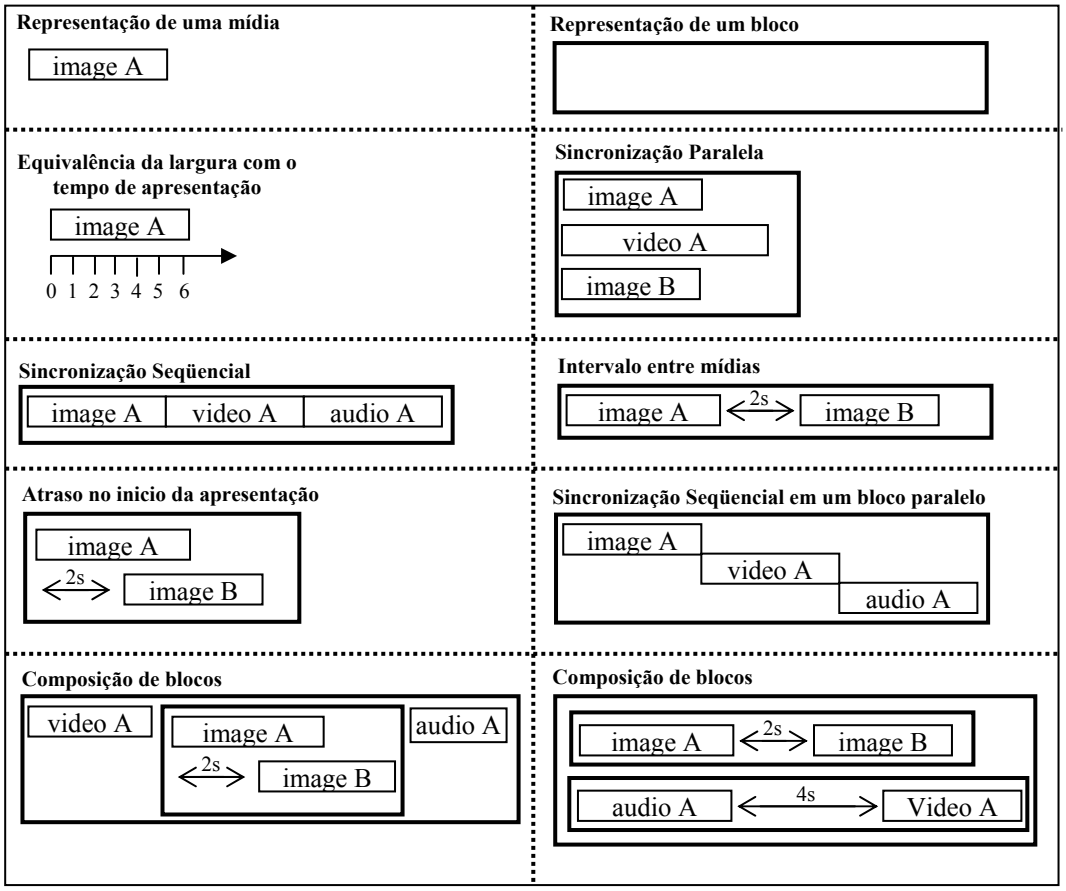


Figura 30 - Notação utilizada pelo Editor Temporal e possíveis sincronizações

Um bloco paralelo se diferencia de um bloco seqüencial somente pelo número de níveis, sendo que um bloco seqüencial tem apenas um nível enquanto que um bloco paralelo tem dois ou mais níveis. Em um bloco seqüencial todas as mídias estão no mesmo nível e só podem se movimentar horizontalmente. Já em um bloco paralelo só existe uma mídia em cada nível e elas também só podem ser movimentadas horizontalmente.

A área 1 da Figura 31 mostra a interface do Editor Temporal criada para a ferramenta MAE, e será usada para exemplificar o funcionamento do modelo de edição temporal.

No exemplo da Figura 31 existem quatro mídias (“caponga.txt”, “bandeira.jpg”, “caponga.jpg” e “A_fuga.mpg”) sendo visualizadas no Editor Temporal. Para a configuração dessas mídias foi criado um bloco paralelo, que possui as quatro mídias, uma em cada nível.

A aplicação inicia com a execução do bloco paralelo, onde a mídia “caponga.txt” inicia no tempo 0s da apresentação, ou seja, a apresentação inicia com essa mídia. No tempo 2s da aplicação é iniciada a execução da mídia “caponga.jpg”. Elas seguem sendo apresentadas simultaneamente até o tempo 8s, quando é encerrada a apresentação da mídia

“caponga.txt”. A mídia “bandeira.jpg” começa a ser executada no tempo 10s, enquanto que “A_fuga.mpg” inicia no tempo 11s. No tempo 18s é encerrada a apresentação de “bandeira.jpg” e “caponga.jpg”, enquanto que no tempo 19s o bloco paralelo é encerrado com o fim da apresentação da mídia “A_fuga.mpg”.

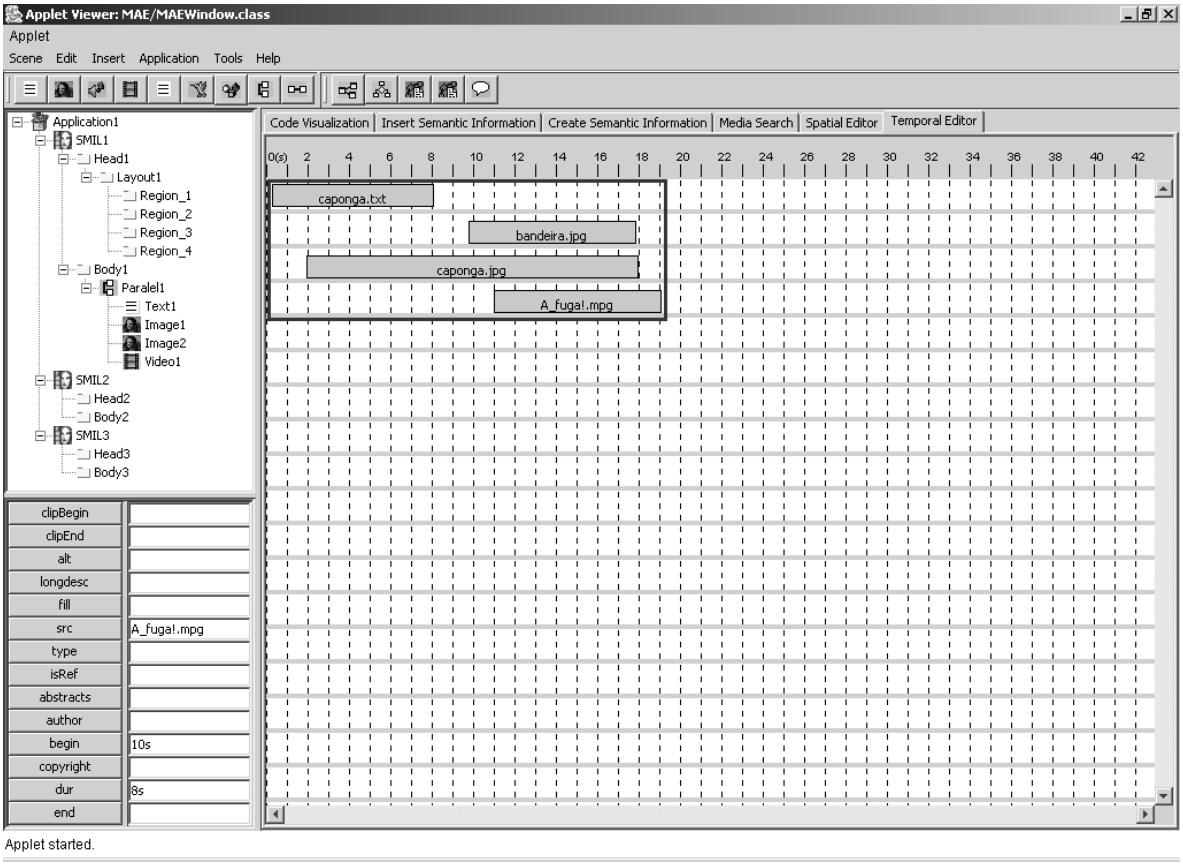


Figura 31 – Editor Temporal da ferramenta MAE

Através do Editor Temporal é possível realizar todas as configurações temporais definidas no padrão SMIL 1.0. A Figura 32 apresenta um outro exemplo, onde dentro de um bloco paralelo existe um outro bloco paralelo.

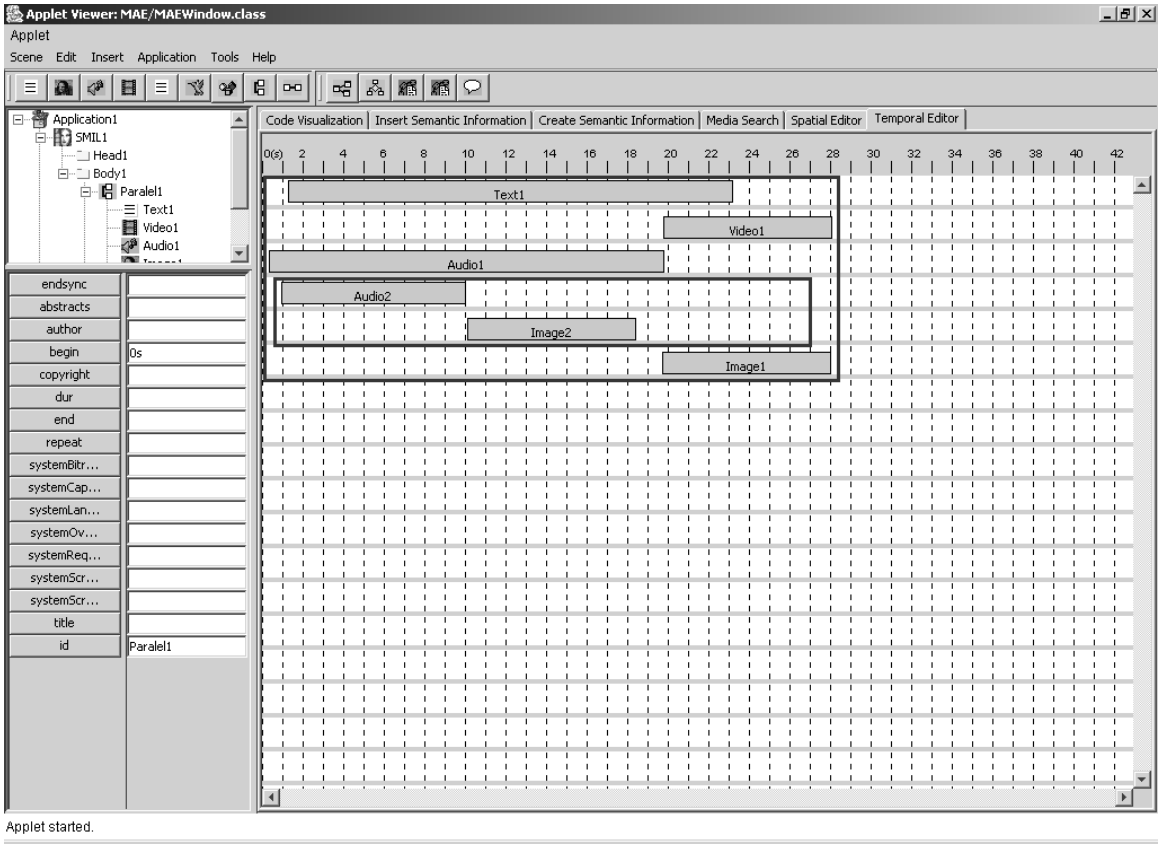


Figura 32 - Editor Temporal da ferramenta MAE

A Figura 33 mostra as classes utilizadas pelo Editor Temporal. A área tracejada, indicada pelo número 1, apresenta as classes usadas para a sua composição, enquanto as outras classes são referentes ao padrão SMIL.

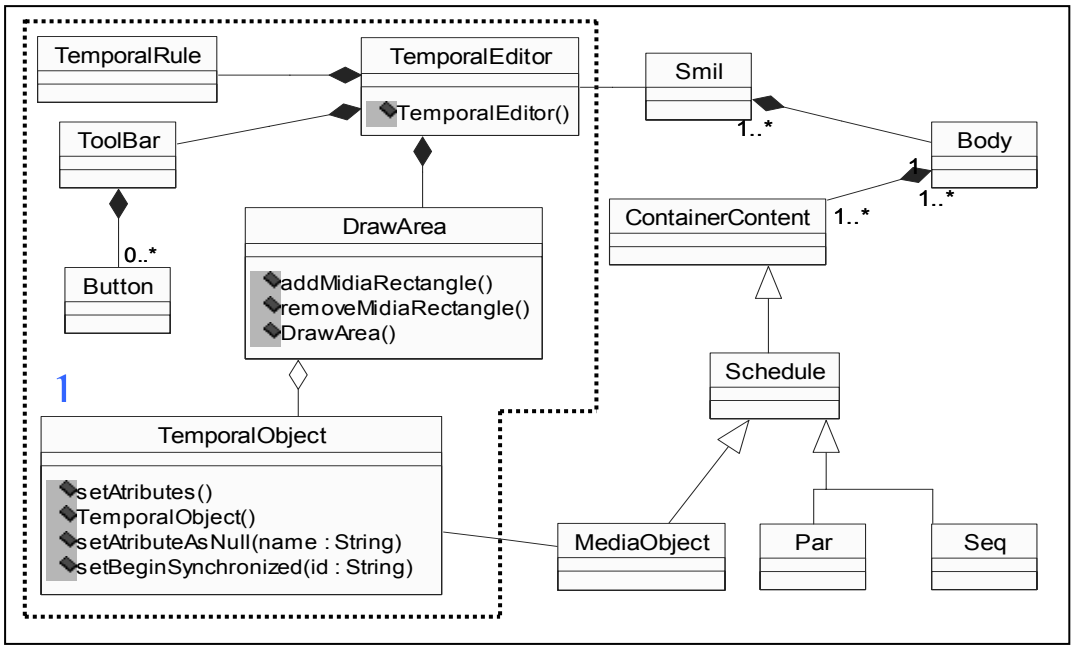


Figura 33 - Classes utilizadas no Editor Temporal

Uma das partes que compõem o Editor Temporal é uma área de desenho (*DrawArea*), que é a área sobre a qual são movimentados os objetos que representam as mídias. Essa classe possui, além do construtor, os seguintes métodos:

- addMidiaRectangle( ): adiciona na área de desenho um objeto que representa uma mídia
- removeMidiaRectangle( ): remove da área de desenho um objeto que representa uma mídia

Os objetos que representam as mídias são da classe *TemporalObject*. Cada objeto criado dessa classe está associado a uma mídia inserida na aplicação e as alterações realizadas no objeto refletem nas configurações da mídia. Além do construtor, essa classe possui os seguintes métodos:

- setAtributes(): realiza a configuração dos atributos temporais da mídia associada com o objeto, de acordo com as alterações realizadas sobre o objeto
- setAtributeAsNull(name: String): configura com "null" o atributo passado como parâmetro. Com isso, a configuração feita anteriormente sobre esse parâmetro é desfeita e ele passa a não ter mais influência na configuração temporal da mídia
- setBeginSynchronized(id: String): configura o inicio de uma mídia sincronizado com o final de outra mídia, identificada pelo parâmetro "id"

As outras partes que compõem o Editor Temporal são: uma régua (*TemporalRule*) que contém as marcações temporais e uma barra de ferramentas (*ToolBar*), que é composta por alguns botões (*Button*).

5.3.6 Inserção de informação semântica

Além dos componentes que fornecem as funcionalidades para edição da aplicação, a ferramenta MAE também apresenta ao usuário uma interface que permite definir as informações semânticas sobre as mídias que foram inseridas na aplicação. Conforme descrito no capítulo 4, as aplicações criadas são armazenadas em uma base de dados que armazena também informações semânticas sobre as mídias que compõem as aplicações (Figura 21). Essa informação é utilizada em consultas para a recuperação de mídias através do seu conteúdo semântico. A próxima seção explica como é utilizada essa informação na recuperação das mídias.

O usuário pode inserir essas informações semânticas, através das interfaces das Figuras 34 e 40 da ferramenta MAE. A Figura 34 apresenta a interface para definição da informação semântica, enquanto que a Figura 40 apresenta a interface para a criação de novos dados para definição da informação semântica.

5.3.6.1 Definição de informação semântica

Na área 1 (em destaque na Figura 35 (a)) da Figura 34 é feita a seleção da mídia sobre a qual será inserida a informação semântica. Pode ser selecionada qualquer mídia que foi inserida na aplicação e, depois de selecionada, ela é então visualizada. No exemplo mostrado, foi selecionada a mídia (uma imagem) "caponga.jpg".

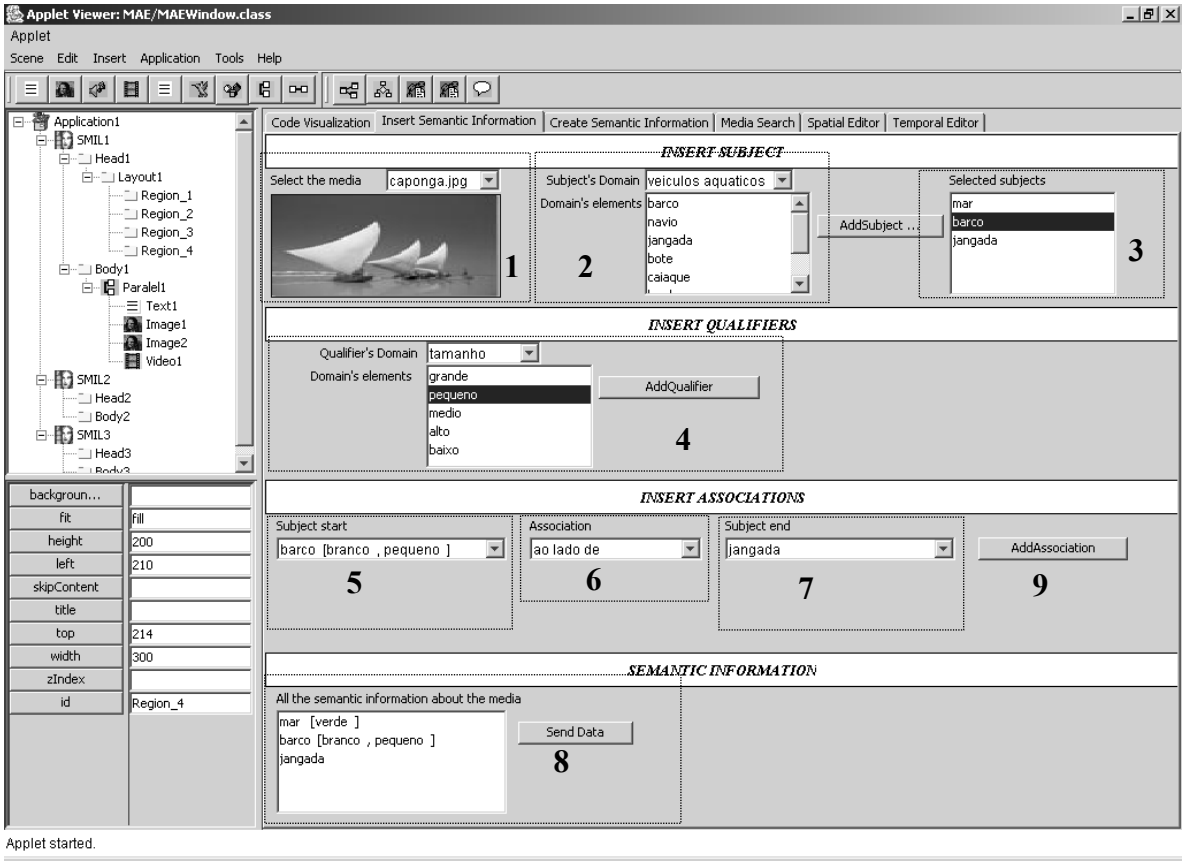


Figura 34 – Interface para inserção de informação semântica

Depois de selecionada a mídia, é inserida a informação semântica sobre ela. Os dados apresentados pela ferramenta são aqueles definidos na estrutura da base de dados de informação semântica.

