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

Utilizando XML para Publicação de Dados Multimídia na Web

Eduardo Cotrin Teixeira

São Carlos
Fevereiro
2002

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

T266ux

Teixeira, Eduardo Cotrin.
Utilizando XML para publicação de dados multimídia na web / Eduardo Cotrin Teixeira. -- São Carlos : UFSCar, 2005.
77 p.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2002.

1. Banco de dados. 2. XML (Linguagem de marcação de documentos). 3. Busca por conteúdo. 4. Metadados multimídia. I. Título.

CDD: 005.74 (20ª)

Índice

Índice de Figuras

Índice de Quadros e Tabelas

Resumo

Sistemas de gerenciamento de banco de dados orientados a objetos têm sido adotados para o gerenciamento de aplicações multimídia, tornando possível implementar formas de recuperação que propiciem maior poder de expressão ao usuário desde que sejam previstos metadados (informações sobre os dados armazenados) que auxiliem nesta tarefa, proporcionando ao usuário uma forma mais natural e poderosa para encontrar os objetos no banco de objetos. Nesse contexto surgiu o projeto AMMO *(Authoring and Manipulation of Multimedia Objects)*, com o objetivo de prover um ambiente para suporte à autoria, armazenamento e manipulação de aplicações multimídia. O trabalho desenvolvido tem como objetivo implementar a criação automática de documentos XML para representação de dados multimídia obtidos como resultado de consultas ao Servidor de Objetos Multimídia (MmOS - *Multimedia Objects Server*) do ambiente AMMO, preservando as vantagens presentes nesse ambiente, como possibilidade de busca por conteúdo, descrição semântica dos dados multimídia e alta reusabilidade, associando-as às vantagens de XML, como facilidade de intercâmbio, independência e flexibilidade na forma de apresentação e descrição semântica de conteúdo. São apresentados os conceitos de XML, o Projeto AMMO, a representação em XML dos dados multimídia com informações semânticas, a criação e o uso dos documentos XML dentro do Projeto AMMO e algumas formas possíveis de visualização dos dados através do uso de Folhas de Estilo XSL.

Abstract

Oriented-Object Database Management Systems have been adopted for the management of multimedia applications, becoming possible to implement recovery forms that propitiate more expression capacity to the user if are foreseen metadata (information about the stored data) that assist in this task, providing a more natural and powerful form to find objects in the objects base. In this context project AMMO (Authoring and Manipulation of Multimedia Objects) appeared, aiming to provide an environment for support the authoring, storage and manipulation of multimedia applications. The developed work aims to implement automatic XML document creation for representation of multimedia data obtained as result of consultations to the Multimedia Objects Server (MmOS) of AMMO environment, preserving the advantages of this environment, as possibility of content-based search, semantics description of multimedia data and high reusability, associating them with XML advantages, as interchange facility, independence and flexibility in presentation form and semantics description of content. The XML concepts, the AMMO Project, the XML representation of multimedia data with semantic information, the creation and use of XML documents inside AMMO Project and some possible forms of visualization of the data through the XSL Style Sheets are presented.

Capítulo 1

Introdução

1.1 – Considerações Iniciais

A informática, como ciência, evolui rapidamente, o que faz com que dispositivos e ferramentas considerados como tecnologia de ponta logo estejam ao alcance do usuário comum. Um dos efeitos desse fator é o barateamento e conseqüente popularização de elementos tecnológicos para armazenamento e manipulação de grandes quantidades de informações, incluindo aqui os equipamentos multimídia. Atualmente, a tecnologia multimídia se difundiu e vem sendo utilizada em diversas áreas da computação, aprimorando sistemas e aplicações tradicionais através do uso de imagens, áudio, vídeo e animações.

Porém, o progresso do uso de sistemas multimídia ainda depende da real absorção de atributos particulares às aplicações multimídia, como sincronização temporal dos elementos de mídia, manipulação de arquivos com grande volume de informações e alta interatividade. Segundo comentado em [Santos et al. 2000] e [Figueiredo 2000], o uso de padrões como SMIL *(Synchronized Multimedia Integration Language)* [SMILW3C 1998] e MHEG-5 (*Multimedia and Hypermedia Expert Group - Part 5)* [ISO 1996b], que conseguem representar de forma adequada a informação semântica de uma aplicação multimídia, torna-se imprescindível para minimizar os fatores de complexidade referentes ao gerenciamento e manipulação dessas aplicações.

O processo de autoria de aplicações multimídia também se apresenta como uma atividade crítica com relação ao custo computacional, pois inclui procedimentos como recuperação, análise, organização e manipulação de mídias. Uma das formas de amenizar esse processo é o reaproveitamento eficiente de mídias e aplicações entre ambientes diversos. Esse reaproveitamento pode ser facilitado através do uso de dados semânticos para descrição do conteúdo das mídias de forma a permitir consultas mais intuitivas e com alto poder de expressão. Muitos trabalhos têm seguido essa linha, com implementações de modelos como em [Vieira & Santos 1997] e [Vieira et al. 2002], implementações de ambientes como em [Cody et al. 1995] ou mesmo estudos sobre recuperação baseada em conteúdo como em [Aigrain et al. 1996].