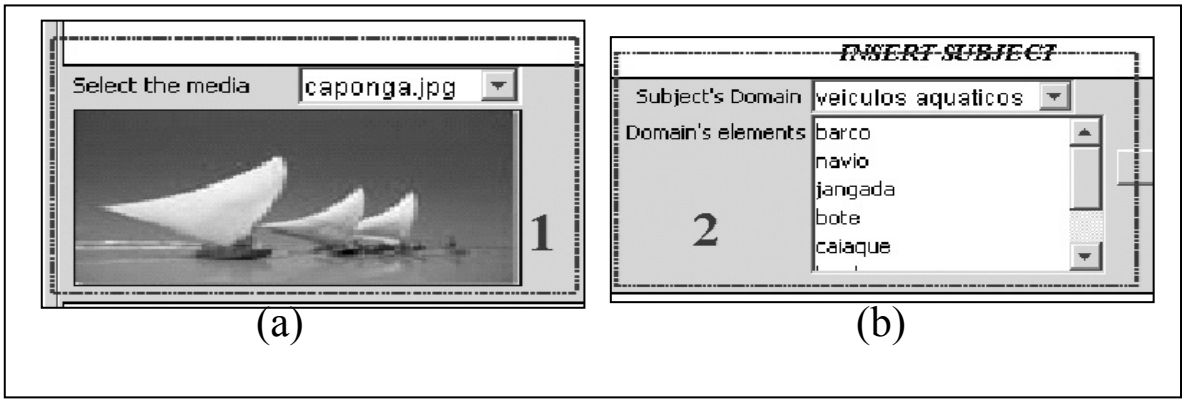


Figura 35 - Destaque das áreas 1 e 2 da figura 34

Na área 2 (em destaque na Figura 35(b)) são selecionados os temas que aparecem na mídia, sendo que eles são agrupados em domínios semânticos. Uma vez selecionado o domínio, os elementos pertencentes a ele são apresentados em uma lista (área 3, em destaque na Figura 36(a)). No exemplo da Figura 34, foi selecionado o domínio "veículos aquáticos" e os elementos "barco", "navio", "jangada", "bote" e "caiaque" foram apresentados. Uma vez escolhido um tema, através do botão "AddSubject" ele é inserido na informação semântica da mídia. No exemplo apresentado foram inseridos os temas "mar" "barco" e "jangada".

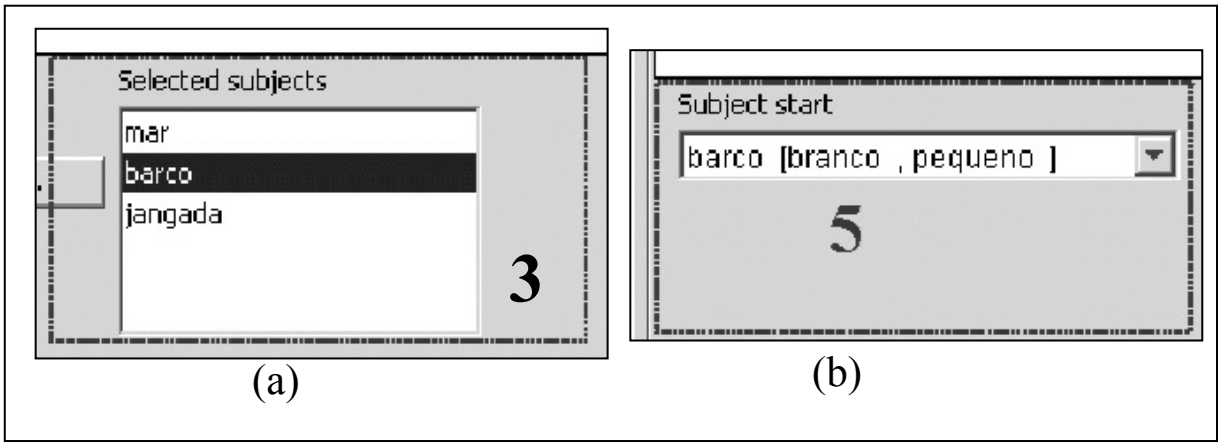


Figura 36 - Destaque das áreas 3 e 5 da figura 34

Podem ser inseridos qualificadores para os temas através dos componentes apresentados na área 4 (em destaque na Figura 37). Os qualificadores também são agrupados em domínios semânticos e a seleção é feita da mesma forma que os temas: é escolhido um domínio e os elementos pertencentes a ele são apresentados. No exemplo da Figura 34, foi selecionado o domínio "tamanho", sendo apresentados os elementos "grande", "pequeno", "médio", "alto" e "baixo". Para inserir um qualificador em um dos temas, é preciso selecionar o qualificador e o tema ao qual ele será associado e então pressionar o botão "AddQualifier".

Nesse exemplo, foram selecionados o qualificador “pequeno” e o tema “barco” (em destaque nas respectivas listas).

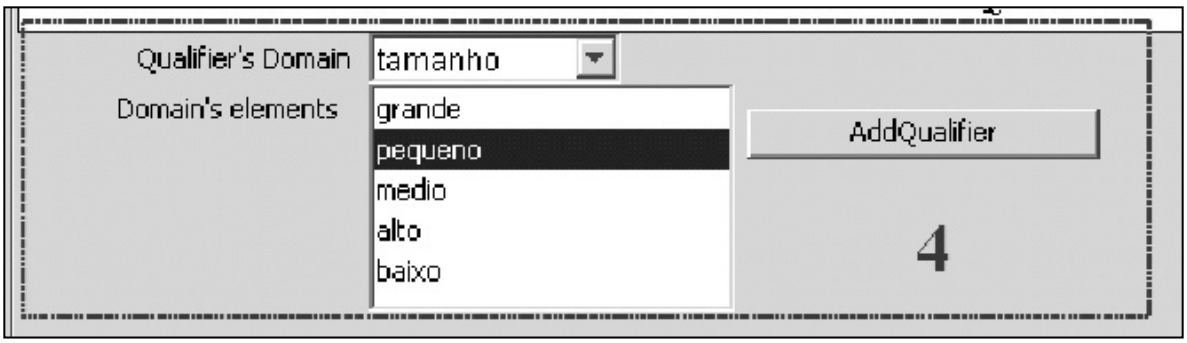


Figura 37- Destaque da área 4 da figura 34

A informação semântica que está sendo inserida é apresentada na área 8 (em destaque na Figura 39) da Figura 34. Existe uma lista com os temas inseridos e seus respectivos qualificadores, sendo que os qualificadores são apresentados entre colchetes. No exemplo apresentado, foi inserido o tema “mar” com o qualificador “verde”, o tema “barco” com os qualificadores “branco” e “pequeno”, e o tema “jangada” sem qualificadores.

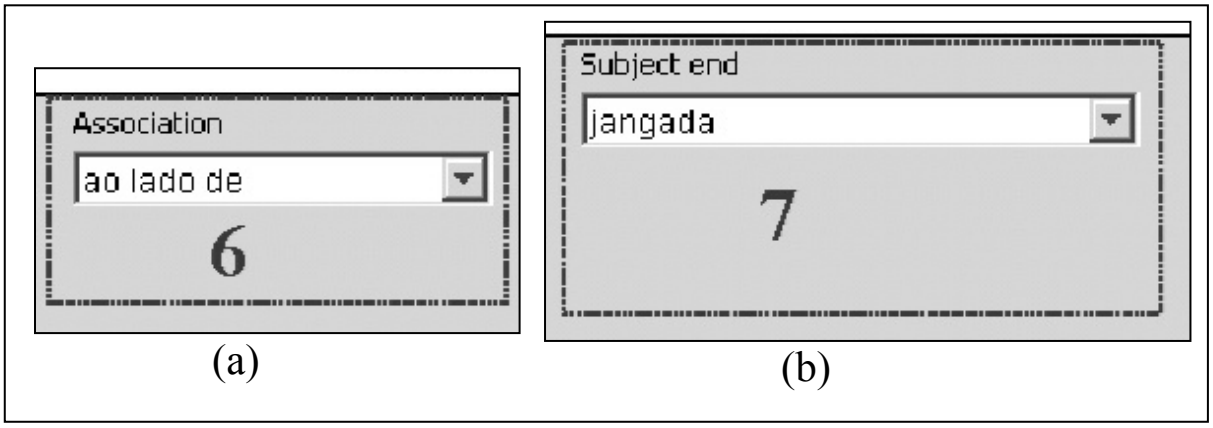


Figura 38 - Desataque das areas 6 e 7 da Figura 34

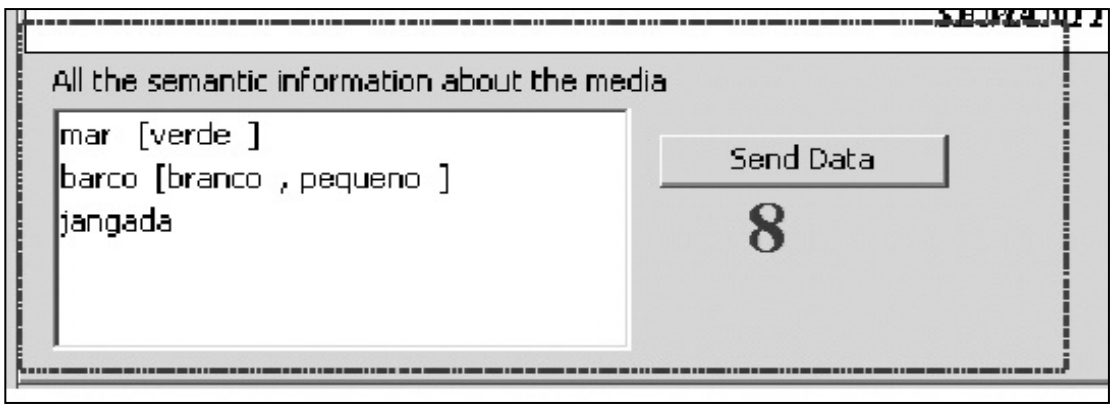


Figura 39- Destaque da área 8 da figura 34

Além de temas e qualificadores, é possível também indicar as associações que ocorrem entre os temas. Essas associações são do tipo "barco *em frente a* mar", onde "barco" é o tema inicial, "em frente a" é a associação e "mar" é o tema final. Para inserir esse tipo de informação através da ferramenta, primeiro é selecionado o tema inicial através do componente indicado na área 5 (em destaque na Figura 36(b)) da figura, depois é selecionada a associação desejada através do componente indicado na área 6 (em destaque na Figura 38(a)) e então é selecionado o tema final através do componente indicado pela área 7 (em destaque na Figura 38(b)). No exemplo da Figura 34, foi selecionado "barco branco, pequeno" como tema inicial, "ao lado de" como a associação e "jangada" como tema final. Através do botão "AddAssociation" é inserida na informação semântica essa associação entre os temas.

5.3.6.2 Criação de nova informação semântica

Quando não existe, entre os dados já cadastrados, a informação necessária para a definição da informação semântica, o usuário pode criar novos dados que atendam as suas necessidades. A criação desses dados é feita através da interface apresentada na Figura 40.

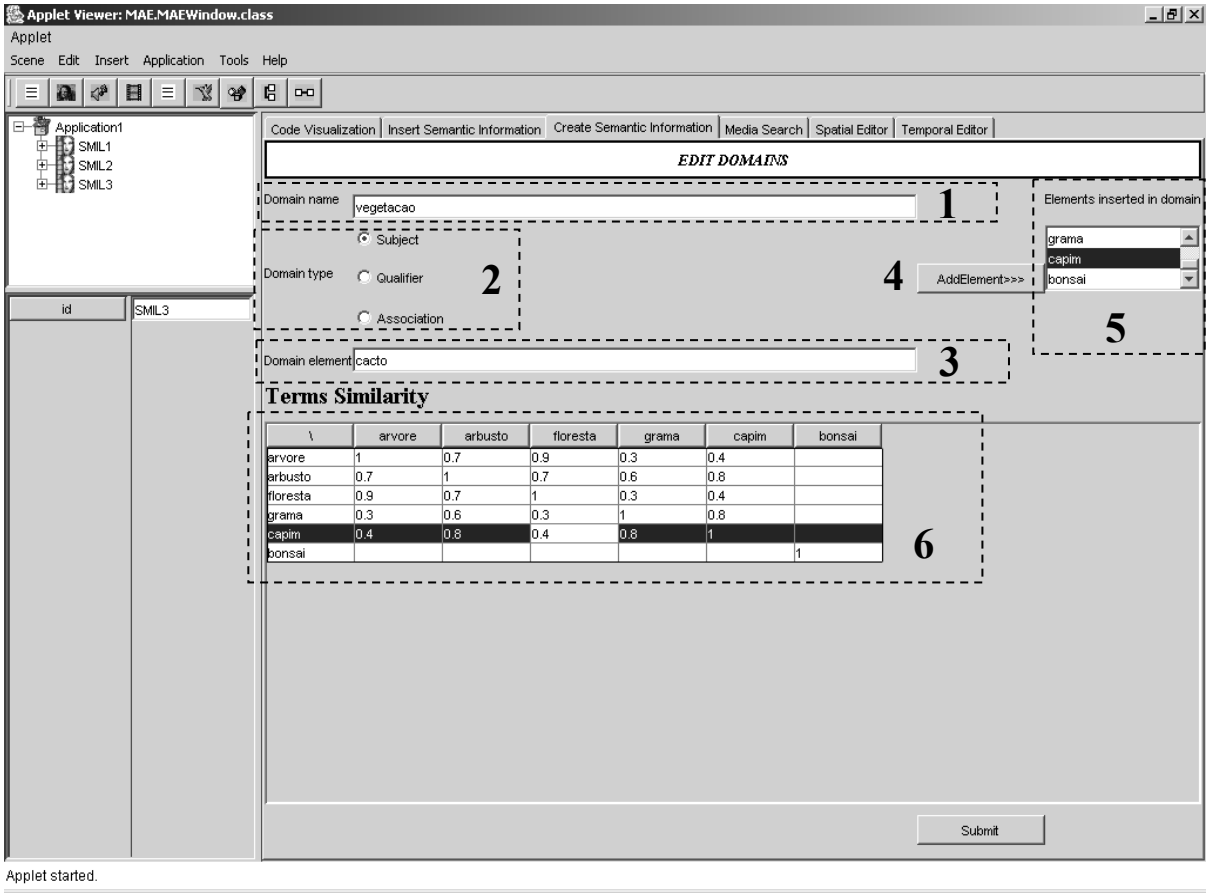


Figura 40 – Interface para criação de dados

Na área 1 (em destaque na Figura 41) é definido o nome do domínio a ser criado. No exemplo, está sendo criado o domínio "vegetação". Além do nome do domínio, deve ser indicado também qual o tipo do domínio, ou seja, definir qual o papel que os elementos desse domínio têm na definição da informação semântica. Os domínios podem ser de temas, de qualificadores ou de associações. A indicação do tipo do domínio é feita através dos componentes indicados na área 2 (em destaque na Figura 42 (a)).

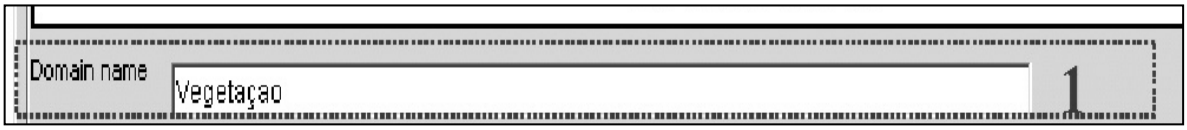


Figura 41 – Destaque da área 1 da Figura 40

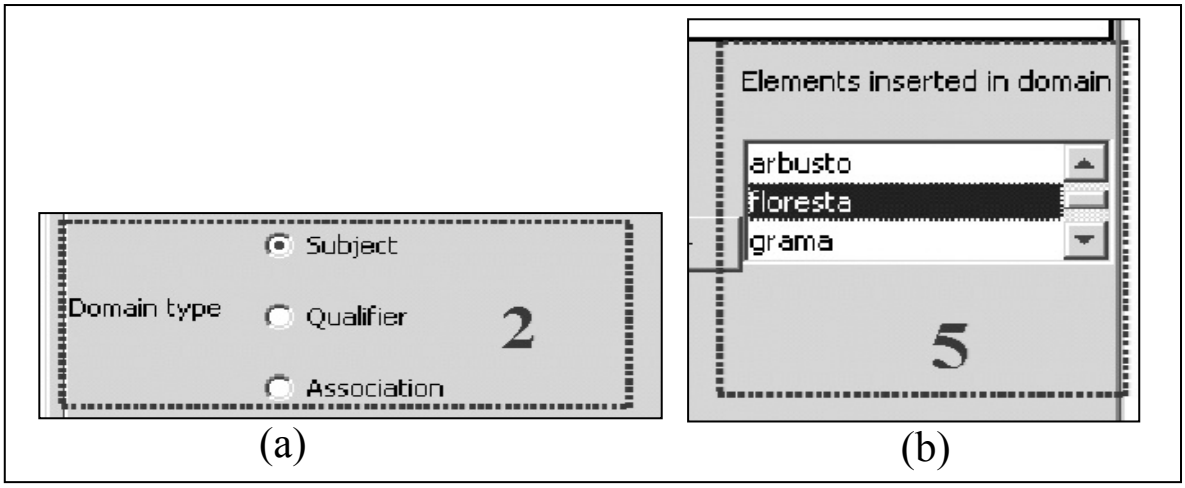


Figura 42 – Destaque das áreas 2 e 5 da Figura 40

A área 3 (em destaque na Figura 43) indica a região onde são definidos os elementos componentes do domínio. No exemplo, está sendo inserido o elemento "cacto" no domínio. A inserção desse novo elemento no domínio é feita através do botão "AddElement", indicado pelo número 4 da Figura 40. Todos os elementos inseridos no domínio são apresentados em uma lista, indicada na área 5 (em destaque na Figura 42 (b)).

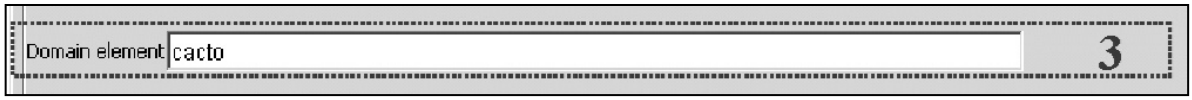
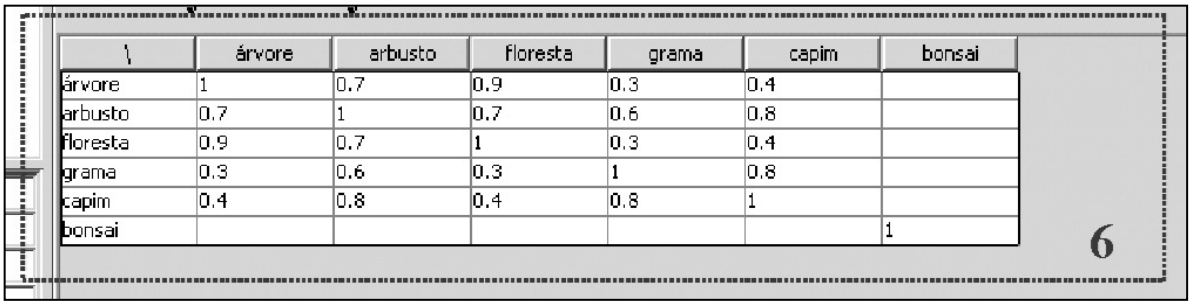


Figura 43 - Destaque da área 3 de Figura 40



\	árvore	arbusto	floresta	grama	capim	bonsai
árvore	1	0.7	0.9	0.3	0.4	
arbusto	0.7	1	0.7	0.6	0.8	
floresta	0.9	0.7	1	0.3	0.4	
grama	0.3	0.6	0.3	1	0.8	
capim	0.4	0.8	0.4	0.8	1	
bonsai						1

Figura 44 – Destaque da área 6 da Figura 40

A área 6 (em destaque na Figura 44) mostra a tabela de similaridade(proximidade) dos elementos inseridos no domínio. Quando um elemento é inserido no domínio, automaticamente ele é inserido nessa tabela e o valor 1 é atribuído à célula que contém o mesmo elemento tanto na linha quanto na coluna. No exemplo da figura, o elemento "bonsai" foi o último a ser inserido e ainda não foram inseridos os valores de similaridade entre ele e os outros elementos. Note que o elemento "cacto", definido na área 3, ainda não foi inserido no domínio.

5.3.7 Consultas e recuperação de mídias

Para dar mais recursos ao autor na criação de uma aplicação, ele pode consultar a base de dados apresentada BDMm e utilizar as mídias já armazenadas na criação de uma nova aplicação.

A informação semântica inserida sobre as mídias tem por objetivo permitir que sejam realizadas consultas, baseadas no conteúdo semântico, para a recuperação de mídias. Um exemplo de consulta seria recuperar os vídeos que contêm mar. Outro exemplo seria recuperar as imagens que contém lago com pássaro branco.

O usuário, ao formular uma consulta, pode informar os temas desejados, os qualificadores para cada um dos temas, e as associações entre os temas. Adicionalmente, pode informar a similaridade mínima desejada para cada tema e qualificador. Além disso, se desejar, pode definir qual é a importância de cada tema e qualificador para a consulta, através da definição de um grau de relevância para eles. Se o usuário não estabelecer graus de similaridade, a consulta tem o comportamento de uma busca exata (similaridade mínima igual a 100%).

Essas consultas são feitas através da interface apresentada na Figura 45, onde são especificados o tipo e o conteúdo semântico da mídia que se deseja recuperar. Essa interface é semelhante a apresentada na Figura 34, uma vez que as mesmas informações inseridas sobre a mídia, são utilizadas para a recuperação.

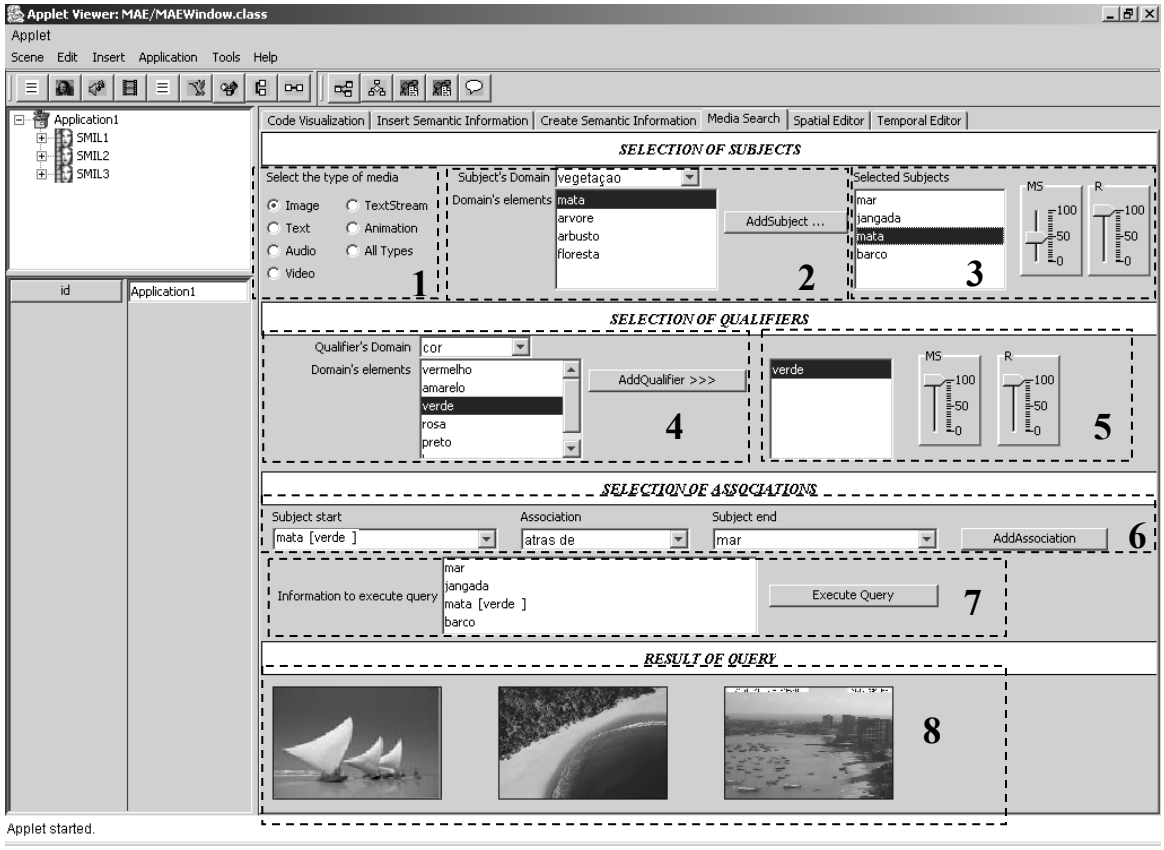


Figura 45 - Interface para consultas

Na área 1 (em destaque na Figura 46) da Figura 45 é feita a escolha do tipo de mídia que se deseja recuperar; pode ser determinado um tipo específico (áudio, vídeo etc), uma combinação de tipos (por exemplo, imagem e vídeo) ou todos os tipos. No exemplo apresentado foi selecionado o tipo *image*. Na área 2 (em destaque na Figura 47) são selecionados os temas usados na busca; a escolha do tema é feita de forma idêntica a que foi apresentada na Figura 34.

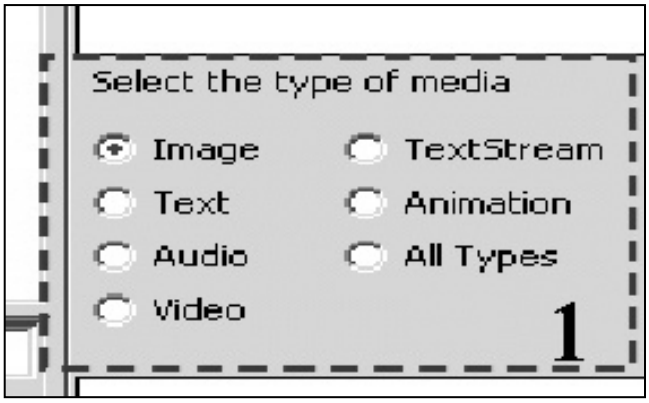


Figura 46 - Destaque da área 1 da figura 45

Figura 47 - Destaque da área 2 da figura 45

Na área 3 (em destaque na Figura 48) são mostrados os temas que foram selecionados e para cada tema é possível determinar a similaridade mínima (MS) e a relevância (R) utilizadas na busca nebulosa. Nesse exemplo, foram determinadas similaridade mínima de 50% e relevância de 100% para o tema "*mata"*.

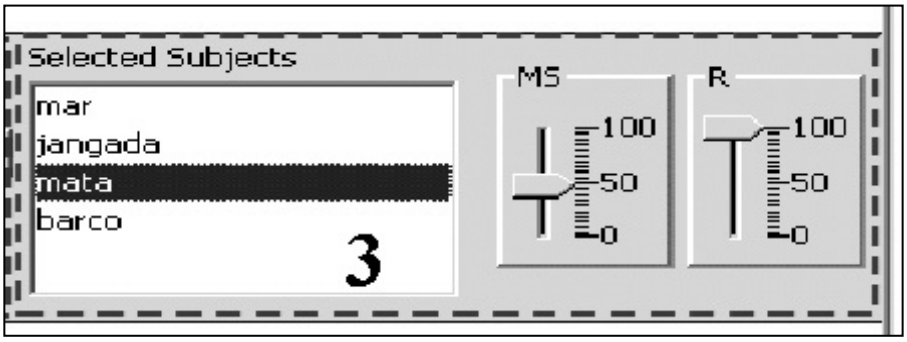


Figura 48 - Destaque da área 3 da figura 45

Na área 4 (em destaque na Figura 49) são selecionados os qualificadores para os temas selecionados na área 3. Essa seleção é feita de forma idêntica aos temas, já apresentada na Figura 34. Na área 5 (em destaque na Figura 50) são mostrados os qualificadores que foram selecionados e para cada um é possível determinar a similaridade mínima (MS) e a

relevância (R) utilizadas nas buscas nebulosas. Nesse exemplo, foi determinada similaridade mínima de 100% e relevância de 100% para o qualificador "*verde*".

Figura 49- Destaque da área 4 da Figura 45

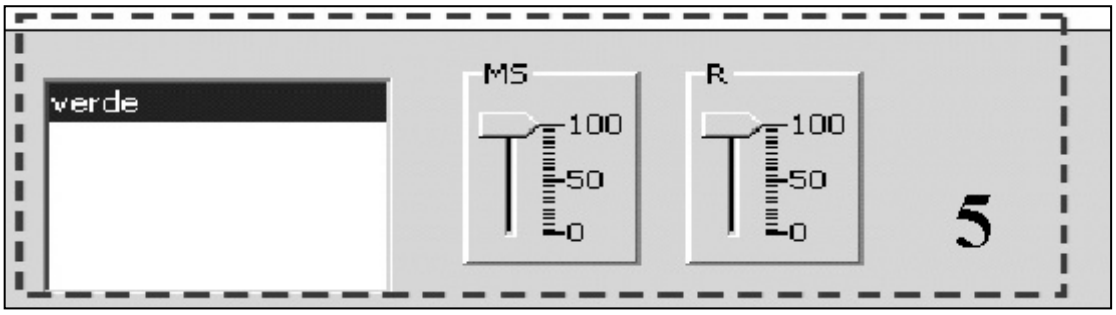


Figura 50 - Destaque da área 5 da figura 45

A área 6 (em destaque na Figura 51) é utilizada pra definir as associações entre os temas especificados na consulta, onde se define o tema inicial (*Subject start*), o tema final (*Subject end*) e a associação entre eles (*Association*). A área 7 (em destaque na Figura 52) mostra toda a informação usada na consulta às mídias.

Figura 51- Destaque da área 6 da figura 45

Figura 52- Destaque da área 7 da figura 45

Apesar de ser possível informar a similaridade mínima e a relevâncias dos elementos utilizados nas consultas, esses parâmetros ainda não são utilizados na execução dessas consultas, pois ainda não foi incorporado o módulo de consultas nebulosas. Esse módulo foi desenvolvido no trabalho de Borges [Borges, 2001], usando o SGBDOO Jasmine [Jasmine, 2000]. Esse módulo ainda não está disponível no Caché. Portanto, no momento, a ferramenta está disponibilizando apenas consultas por conteúdo exatas. O resultado das consultas incluirá todas as mídias que apresentem pelo menos um dos temas selecionados (apresentados na área 7). No exemplo apresentado foram recuperadas as mídias que continham "mar" ou "jangada" ou "mata verde" ou "barco". A área 8 (em destaque na Figura 53) apresenta as mídias que foram recuperadas após a realização da consulta.

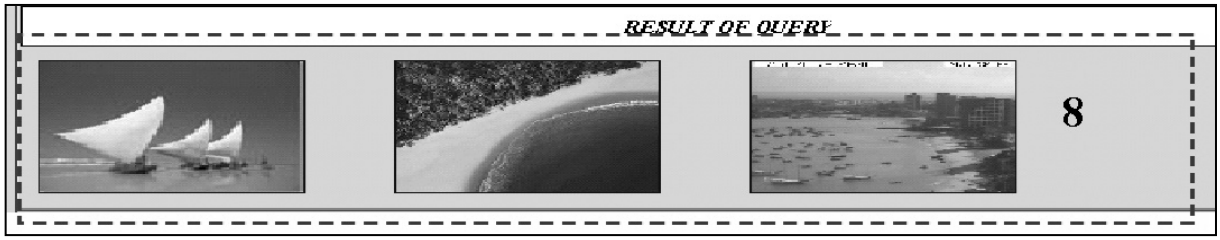


Figura 53 - Destaque da área 8 da figura 45

5.3.8 Parser

Na ferramenta MAE é utilizado um parser XML (JAXP) [Sun, 2000] que faz a leitura de arquivo textual baseado em SMIL (Figura 54) e cria objetos SMIL que serão manipulados pelo ambiente. Dessa forma a ferramenta pode atuar sobre os objetos criados e alterar as suas configurações.

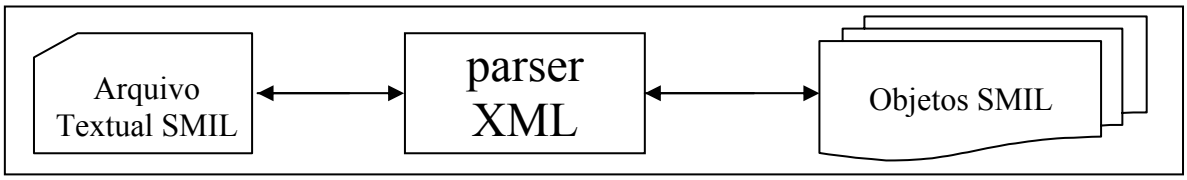


Figura 54 – Forma de atuação do parser

No processo inverso, o parser gera arquivo textual a partir de objetos SMIL. Esse processo inverso é realizado enquanto o usuário cria uma aplicação, que envolve a

manipulação interna dos objetos SMIL. Esse texto gerado pelo parser é apresentado no Editor Textual.

No contexto do ambiente existem algumas alterações a serem feitas no processo de transformação de uma aplicação multimídia, uma vez que ela é composta por vários documentos SMIL. Dessa forma o Parser MAE (Parser XML combinado com o mapeador MAE) deve realizar alguns passos a mais para realizar a conversão de uma aplicação multimídia, sendo que isso é feito pelo mapeador MAE. Como a aplicação pode ser formada por vários arquivos SMIL, todos devem ser lidos e convertidos em objetos SMIL. Esse processo é mostrado na Figura 55. Os Objetos SMIL apresentados na figura abaixo são aqueles apresentados na Figura 25 (página 35) que são manipulados pela ferramenta MAE para a criação e edição de aplicações.

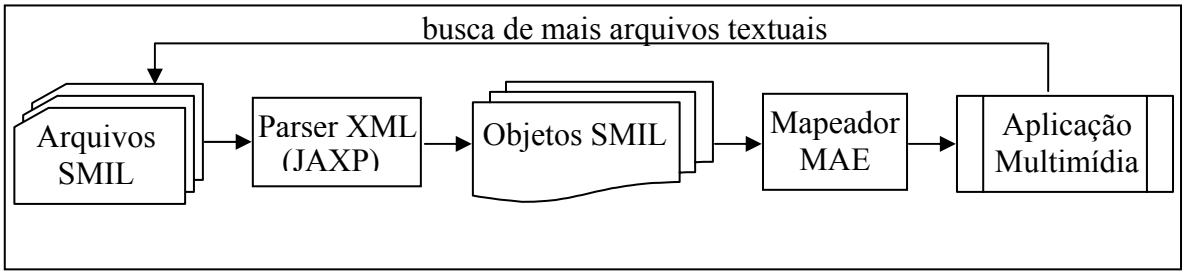


Figura 55 - Processo de leitura dos arquivos textuais

No processo inverso, o Parser MAE (Parser XML combinado com o mapeador MAE) gera arquivos SMIL referentes às aplicações criadas através da MAE, sendo que é necessário inserir informações indicando a quais aplicações pertencem os arquivos gerados. Dessa maneira é possível saber quais arquivos compõem uma aplicação.

5.4 Comparação com outras ferramentas

A ferramenta MAE possui algumas funcionalidades comuns às de outras ferramentas de autoria multimídia, como por exemplo a Director [Director, 2001], a Fluition [Fluition, 2001] e a GRiNS [Grins, 2002]. A Fluition e a GRiNS, assim como a MAE, utilizam a linguagem SMIL e a Director é uma ferramenta bastante conhecida e poderosa.

A seguir é feito um comparativo da MAE com essas ferramentas para mostrar as funcionalidades e soluções diferenciais adotadas na MAE que foram implementadas na tentativa de torná-la mais amigável e poderosa. Como já foi mencionado no capítulo 3, nenhuma delas apresenta facilidades para o reuso de aplicações e também não estão associadas a um banco de dados.

5.4.1 Árvore de objetos

Outras ferramentas de autoria multimídia também possuem uma janela que contém todos os objetos utilizados na criação de uma aplicação. Na ferramenta Director existe uma janela onde podem ser visualizadas todas as mídias inseridas na aplicação. Essa janela está sempre visível e apresenta o nome e o tipo da mídia, além de uma versão miniatura quando possível. Não existe uma árvore hierárquica, pois ela não trabalha com esse tipo de representação para a construção de uma aplicação. A ferramenta Fluition também possui uma janela onde estão as informações sobre as mídias inseridas na aplicação. Ela também não trabalha com uma árvore hierárquica, mas com uma lista de atributos indicando o nome, o tipo e a região a qual tal mídia está associada. Sem uma árvore fica mais difícil identificar todas as mídias e as suas associações com as regiões.

Já a ferramenta GRiNS, assim como a MAE, possui uma árvore de objetos, mas que somente é apresentada quando o Editor Espacial está ativo. Inicialmente todas as mídias inseridas na aplicação estão no mesmo nível, dentro da aplicação, agrupadas pelo tipo da mídia. Diferentes tipos de ícones identificam os tipos das mídias. Apesar da ferramenta trabalhar com o padrão SMIL, não existe a indicação das regiões criadas e quais mídias estão em quais regiões. Se o autor desejar, é possível criar regiões e associar mídias a elas, sendo que isso é refletido na árvore. Dessa maneira, pode haver mídias inseridas em regiões e mídias sem regiões na árvore de objetos, o que pode confundir um pouco o autor.

Resumindo, tanto a Director como a Fluition não possuem uma árvore de objeto. Estas têm propostas diferentes e não necessitam de uma árvore. Já a Grins trabalha com uma árvore mostrando as regiões e mídias inseridas na aplicação, mas não tem um tratamento uniforme na árvore. Nenhuma dessas ferramentas citadas trabalha com o conceito de cenas para a composição de uma aplicação, portanto não precisaram tratar esse tipo de questão. Esse conceito da utilização de cenas, utilizado pela MAE, torna importante a utilização de uma árvore para a compreensão da estrutura da aplicação.

5.4.2 Editor de Propriedades

Não são todas as ferramentas que possuem um único Editor de Propriedades; algumas vezes a configuração das propriedades pode estar distribuída em duas ou mais janelas. Na Director existe um único editor que, dependendo do tipo de objeto selecionado, apresenta as opções que podem ser configuradas para tal objeto. Esse editor fica visível durante todo o tempo, sendo que o usuário pode fechá-lo se desejar. A GRiNS também possui um Editor de Propriedades que é ativado através de um duplo clique no objeto desejado. Já a Fluition divide

a configuração das propriedades dos objetos, sendo que existe uma janela onde são configuradas as regiões e outras onde são configuradas as mídias. A vantagem do editor da MAE é que ele está sempre visível e todos os atributos do objeto selecionado podem ser configurados através dele.

5.4.3 Editor Espacial

A maioria das ferramentas trabalha com a própria mídia no Editor Espacial, ou seja, a mídia é visualizada na tela do editor. Essa é a forma mais clara de posicionar e dimensionar um objeto de mídia na aplicação criada, dando uma visão ao autor de como será o *layout* da apresentação em desenvolvimento. Esse módulo é um dos mais importantes em uma ferramenta de autoria multimídia, uma vez que em uma ferramenta de desenvolvimento de aplicações visuais é através dele que o autor configura a aparência da aplicação. A ferramenta Director trabalha dessa forma: todas as mídias são visíveis na janela do editor, na posição e tamanho que serão apresentados.

Essa abordagem é mais complexa quando se trabalha com a linguagem SMIL, uma vez que é através de regiões que são posicionadas e dimensionadas as mídias na aplicação e uma região também pode ter uma cor de fundo e ter mais de uma mídia associada a ela. A ferramenta GRiNS utiliza a linguagem SMIL, mas trabalha com regiões somente se o usuário desejar criá-las. Dessa maneira ela trabalha de duas formas no Editor Espacial, o que torna mais confuso o seu uso. Além disso, as mídias não são todas visíveis na área do editor ao mesmo tempo; somente aquela selecionada na árvore de objetos se torna visível. Isso dificulta bastante para o usuário perceber o *layout* da aplicação. Quando são criadas regiões, essas sim ficam visíveis o tempo todo, mas não mostram as mídias nelas inseridas.

A ferramenta Fluition também trabalha com regiões e as mídias são visualizadas na região a qual está associada, mas somente é possível visualizar a mídia da região selecionada e não as mídias de todas as regiões ao mesmo tempo. Isso dificulta a visualização do *layout* da aplicação.

A ferramenta MAE trabalha com a criação de regiões, mas adota uma solução onde se tenta chegar o mais próximo possível da utilização direta das mídias. As mídias são visualizadas na região a qual está associada e todas as mídias são visíveis ao mesmo tempo no Editor Espacial de uma cena da aplicação.

5.4.4 Editor Textual

Não são todas as ferramentas que disponibilizam um editor textual; muitas vezes apenas algumas partes do código são editáveis e outras vezes o código pode ser visualizado, mas não alterado. A ferramenta Director utiliza uma linguagem chamada Lingo para criar recursos mais avançados, sendo que ela não necessariamente precisa ser utilizada. A edição do código só é feita em partes específicas. Não é disponibilizado o código todo da aplicação criada.

Nas ferramentas Fluition e GRiNS é possível visualizar todo o código gerado para a construção da aplicação, mas apenas na Fluition é possível editá-lo diretamente. Na MAE o código é visível ao autor e ele pode ser editado.

5.4.5 Editor Temporal

A ferramenta GRiNS, que também utiliza SMIL, trabalha de forma semelhante à MAE com relação ao Editor Temporal, mas utiliza cores diferentes para diferenciar blocos paralelos e seqüenciais, o que torna a interface bastante confusa quando estão sendo usados muitos blocos. Além disso, coloca indicações quando o tempo de uma mídia ultrapassa a área de visualização e outras informações que poluem muito o editor. Com tudo isso o Editor Temporal se torna um pouco confuso, difícil de utilizar e de perceber o comportamento geral da aplicação.

Já a ferramenta Fluition, outra que trabalha com SMIL, não utiliza um editor gráfico para realizar todas as configurações temporais. Ela utiliza o conceito de grupos paralelos e seqüenciais, sendo que os grupos são sempre seqüenciais entre si. Toda essa configuração temporal é feita através de atributos, sem um ambiente visual, o que torna bastante complicada a realização da edição, também não fornecendo uma maneira fácil de perceber o comportamento geral da aplicação.

A Tabela 1 apresenta uma comparação entre as ferramentas apresentadas. As características usadas nessa comparação são aquelas consideradas mais relevantes para este trabalho.

	Visual Class	Director	GRiNS	Fluition	MAE
Editor Temporal	Poucos recursos. Não possui um Editor Temporal propriamente dito como as outras	Muitos recursos e boa interface. Tem uma relação direta com o Editor Espacial que ajuda bastante	O uso de blocos com cores confunde um pouco no inicio e fica muito poluído visualmente.	Bastante confuso, é difícil entender a configuração realizada	O editor é bastante simples e intuitivo. Permite a visualização de todas as mídias
Editor Espacial	Amigável, tem uma área bastante grande e todos os objetos podem ser visualizados	A área é pequena , mas bastante amigável. Está associado ao Editor Temporal	Um pouco confuso, trabalha tanto diretamente com a mídia como com regiões	A área é pequena mas amigável. Trabalha sempre com regiões e não com mídias	A área é grande e de fácil utilização. Trabalha só com regiões e não com mídias diretamente
SMIL	Não utiliza o padrão SMIL. As aplicações são executadas através da própria ferramenta	Não utiliza o padrão SMIL. Gera aplicações Shockwave	Utiliza o padrão SMIL, mas cria arquivos com a extensão .grins	Utiliza o padrão SMIL e a aplicação pode ser executada em qualquer player que suporte SMIL	Utiliza o padrão SMIL e a aplicação pode ser executada em qualquer player que suporte SMIL
Cenas	Uma aplicação pode ser constituída de várias cenas, mas a navegação é linear	Não utiliza cenas	Não utiliza cenas	Não utiliza cenas	Uma aplicação pode ser constituída de várias cenas e a navegação é livre entre as cenas
Apresentação	É preciso sair do módulo de criação e entrar no módulo de visualização para executar a aplicação	A visualização é feita através do próprio Editor Espacial	A visualização é feita através da própria ferramenta, de forma simples	A visualização é feita através da própria ferramenta, de forma simples	Ainda não é possível visualizar diretamente na ferramenta a aplicação criada. É necessária uma ferramenta externa
Alteração do código da aplicação	Não é possível ver o código da aplicação criada	Utiliza uma linguagem chamada Lingo, para criar recursos mais avançados	É possível ver o código da aplicação criada, mas não alterá-lo	É possível ver o código da aplicação que está sendo desenvolvida e alterá-lo	É possível ver o código da aplicação que está sendo desenvolvida e alterá-lo
Utilização através da WEB	A utilização da ferramenta é local e não através da WEB	A utilização da ferramenta é local e não através da WEB	A utilização da ferramenta é local e não através da WEB	A utilização da ferramenta é local e não através da WEB	A ferramenta de autoria pode ser utilizada através da WEB

Tabela 1 - Tabela de comparação entre as características das ferramentas

5.5 Considerações finais

Neste capítulo foram apresentadas as principais características da ferramenta MAE, mostrando toda a sua arquitetura, os seus módulos, as estruturas de classe utilizadas, sua interface e como é utilizada pelo usuário.

No estágio atual de desenvolvimento da ferramenta, os principais módulos foram implementados. No Editor Espacial foi desenvolvida uma interface que permite uma boa visualização da aparência final da aplicação, componente muito importante em uma ferramenta de autoria de aplicações visuais.

Foi desenvolvido e implementado um modelo para o Editor Temporal, que procurou seguir o modelo utilizado pela linguagem SMIL. Como essa é a linguagem utilizada pela ferramenta, existe a necessidade de cobrir todas as possibilidades de configuração temporal que ela oferece, o que tornou mais complicada a sua implementação.

O modelo de utilização de cenas para a composição de uma aplicação já era previsto na primeira versão da ferramenta, mas somente foi implementado neste trabalho. Foi definida também uma estrutura de classes para o armazenamento das aplicações criadas.

A principal diferença da ferramenta MAE sobre as outras ferramentas é a integração com um SGBD, onde a ferramenta oferece algumas interfaces para a interação com a base de dados que fornecem recursos adicionais para o autor de uma aplicação, como consultas nebulosas e o uso de cenas para compor uma aplicação.

Foram encontradas muitas dificuldades na implementação e integração dos módulos componentes da ferramenta, devido à quantidade e complexidade de operações envolvidas em cada um deles. Ainda são necessários alguns ajustes na implementação de alguns desses módulos para que estejam estáveis, realizando todas as operações e tratamento de erros necessários. Como continuidade deste trabalho estão sendo realizados esses ajustes.

6. CONCLUSÕES

6.1 Considerações iniciais

Neste trabalho foram apresentados a ferramenta MAE e seu papel dentro de um ambiente de autoria e manipulação de aplicações multimídia. Os principais aspectos do ambiente foram descritos, destacando-se a forma de armazenamento e recuperação das aplicações em seu banco de dados multimídia.

Foram dicutidas as principais funcionalidades da MAE, mostrando suas vantagens em comparação com outras ferramentas de autoria. Entre essas funcionalidades, as que se destacam como mais relevantes da MAE são:

- Facilidade para modularização de uma aplicação através do conceito de cenas;
- Facilidade de reuso de mídias, cenas e aplicações;
- Utilização de um banco de dados multimídia com facilidades para consultas sobre mídias, cenas e aplicações;

Além desses aspectos, outra característica interessante, e também não encontrada nas outras ferramentas, é o fato da MAE ser implementada como um applet Java, o que possibilita a sua execução através da WEB em qualquer plataforma com suporte para essa linguagem.

Essa característica da ferramenta também torna interessante a sua utilização em um ambiente de Educação a Distância, para a criação de material didático. Os professores poderiam criar, acessar e editar seu material didático através da WEB. Para tal utilização são necessárias algumas alterações na interface da ferramenta para adequá-la a um ambiente de desenvolvimento de material didático, tornando-a mais amigável para os professores.

6.2 Contribuições

Como resultados deste trabalho pode-se destacar a definição e implementação de um modelo para o Editor Temporal, onde foram seguidas as especificações apresentadas pela linguagem SMIL. Também foi feita a integração com o SGBD Caché, utilizando a estrutura de classes definida no projeto AMMO para o armazenamento de informação semântica e definida uma estrutura para o armazenamento das aplicações. Foi definida toda a interface e o modo de interação da ferramenta com essa base de dados. Além disso, foi refinado o Editor Temporal, melhorando o seu funcionamento e aumentando seus recursos.

Neste trabalho foi efetivamente implementada a utilização de cenas para a composição de uma aplicação, realizando todas as alterações que isso implica no funcionamento global da ferramenta.

A possibilidade de autoria de aplicações multimídia através da Web e a funcionalidade de integração da ferramenta com uma base de dados que forneça facilidades para a edição e reuso de aplicações, ou parte delas, são características que diferem das ferramentas de autoria atualmente existentes.

6.3 Trabalhos futuros

Como continuidade deste trabalho, existem algumas funcionalidades que seriam importantes para aumentar os recursos da ferramenta e facilitar o desenvolvimento de aplicações. Essas funcionalidade são:

- Para que se mantenha uma homogeneidade na ferramenta, é interessante que se acople a ela um dispositivo que permita a visualização das aplicações construídas, através da própria ferramenta. Ou seja, a implementação de um módulo de visualização, que permite ao usuário visualizar, de forma rápida e fácil, a aplicação com a qual está trabalhando;
- Criação de um Editor de Navegação, que crie, gerencie e apresente ao usuário a navegação entre as várias cenas da aplicação; A importância desse tipo de funcionalidade e algumas abordagens podem ser encontradas em [Andersson, 1994] e [Fisher, 1994];
- Ampliação do Editor Temporal, possibilitando que o usuário defina diferentes intervalos de tempo para a barra temporal, que atualmente trabalha apenas com intervalos de 1 segundo;
- Implementar funcionalidade que permita à ferramenta trabalhar com a composição de aplicações, não só por cenas, mas também por outras aplicações;
- Criação de uma nova versão da ferramenta que trabalhe com SMIL 2.0, implementando as novidades apresentadas nessa nova versão da linguagem;
- Incorporação à ferramenta do módulo de consultas nebulosas;

7. REFERÊNCIAS BIBLIOGRÁFICAS

[Andersson, 1994] Andersson, J. J. E.; Using Conceptual Maps in Hypermedia. In Multimedia – Systems Architectures and Applications. Springer-Verlag, J. L. Encarnação, and J. D. Foley, Eds. 1994.

[Borges, 2001] Borges, S. R; Consultas Nebulosas Baseadas em Informações Semânticas em um Banco de Dados Multimídia, Programa de Pós-Graduação em Ciência da Computação, Departamento de Computação, UFSCar, São Paulo – Brasil, Agosto de 2001

[Buford, 1994] Buford, J. F. K.; Multimedia Systems. Addison-Wesley. 1994.

[Caché, 2002] Intersystems; Caché 4.1.6 - http://www.intersystems.com.br/; Julho de 2002

[Director, 2001] Macromedia; Director Shockwave Studio http://www.macromedia.com/software/director/; Dezembro de 2001

[Figueiredo, 2000] Figueiredo, J. M; Desenvolvimento de um Ambiente para Autoria e Manipulação de Aplicações Multimídia na World Wide Web, Dissertação de Mestrado, Programa de Pós-Graduação em Ciência da Computação, Departamento de Computação, UFSCar, São Paulo – Brasil, Agosto de 2000

[Fisher, 1994] Fisher, S.; Multimedia Authoring – Bulding and Developing Documents. Academic Press. 1994

[Fluition, 2002] Confluent Technologies; Fluition; http://www.confluenttechnologies.com/fluition.html; Janeiro de 2002

[Fornazari et al ,1999] Fornazari, F; Vieira, M. T. P; Biajiz, M; Busca nebulosa baseada em conteúdo em uma base de dados de aplicações MHEG-5, Anais do V Simpósio Brasileiro de Sistemas Multimídia e Hipermídia, Goiânia – Brasil, 1999

[Fornazari,1999] Fornazari, F; Sistema para tratamento de consultas nebulosas em aplicações multimídia para um servidor de objetos MHEG-5. Dissertação de Mestrado, Programa de Pós-Graduação em Ciência da Computação, Departamento de Computação, UFSCar, São Paulo – Brasil, Agosto de 1999

[Grins, 2002] Oratrix; Grins; http://www.oratrix.com/GRiNS; Janeiro de 2002

[Guojun, 1996] Guojun, L.;Communication and Computing for Ditributed Multimedia Systems. Artech House. 1996.

[Herzner, 1994] Herzner, W.; Finegrained Syncronisation in Dynamic Docuemnts. In Multimedia – Systems Architectures and Applications. Springer-Verlag, J. L. Encarnação, and J. D. Foley, Eds. 1994.

[Jasmine, 2000] Computer Associates; Jasmine *ii* - Documentação do Software, 2000

[Java, 2002] Sun Microsistem; Java Language; http://www.java.sun.com; Junho de 2002

[Little & Ghafoor, 1995] Little. D. C. T.; Ghafoor A.; *Interval-Based Conceptual Models for Time-Dependent Multimedia Data*. In A Guide Tour of Multimedia Systems and Applications. IEEE Computer Society Press, Borko Furht, and Milan Milenkovic, Eds. 1995

[Marshall, 2002] Marshall A.D.; Research Notes http://www.cs.cf.ac.uk/Dave/Multimedia/; Janeiro de 2002

[Mayes, 1994], Mayes, J. T.; The 'M-Word': Multimedia Interfaces and Their Role in Interactive Learning Systems. In Multimedia Interface Design in Education. Springer-Verlag, Edwards D. N. A., and Holland S., Eds. 1994

[Quick, 2002] Apple; QuickTime; http://www.apple.com/quicktime/; Janeiro de 2002

[Real, 2002] Real Networks; www.real.com; Dezembro de 2001

[Santos et al, 2002a] Santos, F.G.; Vieira, M.T.P.; Figueiredo, J.M. MAE: uma ferramenta de autoria multimídia através da WEB. VIII Simpósio Brasileiro de Sistemas Multimídia e Hipermídia – SBMIDIA 2002. Fortaleza – Ceará – Brasil, Outubro de 2002.

[Santos et al, 2002b] Santos, F.G.; Vieira, M.T.P.; Figueiredo, J.M. A ferramenta de autoria multimídia MAE. Workshop de Ferramentas e Aplicações. Fortaleza – Ceará – Brasil, Outubro de 2002.

[Santos et al., 2000] Santos, M. T. P.; Vieira, M. T. P.; Borges, S. R.; Figueiredo, J. M.; Fornazari, F. P.; Biajiz, M. (2000). Semantic Information Search Facilities for MHEG-5 and SMIL Application. FQAS 2000: Fourth International Conference on Flexible Query Answering System, Warsaw – Poland, October 2000. Procedings publicados pela Springer-Verlag, na série "Advances in Soft Computing"

[Schank, 1995] Schank, R. C.; *Active Learning through Multimedia*. In A Guide Tour of Multimedia Systems and Applications. IEEE Computer Society Press, Borko Furht, and Milan Milenkovic, Eds. 1995

[SMIL, 1998] W3C Recommendation; Synchronized Multimedia Integration Language (SMIL) 1.0 Specification; http://www.w3.org/TR/REC-smil/; Junho de 1998

[SMIL, 2001] W3C Recommendation; Synchronized Multimedia Integration Language (SMIL) 2.0 Specification; http://www.w3.org/TR/smil20/; Agosto de 2001

[Steinmetz & Nahrsted, 1995]. Steinmetz, R.; Nahrsted, K.; Multimedia: Computing, Communications and Applications. Prentice-Hall. 1995

[Sun, 2000] Sun Microsystems; JAXP – Java API for XML Parsing – version: 1.0. http://java.sun.com/xml; Fevereiro de 2000

[Tatizana, 1999] Tatizana, C., Visual Class 4.2 Multimídia – Software para Criação, Editora Érica, 1999

[Vieira & Santos, 1997] Vieira, M.T.P.; Santos, M.T.P. (1997). Content-based Search on an MHEG-5 Standard-based Multimedia Database. In Proceedings of the QPMIDS DEXA 97, Toulouse – France, pp. 154-159

[Vieira et al, 2002] Vieira, M.T.P.; Biajiz, M.; Borges, S.R.; Teixeira, E.C.; Santos, F.G.; Figueiredo, J.M. Content-Based Fuzzy Search in a Multimedia Web Database. In Intelligent Exploration of the Web. Springer-Verlag, "Studies in Fuzziness and Soft Computing" series, P.S.Szczepaniak, F.Segovia, J. Kacprzyk, and L.A.Zadeh, Eds, 2002 (to appear).

[Vieira et al., 1999a] Vieira, M.T.P.; Biajiz, M; Santos, M.T.P; Ribeiro, L.C; Fornazari, F.P. Metadata for Content-Based Search on an MHEG-5 Multimedia Objects Server. In Proceedings of the Third IEEE-Meta-Data Conference, NIH Campus, Bethesda Maryland – USA, 1999

[Vieira et al., 1999b] Vieira, M. T. P.; Seno, W. P; Santos, M. T. P; Figueiredo, J. M. Ambiente EAD-MAW para Educação a Distância. Workshop Internacional sobre Educação Virtual – WISE'99, Fortaleza – Ceará, Brasil, 1999

[XML, 2001] W3C Recommendation; Extensible Markup Language (XML); http://www.w3.org/XML/; Novembro de 2001;

Apêndice A Documentação dos módulos da ferramenta MAE

Neste apêndice é apresentada a documentação da ferramenta MAE, onde são descritos os algoritmos dos vários módulos que a compõem. Os diagramas de classes utilizados na concepção da ferramenta, que foram apresentados nos capítulos desta dissertação, são novamente aqui mostrados com mais detalhes sobre os métodos e atributos das classes, sendo explicadas suas funcionalidades e como são utilizados na ferramenta. Também são apresentadas novamente as interfaces da ferramenta, de forma a reunir aqui toda a informação necessária para o detalhamento sobre a implementação da ferramenta MAE.

1. Definição dos Atores

Existe apenas um ator (Usuário – Figura 56) que interage com o sistema. Ele é quem utiliza a ferramenta MAE para a criação e edição de aplicações multimídia.

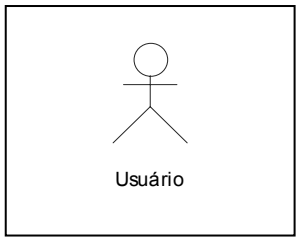


Figura 56 - Ator do sistema

2. Diagrama de Casos de Uso

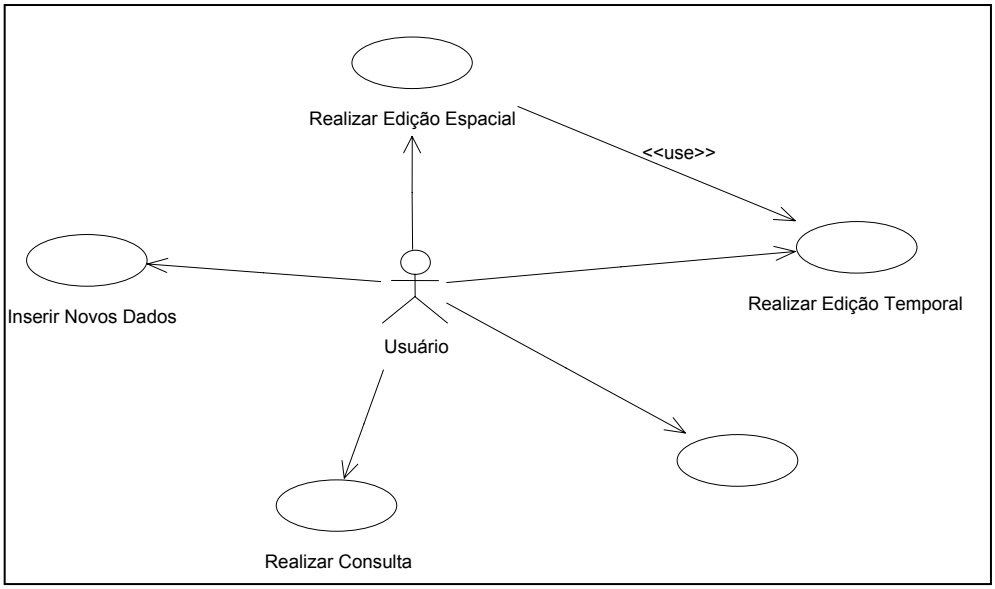


Figura 57 - Diagrama de Casos de Uso

3. Descrição dos Casos de Uso

- **Caso de Uso: Realizar Edição Espacial**

  Esse Caso de Uso é responsável pela edição espacial das mídias contidas em uma cena dentro da aplicação que está sendo desenvolvida através da ferramenta MAE. A edição espacial é responsável pela definição do tamanho de cada mídia na tela da cena e o posicionamento espacial dessas mídias. É através do Editor Espacial que são inseridas as mídias na aplicação.

- **Caso de Uso: Realizar Edição Temporal**

  Esse Caso de Uso é responsável pela edição temporal das mídias inseridas em uma cena da aplicação que está sendo desenvolvida. É através da edição temporal que é definido o tempo de execução de cada mídia da cena e é feita a sincronização entre essas mídias.

- **Caso de Uso: Inserir Informação Semântica**

  Esse Caso de Uso é responsável pela inserção de informação semântica sobre as mídias que foram inseridas na aplicação em desenvolvimento. Essas informações ficam armazenadas no banco de dados e são utilizadas na realização de consultas.

- **Caso de Uso: Inserir Novos Dados**

  Esse Caso de Uso é responsável pela criação dos dados que são utilizados na definição da informação semântica das mídias. É utilizado quando os dados já existentes na base de dados não são suficientes para descrever a informação semântica de alguma das mídias inseridas na aplicação.

- **Caso de Uso: Realizar Consulta**

  Esse Caso de Uso é responsável pela consulta à base de dados para a recuperação de mídias e cenas que são utilizadas na construção de novas aplicações. Essa consulta é feita utilizando as informações semânticas inseridas sobre as mídias.

4. Diagrama de Classes

A Figura 58 apresenta as principais classes utilizadas no desenvolvimento da ferramenta MAE. Muitas outras classes também foram utilizadas, sendo que tais classes são comentadas nas próximas subseções, que apresentam uma descrição mais detalhada de cada um dos módulos da ferramenta. Através dessas classes é possível se ter uma visão geral de como a ferramenta foi implementada. A ferramenta possui aproximadamente 9000 linhas de código, sendo aqui transcritos os códigos de alguns métodos para fins de ilustração.

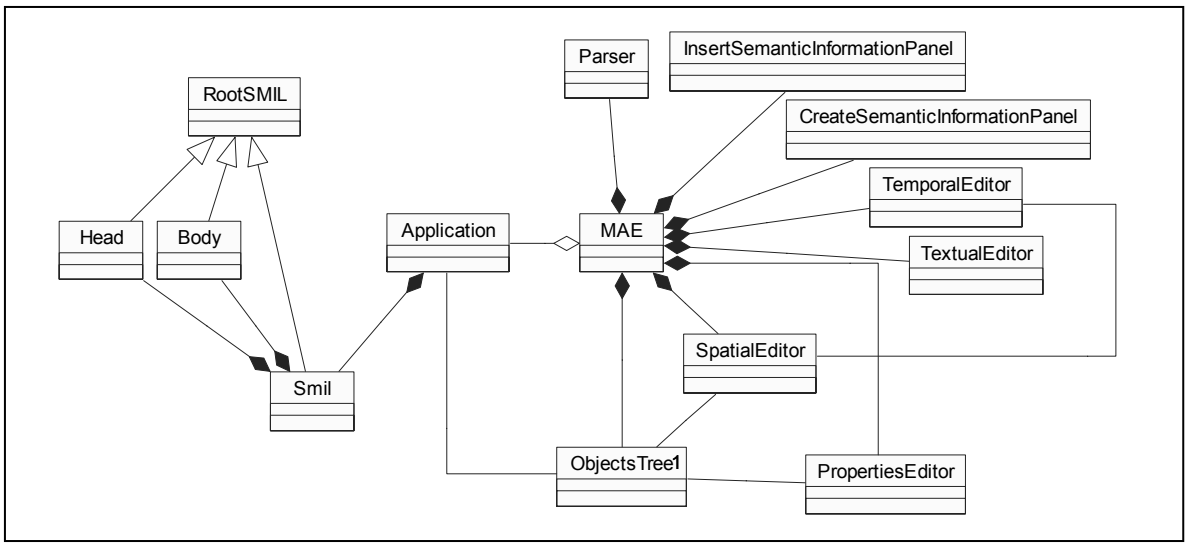


Figura 58 - Diagrama de classes da Ferramenta MAE

Para manipular as informações referentes à aplicação que está sendo desenvolvida pela ferramenta, foi criada uma estrutura de classes (Figura 59) onde existe uma classe para cada tag definida na DTD do SMIL 1.0. Os atributos dessas classes representam os atributos das tags SMIL e através do método *setAttribute( )* é que se altera os valores desses atributos. Tal método recebe como parâmetros o nome do atributo e o valor que ele deve receber. Como já dito anteriormente, foi introduzida uma classe adicional, a classe Application, não prevista no padrão. Seu objetivo é permitir representar uma aplicação multimídia como constituída de vários documentos SMIL, onde cada documento é uma cena da aplicação. É através dessas classes que são gerados os códigos das cenas da aplicação que está sendo desenvolvida, com base nos valores dos atributos de cada objeto criado para a aplicação. Todos os editores estão de alguma forma associados a essa estrutura, pois cada alteração feita em um editor deve refletir na criação de um objeto ou na alteração do valor de um ou mais atributos. A classe MediaContent não faz parte da definição SMIL, sendo usada para armazenar informações

sobre as mídias que são utilizadas na aplicação. Essa classe contém o nome, o tipo, a extensão e um breve resumo sobre o conteúdo da mídia.

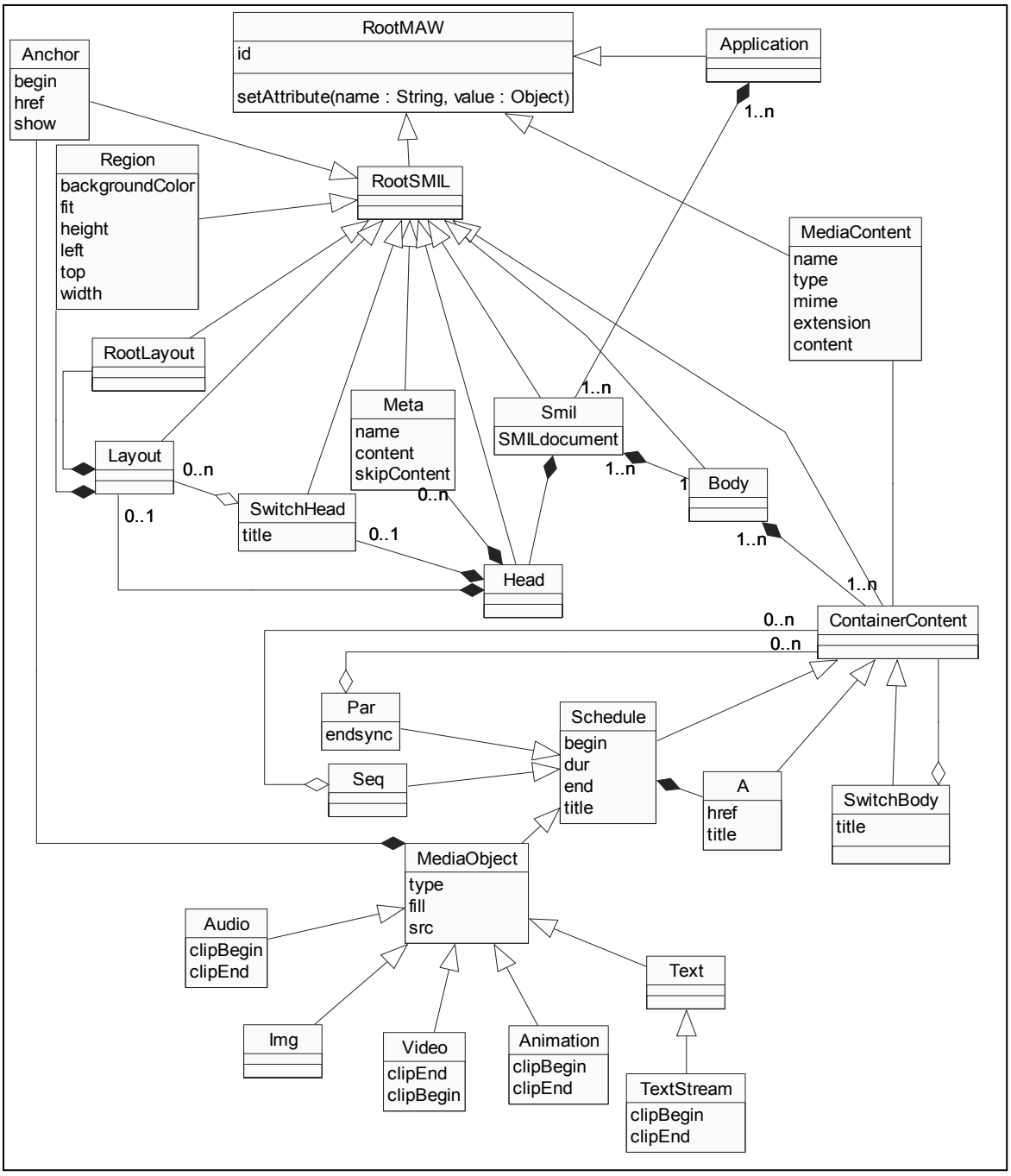


Figura 59 - Estrutura de classes SMIL

A estrutura de classes da Figura 60 mostra as classes criadas para armazenar a informação semântica das mídias e que dão suporte à busca por conteúdo. Essa estrutura já foi apresentada anteriormente, mas é aqui colocada novamente pois essas classes estão associadas a algumas das classes da Ferramenta MAE, que serão apresentadas nas próximas seções.

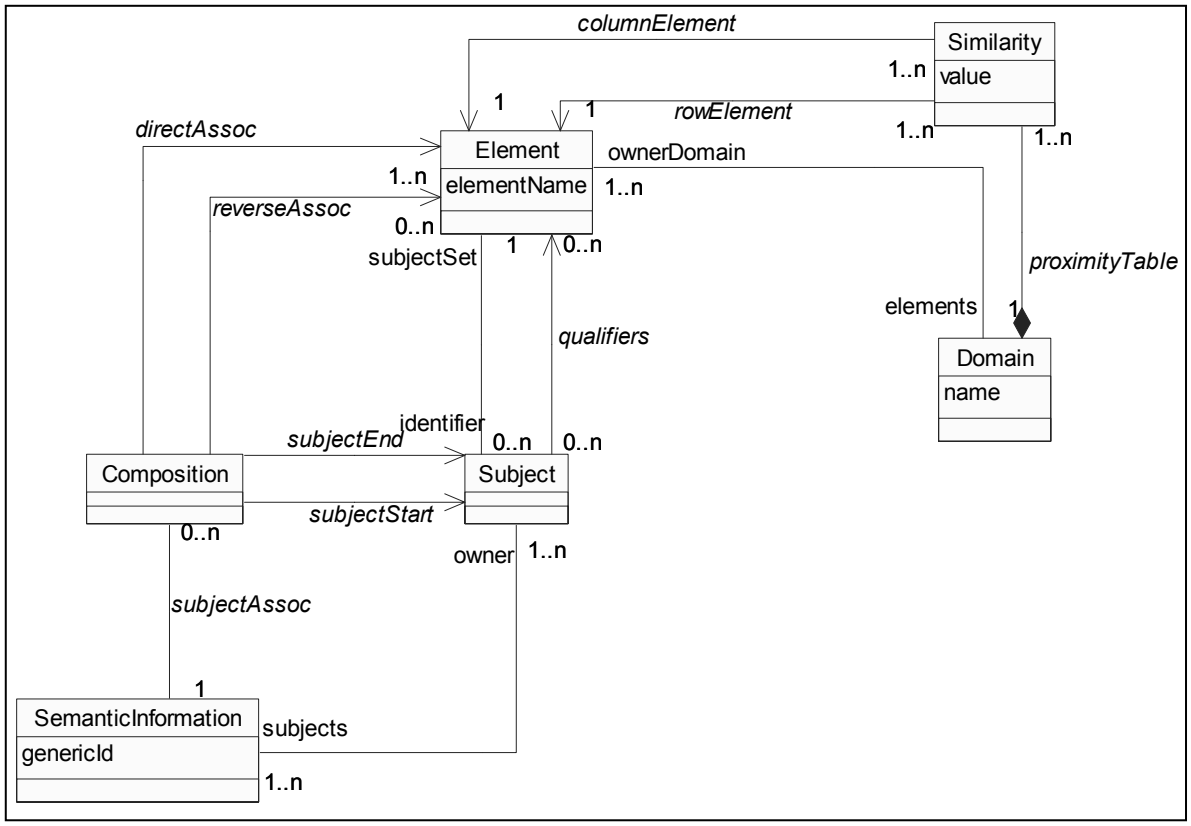


Figura 60 - Estrutura de classes do BDMm para o armazenamento da informação semântica

4.1. Editor Temporal

Figura 61 mostra a estrutura de classes criada para o desenvolvimento do Editor Temporal, onde é possível observar que existe uma associação com algumas classes do Editor Espacial. Isso acontece para que no momento em que o usuário insere uma mídia na sua aplicação através do Editor Espacial, é automaticamente inserido no Editor Temporal um objeto da classe TemporalObject que representa essa mídia (Figura 62). O Editor Espacial informa o nome da mídia e a sua duração, que são informações utilizadas na criação do objeto. É através desse objeto que é feita a edição temporal dessa mídia dentro da cena que está sendo desenvolvida.

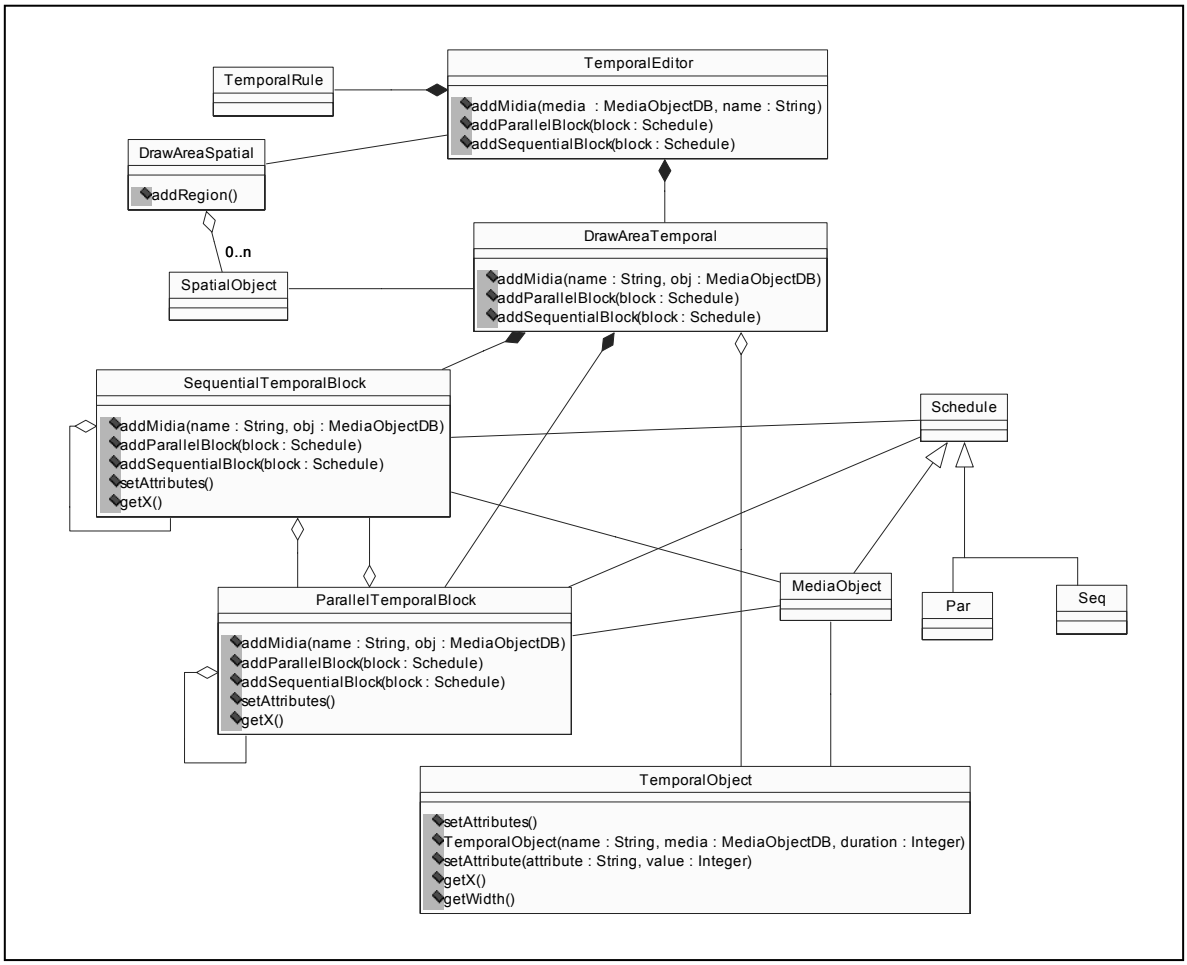


Figura 61 - Diagrama de classes do Editor Temporal

A Figura 63 apresenta o algoritmo que descreve o comportamento do Editor Temporal; nesse algoritmo a largura, na tela, do objeto que representa a mídia indica a sua duração e o seu posicionamento na tela indica o tempo em que será iniciada a sua exibição. O código do método setAttributes ( ), descrito no algoritmo da Figura 63, é mostrado na figura 64.

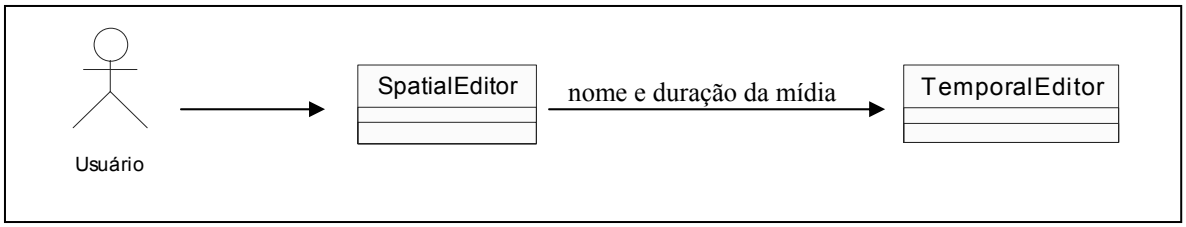


Figura 62 - Exemplo da seqüência de ações que ocorrem na MAE

```
Ao inserir uma mídia M no Editor Espacial ativar o método addMidia( M ) da classe DrawAreaSpatial //Desenha na tela um objeto T (da classe TemporalObject) que representa
                    //tempo de duração da mídia
Caso evento e  //Se solicitado pelo usuário através do Editor Temporal
     e1: alterar a posição do objeto M na tela; //e1=alterar o tempo de inicio da apresentação de M
          ativar o método setAttributes( )  de T // atualiza os atributos de duração e inicio de M
     e2: alterar a largura do objeto M na tela; // e2 = alterar a duração da mídia
          ativar o método setAttributes( ) de T // atualiza os atributos de duração e inicio de M
Se alterado um atributo pelo usuário através do Editor de Propriedades
          Ativar o método setAttribute( ) de T// alterar a posição ou a largura do objeto T

addMidia( )
inicio
          new TemporalObject T( M) //Associa ao objeto T o objeto M da classe MediaObject
          Ativar o método getX( ) da classe TemporalObject // obtém a posição de T na tela, que
          representa o tempo de inicio da exibição
          Ativar o método getWidth( ) //verifica a largura de T, determinando a duração da mídia M
          Ativar o método setAttributes( ) // atualiza os atributos de inicio e duração do objeto M
fim

setAttributes( )
inicio
          ativar o método  getX( ) de T //obtém a posição de T, indicando o tempo de inicio de M
          ativar o método setAttribute( ) de M //método herdado da classe RootMAW. Atualiza o
                                                    //atributo de inicio do objeto M
          ativar o método getWidth( ) de T //obtém a largura de T, indicando a duração de M
          ativar o método setAttribute( ) de M //atualiza o atributo de duração do objeto M
          sincronizar os módulos da ferramenta //altera os valores no Editor de Propriedades e no
          código
fim
```

Figura 63 - Algoritmo do funcionamento da edição temporal

```
private void setAttributes()
{
  media.setAttribute("begin",String.valueOf(getX()/18)+"s");
  media.setAttribute("dur",String.valueOf(getWidth()/18)+"s");
  try
  {
    callSynchronizeModules(media);
  } catch (java.net.MalformedURLException exc)
    {
      System.out.println("Erro: "+exc);
    }
}
```

Figura 64 - Código do método setAttributes( )

O algoritmo da Figura 63 trabalha dentro de um bloco paralelo, ou seja, o Editor Temporal possui apenas um bloco paralelo e todas as mídias estão dentro desse bloco. Por esse motivo, os blocos (paralelos e seqüenciais) não são contemplados no algoritmo. Essa é a forma na qual a ferramenta trabalha atualmente, sendo que a combinação de vários blocos está

sendo desenvolvida. A Figura 65 mostra a interface do Editor Temporal, em destaque na área 1.

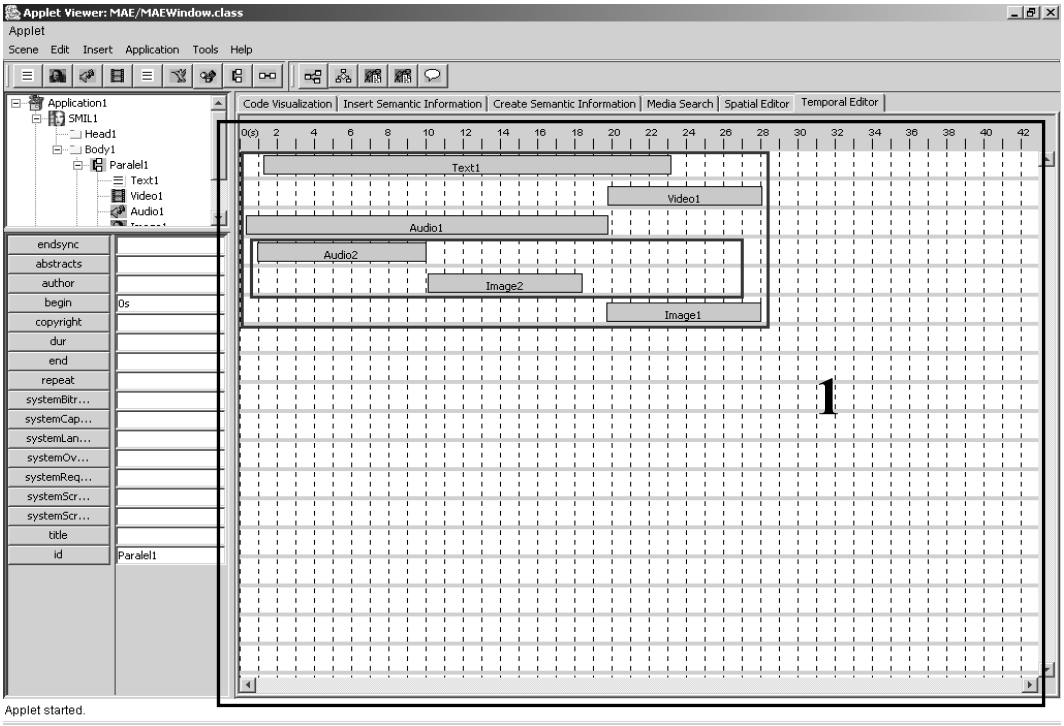


Figura 65 - Editor Temporal da ferramenta MAE

A Figura 66 mostra através de um exemplo, como deve ficar a composição de objetos do Editor Temporal e em relação à ferramenta MAE. Nesse exemplo, é apresentada uma possível instanciação das classes que compõem o Editor Temporal, sendo criados vários objetos e mostrando como é feita a composição visual desses objetos.

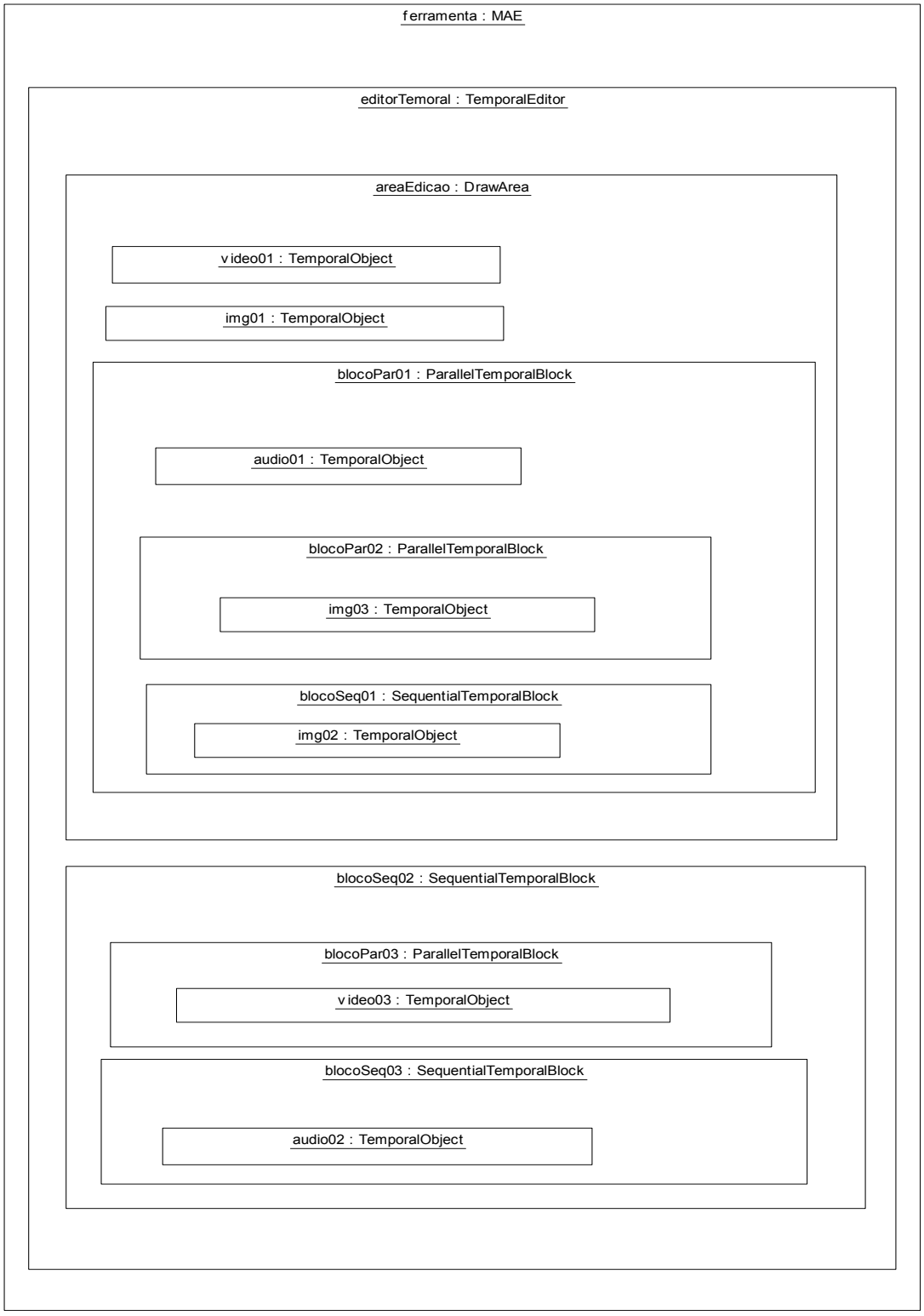


Figura 66 - Composição de objetos do Editor Temporal

4.2. Editor Espacial

A área 1 da Figura 67 apresenta a interface do Editor Espacial. É através desse editor que são inseridas as mídias nas cenas da aplicação. Existe uma área de edição para cada cena, onde são visíveis todas as mídias inseridas na respectiva cena. Através desse editor, o usuário realiza a configuração do layout da aplicação, definindo o tamanho e o posicionamento das mídias durante a execução da aplicação.

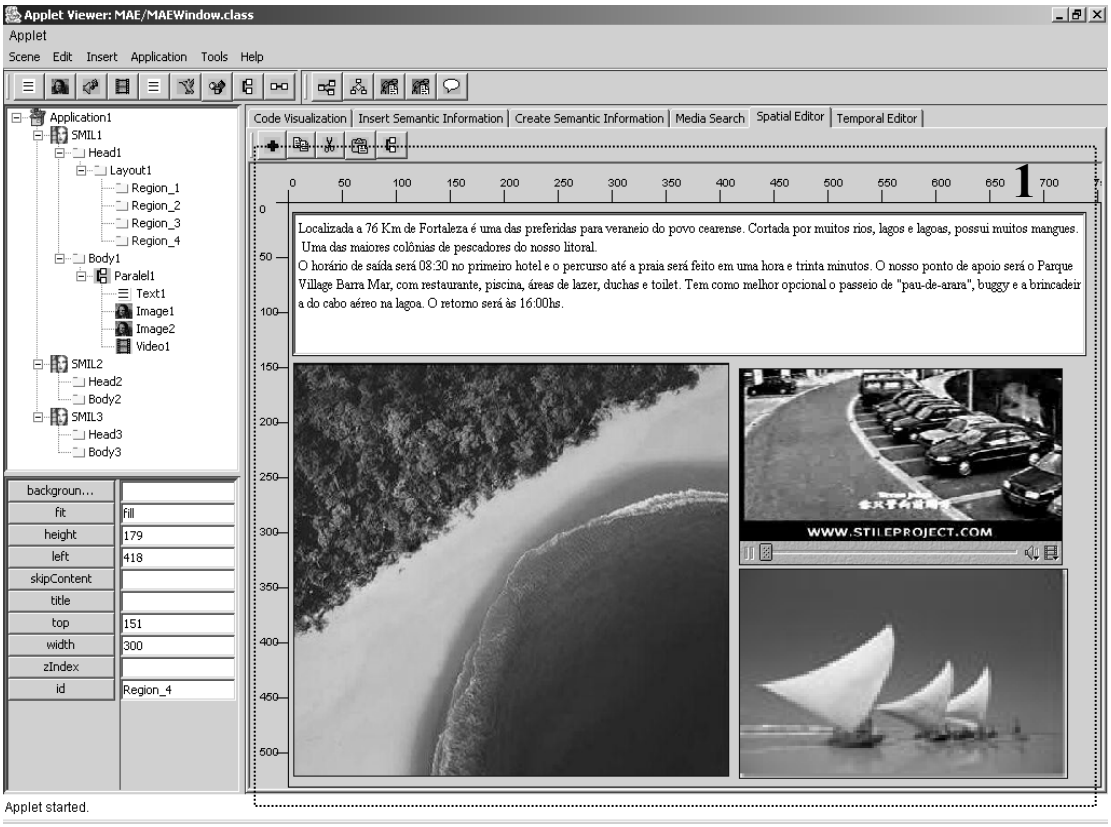


Figura 67 - Interface do Editor Espacial da ferramenta MAE

A Figura 68 mostra a estrutura de classes criada para o desenvolvimento do Editor Espacial, que é composto por uma régua (Ruler), a qual contém a numeração que indica a posição da mídia, por uma barra de ferramentas (ToolBar), que possui alguns botões (Button) usados pelo usuário para manipular o editor, além de uma ou mais áreas de edição (DrawAreaSpatial). Existe uma área de edição para cada cena da aplicação que está sendo desenvolvida e cada área possui uma ou mais mídias (SpatialObject), que são aquelas inseridas na cena associada a essa área de edição. Além disso, o Editor Espacial está associado às classes da DTD SMIL, que recebem os dados referentes à configuração espacial realizada pelo usuário.

Na classe SpatialEditor, o método addDrawArea( ) adiciona uma nova área de edição para cada nova cena inserida na aplicação e o método changeDrawArea( ) altera a área de edição corrente. Já o método addRegion( ) chama o método addRegion(reg: Region) da área de edição corrente, que cria uma nova região. Essa nova região é um objeto da classe SpatialObject que associa uma mídia a ele e realiza a inserção dessa mídia no BDMM (insertDB()) e também configura os atributos (setAttributes()) da tag SMIL associada a esse objeto.

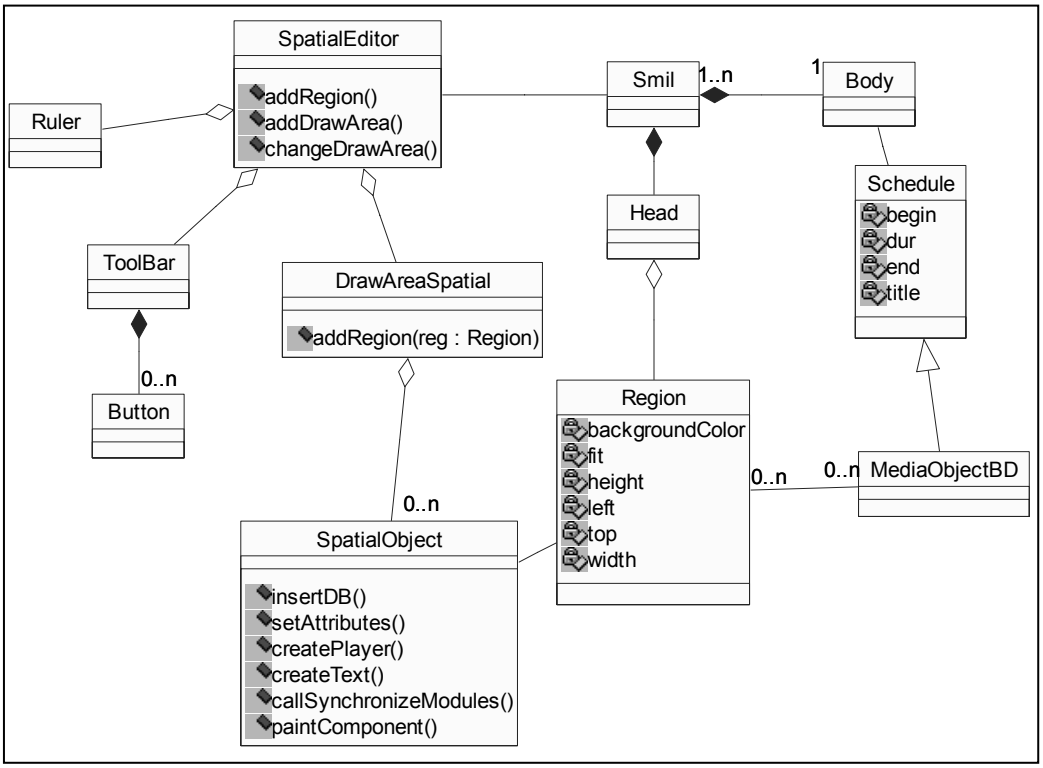


Figura 68 - Diagrama de classes do Editor Espacial

O algoritmo da Figura 69 descreve o funcionamento do Editor Espacial, sendo detalhados os métodos addRegion( ) e setAttributes( ). Já na Figura 70 é apresentado o código do método setAttributes( ) da classe SpatialObject.

```
Ao acionar o botão B é ativado o método addRegion ( ) da classe SpatialEditor
//botão com o símbolo + ativado pelo usuário
Caso evento e  //Se solicitado pelo usuário através do Editor Espacial
      e1: alterar a posição do objeto M na tela;
            ativar o método setAtributes( )  de M// atualiza os atributos de posicionamento de M
      e2: alterar a largura do objeto M na tela;
            ativar o método setAtributes( ) de M// atualiza o atributo de largura de M
      e3: alterar a altura do objeto M na tela;
            ativar o método setAtributes( ) de M// atualiza o atributo de altura de M
Se alterado um atributo pelo usuário através do Editor de Propriedades
           Ativar o método setAtribute( ) de M// alterar a posição, altura ou a largura do objeto M


addRegion()
inicio

           captura evento de seleção de mídia do usuário
           criar um objeto M da classe SpatialObject na área do objeto D da classe DrawAreaSpatial
           //o objeto M apresenta a mídia selecionada
fim

setAttributes( )
inicio
           ativar o método  getX( ) de M //obtém a posição de M
           ativar o método setAttribute ( ) de M //método herdado da classe RootMAW. Atualiza o
                                                 //atributo “left” do objeto M
           ativar o método  getY( ) de M //obtém a posição de M
           ativar o método setAttribute ( ) de M //método herdado da classe RootMAW. Atualiza o
                                                 //atributo “top” do objeto M
           ativar o método getWidth( ) de M //obtém a largura de M
           ativar o método setAttribute ( ) de M //atualiza o atributo “width” do objeto M
           ativar o método getHeight ( ) de M //obtém a altura de M
           ativar o método setAttribute ( ) de M //atualiza o atributo “height” do objeto M
           sincronizar os módulos da ferramenta //altera os valores no Editor de Propriedades e
                                                 //no código
fim
```

Figura 69 - Algoritmo do funcionamento da edição espacial

```
private void setAtributes()
  {
   media.setAttribute("left",String.valueOf(getX()));
   media.setAttribute("top",String.valueOf(getY()));
   media.setAttribute("width",String.valueOf(getWidth()));
   media.setAttribute("height",String.valueOf(getHeight()));
   try
   {
    callSynchronizeModules(media);
   } catch (java.net.MalformedURLException exc)
     {
      System.out.println("Erro: "+exc);
     }
  }
```

Figura 70 - Código do método setAttributes( ) da classe SpatialObject

4.3. Árvore de Objetos e Editor de Propriedades

A área 1 da Figura 71 mostra a interface da Árvore de Objetos utilizada na ferramenta MAE. Ela apresenta todas as cenas, regiões e mídias inseridas na aplicação. Como pode ser observado na estrutura de classes (Figura 72) da Árvore de Objetos, ela está associada ao Editor de Propriedades, ao Editor Espacial e ao Editor Temporal. Isso ocorre pelo fato de que o usuário ao clicar em um nó da árvore, o Editor de Propriedades (área 2 da Figura 71) deve apresentar as propriedades do objeto, associado a esse nó, e seus valores, enquanto que o Editor Temporal deve apresentar a configuração temporal da cena a qual pertence esse objeto, o Editor Espacial deve apresentar a configuração espacial da cena e o Editor Textual deve mostrar o código gerado para tal cena. Um clique em um nó da árvore tem reflexo em vários outros módulos da ferramenta. Atualmente o Editor Espacial simplesmente mostra o código gerado pelo Parser e não tem uma classe especifica para ele, é apenas uma área de texto.

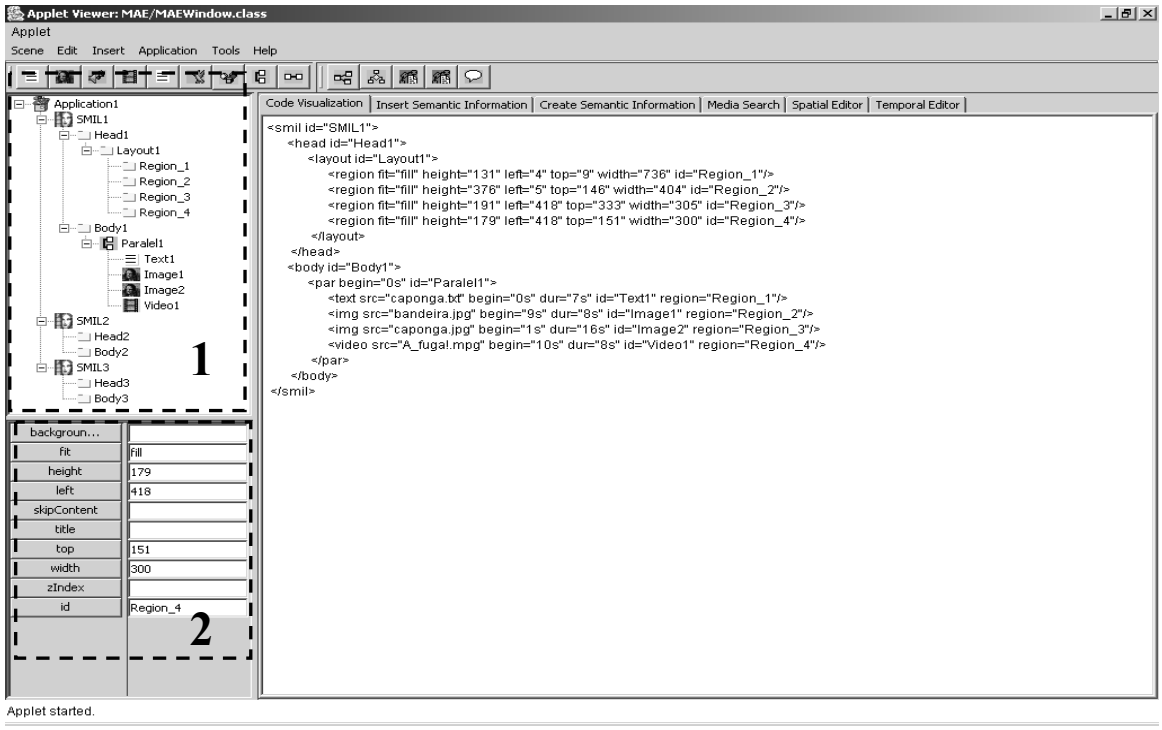


Figura 71 - Interface da ferramenta MAE com o Editor Textual

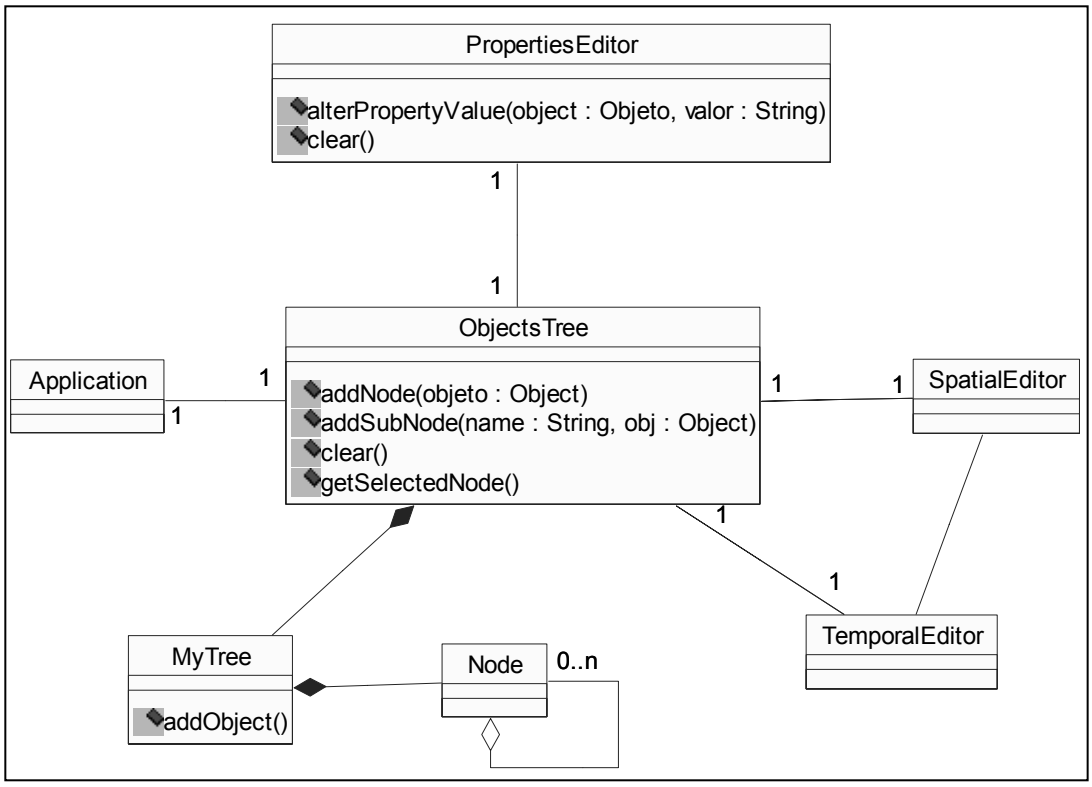


Figura 72 - Diagrama de classes da Árvore de Objetos e Editor de Propriedades

Algumas funcionalidades devem ser inseridas nesse editor, como por exemplo, a verificação da correção da sintaxe caso o usuário altere o código diretamente no editor. Essas alterações levarão à necessidade da criação de uma classe para o Editor Textual, assim como existe para os outros editores.

A Árvore de Objetos reflete o que está sendo criado e editado nos editores espacial e temporal. Para que isso ocorra, a Árvore de Objetos utiliza a estrutura SMIL (Figura 59) como base para a sua criação. A manipulação dos editores temporal e espacial pelo usuário gera a criação de objetos dessa estrutura, fazendo com que isso reflita na árvore. O algoritmo da Figura 73 descreve o comportamento da Árvore de Objetos, que é configurada na ocorrência de alguns eventos gerados pelo usuário ao manipular a aplicação em desenvolvimento.

```
Caso evento e //eventos que ocorrem no sistema através da ação do usuário
    e1: inserir uma nova cena na aplicação
        ativar o método addNode( ) //adiciona um novo nó na árvore
    e2: inserir uma nova mídia em uma cena
        ativar o método addSubNode( ) //adiciona um novo nó dentro da cena especifica
    e3: clicar em um nó da árvore
        ativar o método getSelectedNode( )
        selecionar o Editor Espacial da cena a qual o nó está associado
        selecionar o Editor Temporal da cena a qual o nó está associado
        apresentar no editor de Propriedades os atributos e valores correspondentes ao nó
        selecionado
```

Figura 73 - Algoritmo de funcionamento da Árvore de Objetos

```
public DefaultMutableTreeNode addObject(Object child)
{
   DefaultMutableTreeNode parentNode = null;
   TreePath parentPath = this.getSelectionPath();
   if (parentPath == null) parentNode = myroot;
   else
   {
     parentNode = (DefaultMutableTreeNode)(parentPath.getLastPathComponent());
   }
   return this.addObject(parentNode, child, true);
}
```

Figura 74 - Código do método addObject( ) da classe MyTree

4.4. Definição de Informação Semântica

A Figura 75 (área 1) mostra a interface utilizada para a inserção da informação semântica referente às mídias utilizadas na construção da aplicação. Para a criação dessa interface e o armazenamento temporário das informações, foi desenvolvida a estrutura da Figura 76. Essa estrutura armazena em memória a informação semântica de uma mídia, inserida pelo usuário, até que essas informações sejam armazenadas na base de dados. A classe SubjectTemp armazena os temas e seus qualificadores, a classe AssociationTemp contém as associações entre esses temas. A classe VisualizationPanel fornece uma área de visualização para a mídia sobre a qual está sendo inserida a informação semântica. Já a classe MediaInformation contém informações sobre as mídias, como o arquivo referente a ela, o tipo e o seu oid na base de dados. Essas últimas informações são necessárias para a apresentação da mídia nessa interface e para o armazenamento dos dados semânticos na base de dados. Elas são geradas

através do Editor Espacial. As classes Domain e SemanticInformation foram apresentadas na estrutura da Figura 60.

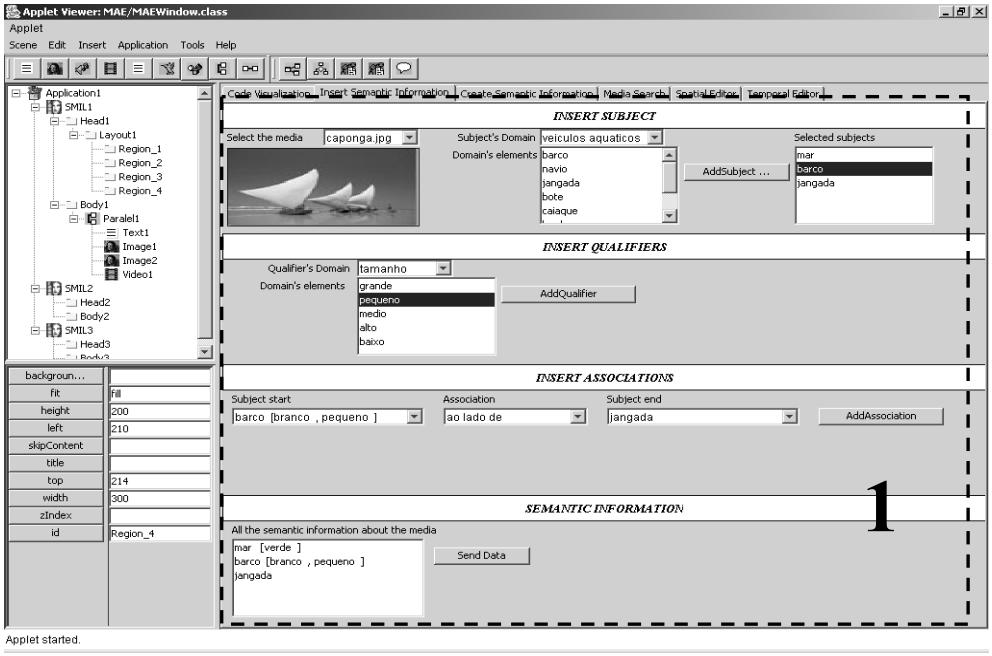


Figura 75- Interface para inserção de informação semântica

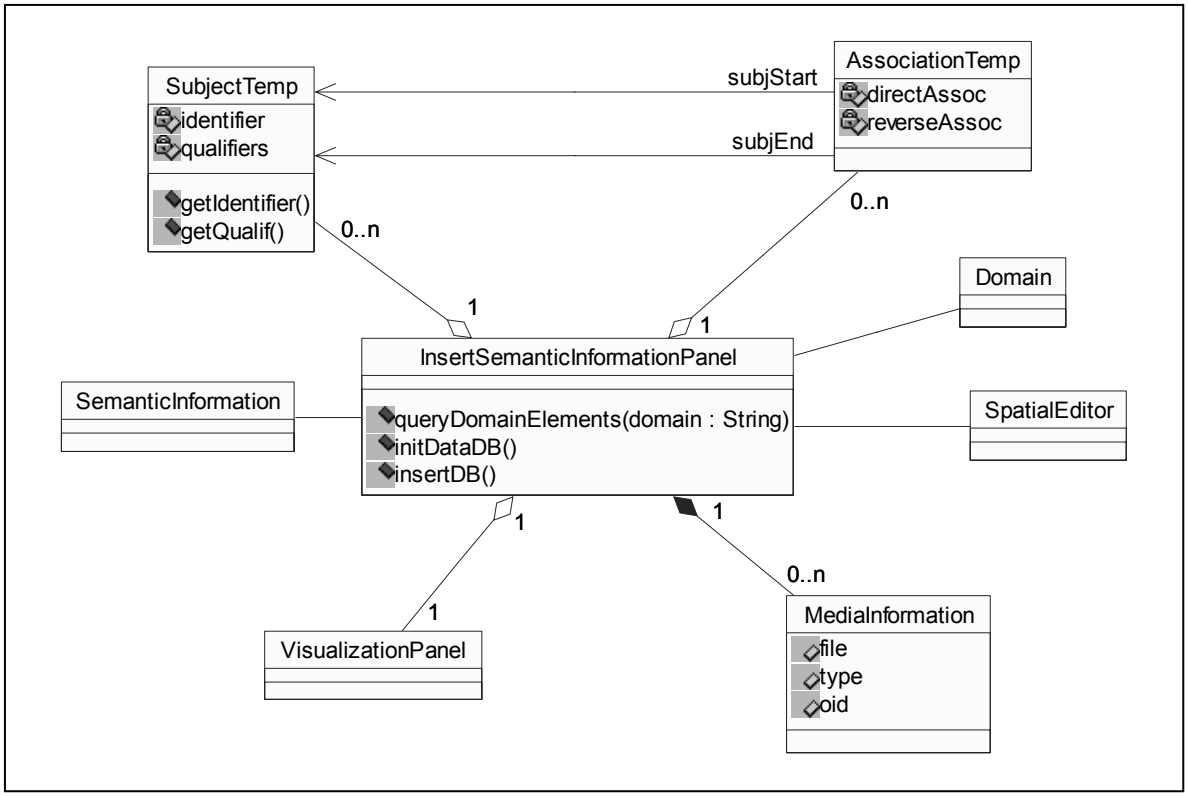


Figura 76 - Diagrama de classes para inserção de informação semântica

O algoritmo da Figura 77 descreve o comportamento da inserção de informação semântica através da ferramenta MAE. Esse algoritmo faz um uso intenso da estrutura de classes para armazenamento da informação semântica apresentada na Figura 60.

```
Ativar o método initDataDB( ) //recupera do BDMm os domínios existentes
Capturar uma mídia M //selecionada pelo usuário
Apresentar a mídia M em um objeto da classe VisualizationPanel
Capturar um domínio D de Subjects //selecionado pelo usuário
Ativar queryDomainElements (D) //recupera todos os elementos do domínio D
Para cada elemento do domínio D selecionado pelo usuário
Inicio
        Criar um objeto S da classe SubjectTemp
        Inserir o elemento selecionado no atributo tema do objeto S
        Atualizar a visualização da informação semântica
Fim
Capturar elemento E entre os selecionados //elemento escolhido pelo usuário
Capturar um domínio Q de Qualifiers //selecionado pelo usuário
Ativar queryDomainElements(Q) //recupera todos os elementos do domínio Q
Para cada elemento X selecionado pelo usuário
Inicio
        Inserir o elemento X no atributo qualifiers do objeto S que contém    o elemento
        E como tema
        Atualizar a visualização da informação semântica
Fim
Capturar um elemento I como subject start //selecionado pelo usuário
Capturar um elemento A como association //selecionado pelo usuário
Capturar um elemento F como subject end //selecionado pelo usuário
Ao acionar o botão "AddAssociation" criar um objeto AT da classe AssociationTemp e
inserir I, A e F nos atributos subjStart, directAssoc e subjEnd
Ao acionar o botão "SendData" ativar o método insertDB( ) //Inserir os dados no BDMm
```

Figura 77 - Descrição do comportamento da ferramenta na inserção de informação semântica

O código da Figura 78 é um trecho do método *queryDomainElements( )*, onde é passado o nome de um domínio e são recuperados todos os elementos inseridos dentro desse domínio, seja um domínio de *subjects* ou de *qualifiers.*

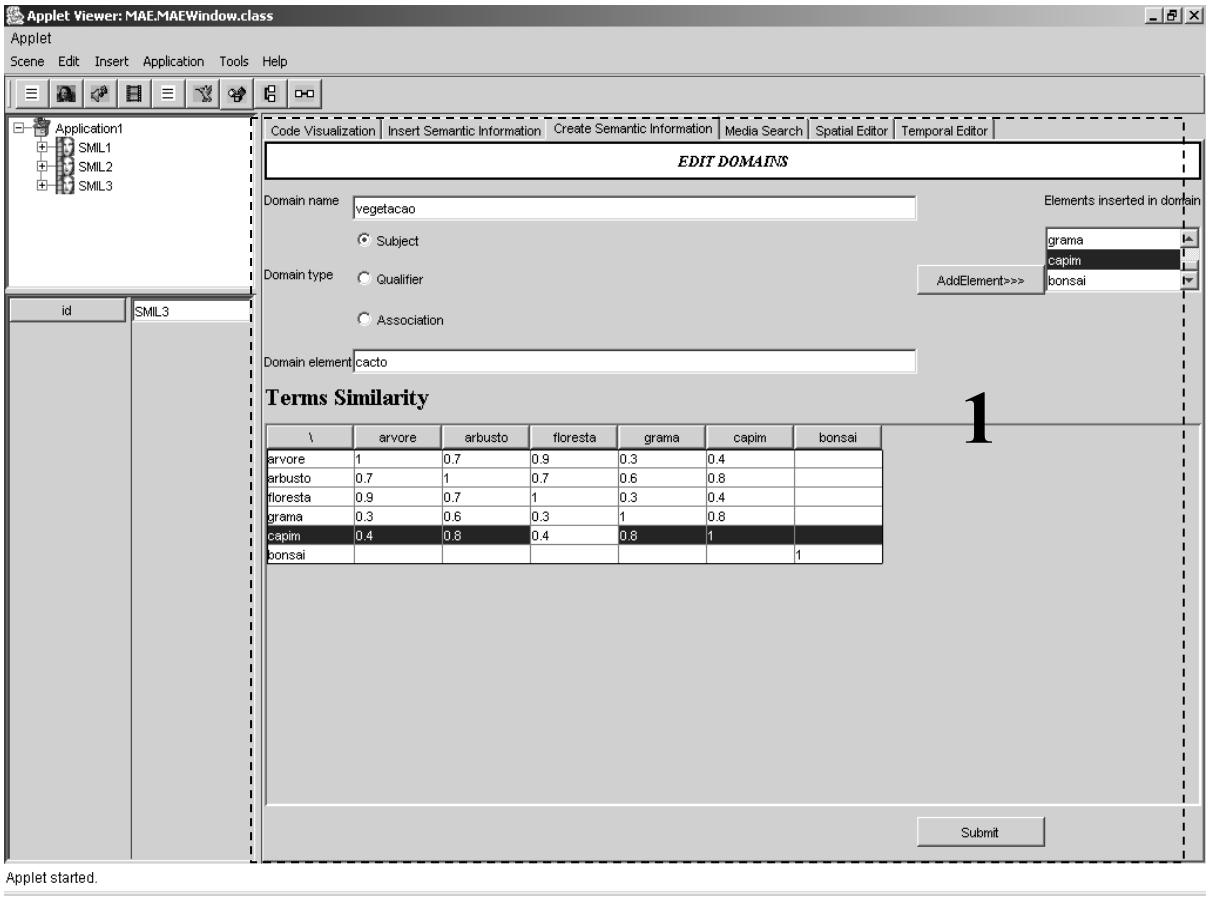
```
private Vector queryDomainElements(String domain)
{
    ResultSet rs;
    int id;
    Vector retorno = new Vector();
    try{
      String consulta = "SELECT ID FROM bdmm.domain WHERE name = '"+ domain +"\'";
      rs = new ResultSet(factory,consulta,1);
      rs.execute();
      rs.next();
      id = rs.getInt(1);
      rs.close();
      Domain domain = new Domain(factory,id);
      RelationshipObject domainElements = domain.getelements();
      String key = null;
      key = domainElements._next
      StringHolder sh = new StringHolder(key
      int num = domainElements._count();
      if(domain.gettype().equals("qualificador")){
         vElementsQual = new Vector();
         for(int i = 0; i < num; i++) {
           Element elem = new Element(factory, domainElements._getObjectNext(sh));
           vElementsQual.add(elem._getOid());
           ListOfDataTypes elementName = elem.getelementName();
           String valores = new String("");
           for(int j = 1; j <= elementName._count(); j++){
              valores = valores.concat(elementName._getAt(j)+" ");
           }
           retorno.add(valores);
           elementName._close();
           elem._close();
           key = domainElements._next(key);
           sh = new StringHolder(key);
         }
         domain._close();
      }
      else{
        if(domain.gettype().equals("tema")) {
           vElementsTema = new Vector();
           for(int i = 0; i < num; i++) {
             Element elem = new Element(factory, domainElements._getObjectNext(sh));
             vElementsTema.add(elem._getOid());
             ListOfDataTypes elementName = elem.getelementName();
             String valores = new String("");
             for(int j = 1; j <= elementName._count(); j++) {
               valores = valores.concat(elementName._getAt(j)+" ");
             }
          …
          …
        domain._close();
      }
     }
    }
    catch (Exception e){
      e.printStackTrace();
    }
    return retorno;
  }
```

Figura 78 - Trecho de código do método queryDomainElements de classe SemanticInformationPanel

4.5. Criação de nova informação semântica

Para a definição da informação semântica de uma mídia (Figura 75) são utilizados os dados disponíveis na base de dados. Caso os dados existentes não sejam suficientes para expressar a informação semântica que o usuário deseja, novos dados devem ser inseridos na base de dados, sendo que essa inserção é feita através da interface apresentada na Figura 79.

A estrutura de classes criada para a implementação dessa interface e manipulação dos dados é mostrada na Figura 80. Essa estrutura é basicamente composta por uma lista (JList) que contém todos os elementos inseridos no domínio que está sendo criado e por uma tabela (JTable) que contém os valores das similaridades entre esses elementos. O algoritmo da Figura 81 descreve o comportamento da ferramenta para a inserção de novos dados no banco de dados.



\	arvore	arbusto	floresta	grama	capim	bonsai
arvore	1	0.7	0.9	0.3	0.4	
arbusto	0.7	1	0.7	0.6	0.8	
floresta	0.9	0.7	1	0.3	0.4	
grama	0.3	0.6	0.3	1	0.8	
capim	0.4	0.8	0.4	0.8	1	
bonsai						1

Figura 79 - Interface para a criação de novos dados para serem utilizados na inserção de informação semântica

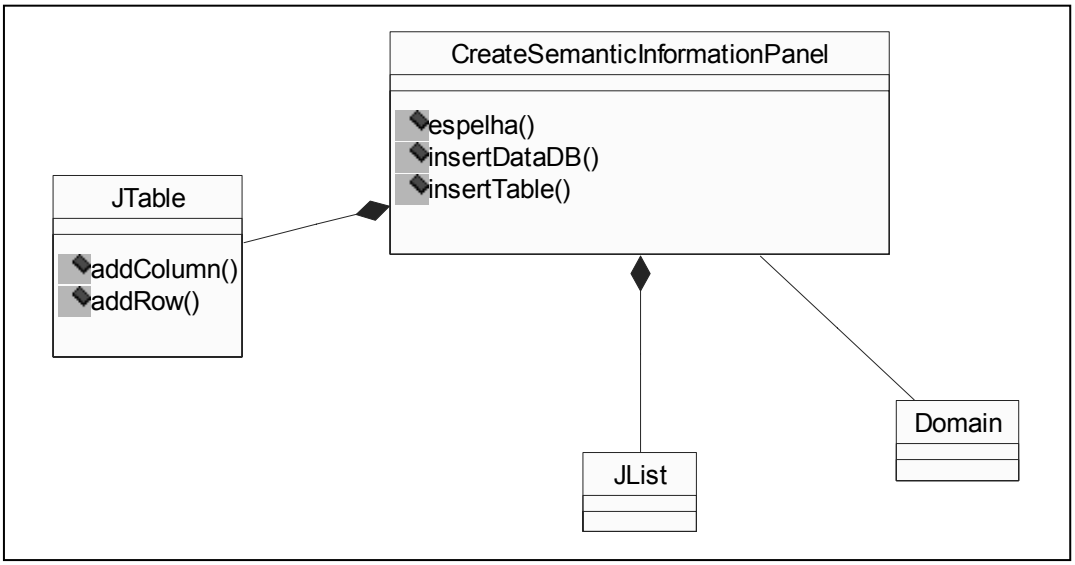


Figura 80 - Diagrama de classes para a inserção de novos dados

```
Capturar o nome do novo domínio D //inserido pelo usuário
Capturar o tipo do domínio D //"subject", "qualifier" ou "association"
Para cada elemento E a ser inserido no domínio D
inicio
    Capturar o nome do elemento E inserido pelo usuário
    Ao acionar o botão "AddElement>>" ativar o método insertTable( )// adiciona o
                                                    //elemento na tabela "Terms Similarity"
fim
Para cada célula da tabela "Terms Similarity"
inicio
    Capturar o valor da similaridade entre os elementos, inserida pelo usuário
    Ativar o método espelha( ) //replica o mesmo valor da similaridade na célula idêntica,
            //na interface do usuário (mesmos elementos linha e coluna, em ordem invertida)
fim
Através do acionamento do botão "Submit" ativar o método insertDataDB( ) //insere os
                                                                    //dados no BDMm
```

Figura 81 - Algoritmo para a inserção de novos dados

A Figura 82 apresenta o código do método insertDataDB da classe CreateSemanticInformationPanel, que recebe o nome do domínio a ser criado, os elementos que devem ser inseridos nesse domínio e o tipo ("subjects", "qualifiers" ou "association") do domínio. Esses dados são armazenados na estrutura de dados do BDMm (Figura 60).

```
private void insertDataDB(String dom, Vector temas, String tipo){
  try{
     Domain domain = new Domain(factory);
     RelationshipObject ro = domain.getelements();
     Iterator i = temas.iterator();//percorre os temas inseridos no dominio
     while(i.hasNext()){
       String aux = (String)i.next();
       Element element = new Element(factory);
       ListOfDataTypes lodt = new ListOfDataTypes(factory);
       lodt._insert(aux);
       element.setelementName(lodt);
       element._save();
       ro._insert(element);
       element._close();
       lodt._close();
     }
     domain.settype(tipo);
     domain.setname(dom);
     domain.setelements(ro);
     domain._save();
     ro._close();
     domain._close();
  }
  catch(Exception e){ e.printStackTrace(); }
}
```

Figura 82 - Código do método insertDataDB da classe CreateSemanticInformationPanel

4.6. Consultas e recuperação de mídias

A Figura 83 (área 1) mostra a interface utilizada para a consulta e recuperação de mídias inseridas no BDMm. Na construção de tais consultas são usados os dados semânticos inseridos no BDMm. Para a manipulação dos dados da interface e das informações fornecidas pelo usuário para a realização das consultas, foi desenvolvida a estrutura da Figura 84. Essa estrutura armazena em memória a informação semântica, inserida pelo usuário, utilizada para a consulta e recuperação das mídias.

A classe SubjectTemp armazena os temas (*identifier*) e seus qualificadores (*qualifiers*) enquanto que a classe AssociationTemp contém as associações, diretas (*directAssoc*) e reversas (*reverseAssoc*), entre esses temas. Essas informações são aquelas passadas pelo usuário para a realização da consulta. Já a classe ConsultaPanel está associada com algumas classes da estrutura de armazenamento de informação semântica do BDMm (Figura 5) e é através dela que são recuperados os domínios e seus elementos (getDomainElements(dominio: String)).

O algoritmo da Figura 85 descreve o comportamento da ferramenta para a obtenção e manipulação dos dados semânticos inseridos pelo usuário, utilizados na realização das consultas. O código do método query() da classe consultaPanel é apresentado na Figura 86.

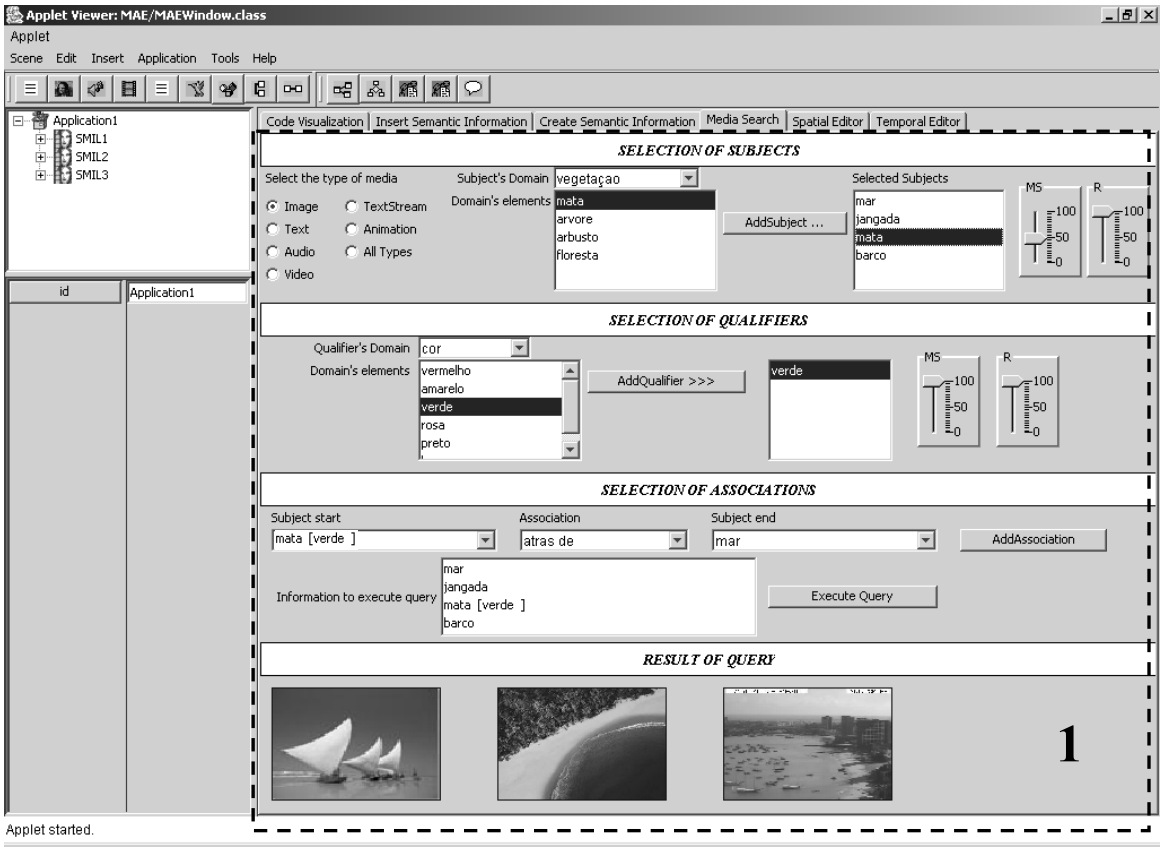


Figura 83 – Interface para a realização de consultas

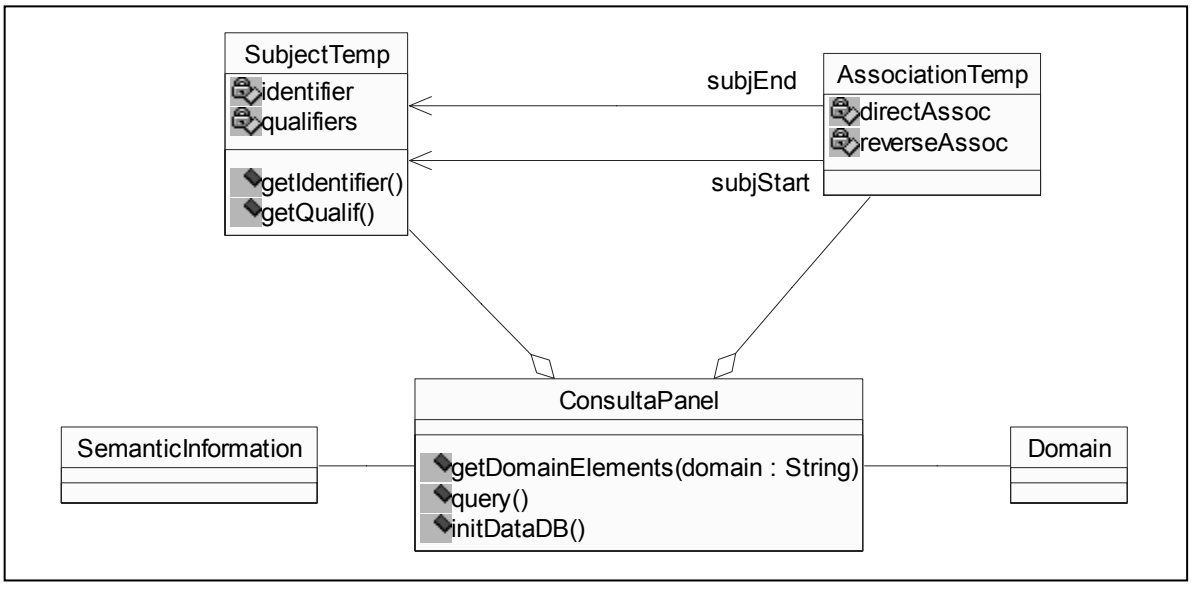


Figura 84 – Diagrama de classes para a realização de consultas

Ativar o método initDataDB( ) //recupera do BDMm os domínios existentes

Capturar o tipo de mídia //fornecido pelo usuário

Capturar um domínio D de *Subjects* //selecionado pelo usuário

Ativar getDomainElements(D) //recupera todos os elementos do domínio D

Para cada elemento do domínio D selecionado pelo usuário

Inicio

    Criar um objeto S da classe SubjectTemp

    Inserir o elemento selecionado no atributo tema do objeto S

    Atualizar a visualização da informação semântica

Fim

Capturar elemento E entre os selecionados //elemento escolhido pelo usuário

Capturar um domínio Q de *Qualifiers* //selecionado pelo usuário

Ativar getDomainElements (Q) //recupera todos os elementos do domínio Q

Para cada elemento X selecionado pelo usuário

Inicio

    Inserir o elemento X no atributo *qualifiers* do objeto S que contém o elemento E como tema

    Atualizar a visualização da informação semântica

Fim

Capturar um elemento I como *subjStart* //selecionado pelo usuário

Capturar um elemento A como *association* //selecionado pelo usuário

Capturar um elemento F como *subjEnd* //selecionado pelo usuário

Ao acionar o botão "Execute Query" ativar o método query( ) //Recupera as mídias do //BDMm

Figura 85 – Algoritmo para a realização de consultas

```
private void query(){
  try{
    ListOfObjects lo = new ListOfObjects(factory);
    Iterator i = listaInfSem.iterator();
    SubjectTemp noAtual;
    while(i.hasNext()) {  //percorre a lista de informação semântica
      noAtual = (SubjectTemp)i.next();
      ListOfObjects lo2 = new ListOfObjects(factory);
      Vector qualificadores = noAtual.getQualif();
      Iterator i2 = qualificadores.iterator();
      while(i2.hasNext()) {
        Element elemAux = new Element(factory, (Oid)i2.next());
        lo2._insert(elemAux);
        elemAux._close();
      }
      Element aux = new Element(factory);
      Element tema = new Element(factory, (Oid)noAtual.getTemaElem());
      lo = aux.recuperaMidias(tema,lo2);
      aux._close();
      if(lo == null) { System.out.println("Nao encontrou nenhuma midia"); }
      else {
        int coord_x = 10;
        for(int x=1;x<=lo._count();x++){
           BDMM.Image lido = (BDMM.Image)lo._getAt(x);
           int nrobytes = lido.getimg()._size();
           byte dados[] = lido.getimg()._getData();
           String name = lido.getname();
           String type = lido.gettype();
           String extension = lido.getextension();
           File fileSaida = new File("E:\\MAE\\Imagens\\"+name+"."+extension);
           DataOutputStream output = new DataOutputStream(new FileOutputStream(fileSaida));
           for(int j = 0; j < nrobytes; j++)
             output.writeByte((byte)dados[j]);
           output.close();
           ImgPanel img1 = new ImgPanel(fileSaida);
           ListOfObjects scenes = lido.getscenes();
           Vector te = new Vector();
           for (int k=1; k<=scenes._count();k++){
             BDMM.Smil teste = (BDMM.Smil)scenes._getAt(k);
             te.add(teste.getname());
           }
           img1.addPopMenu(te);
           semInfPanel.add(img1, 0);
           img1.setLocation(coord_x,35);
           coord_x += 200;
        }
      }
      tema._close();
    }
  }catch (Exception e){ e.printStackTrace(); }
}
```

Figura 86 - Código do método query( ) da classe ConsultaPanel

As consultas realizadas atualmente pela ferramenta não cobrem todas aquelas previstas no projeto AMMO. Como foi apresentado no Capítulo 5, não são usados nas consultas os

valores da similaridade e relevância dos elementos inseridos na expressão de busca. Além disso, não são abordados os operadores AND e OR na composição da expressão de busca. Estão sendo implementadas novas funcionalidades para cobrir essas questões, tornando possível a realização, através da ferramenta, de qualquer tipo de consulta suportada pelo projeto AMMO.