Nesse sentido, a forma de armazenamento e gerenciamento torna-se um fator importantíssimo quando se trata de aplicações multimídia, já que devem suprir características próprias a dados multimídia e prover recursos para manutenção de metadados. Sistemas de Gerenciamento de Banco de Dados Orientado a Objetos (SGBDOO) apresentam atributos importantes para o gerenciamento de aplicações multimídia, devido à capacidade de manipulação de estruturas complexas, o que permite um tratamento mais adequado para tais aplicações [Ozden et al. 1997, Rakow et al. 1995].

Quando um SGBDOO é utilizado, torna-se possível implementar formas de recuperação que propiciem maior poder de expressão ao usuário desde que sejam previstos metadados (informações sobre os dados armazenados) que auxiliem nesta tarefa, disponibilizando ao usuário uma forma mais natural e eficiente de encontrar os objetos que deseja no banco de dados [Santos & Vieira 1998].

Com esse objetivo surgiu o projeto AMMO (*Authoring and Manipulation of Multimedia Objects*) [Santos et al. 1997a, Santos et al. 1997b, Vieira et al. 1999], para dar suporte à autoria, armazenamento e manipulação de aplicações multimídia.

Porém, outros aspectos também devem ser abordados, como a forma de disponibilização dos dados e das aplicações, considerando a Internet como forma de publicação de dados.

1.2 – Motivação

A Web cresceu e se tornou atualmente o principal meio para difusão de informações, incluindo aí dados multimídia. Um dos maiores propulsores desse grande crescimento foi, sem dúvida, a linguagem HTML (*Hyper Text Markup Language*), com uma sintaxe simples e compacta, de fácil aprendizagem e aplicação, que permitiu que autores distribuíssem seus documentos via Web de forma rápida e barata, com alcance internacional.

Com o passar do tempo, mais e mais documentos foram disponibilizados via Web, surgiram novas necessidades, os documentos tornaram-se mais complexos, e o que antes era visto como vantagem tornou-se limitação. A formatação fixa, que tornava HTML uma linguagem de fácil aprendizado, acentuou a inflexibilidade inerente à linguagem. A sintaxe simples tornou-se insuficiente para atender as necessidades e peculiaridades que surgiram com a grande difusão da WWW e das aplicações multimídia [Berners-Lee 1996].

Tornou-se necessária, então, a criação de novos padrões que não perdessem as vantagens outrora atribuídas a HTML, porém com características que permitissem suporte aos novos atributos exigidos para distribuição de documentos complexos e, principalmente, dados multimídia [Mace et al. 1998, Bulterman 1997].

Nesse contexto, novas tecnologias, linguagens e padrões surgiram, destacando-se a *Extensible Markup Language* (XML) [XMLW3C 1998], que tenta suprir essas necessidades de uma forma geral.

A XML surgiu da idéia de criação de uma linguagem que resolvesse os problemas apresentados por HTML para publicação de dados na Internet. Com XML, os projetistas podem criar elementos de acordo com sua aplicação, de maneira formal, porém rápida e fácil. Isso permite a geração de linguagens de marcação próprias, que podem descrever o conteúdo semanticamente de acordo com o objetivo da aplicação [Bryan 1998]. Desenvolvida para ser utilizada diretamente na Internet, a XML não se preocupa com a forma de apresentação da informação, mas sim com sua descrição e estruturação [Bosak & Bray 1999, Bosak 1997].

Conforme a implementação demonstrada em [Teixeira & Vieira 2001], que aborda um exemplo de criação automática de documentos XML com base em dados armazenados em um BDOO, e estudos como os de [Kleiner & Lipeck 2001] e [van Harmelen & Fensel 1999] que discutem a geração automática de documentos XML para representação de bancos de dados na Web a partir de esquemas conceituais, percebe-se que a facilidade de implementação e o alto poder de estruturação de XML tornam possível a representação direta de um objeto armazenado em um banco de dados, na forma de um documento XML naturalmente utilizável na Internet.

1.3 - Objetivos

O trabalho aqui apresentado está baseado em padrões de documento XML definidos especificamente para representar dados multimídia obtidos como resultado de consultas ao Banco de Dados Multimídia do Projeto AMMO. A geração automática de documentos baseados nesses padrões também é parte deste trabalho. São definidas formas de apresentação dos dados representados, aproveitando a flexibilidade de XML para representação dos dados, trazendo vantagens de acessibilidade ao ambiente, já que o padrão XML torna a formatação de visualização independente do conteúdo do documento, possibilitando a visualização dos dados recuperados com base em vários parâmetros, como por exemplo, tipo de usuário,

recursos disponíveis para apresentação, ou escolha do usuário quanto a um formato de documento específico. O presente trabalho propõe também uma arquitetura para utilização de documentos XML como intermediários para publicação na Web de dados obtidos a partir de um Banco de Dados Orientado a Objeto. A utilização de documentos XML como parte do processo de consulta proporciona ao ambiente a independência de plataforma, assim como torna mais flexível o intercâmbio dos dados e sua apresentação, características importantes na manipulação de dados multimídia [Bartlett 2001].

1.4 – Organização do documento

Este trabalho está organizado da seguinte maneira: o Capítulo 2 apresenta a linguagem XML e as tecnologias que a envolvem. O Capítulo 3 contextualiza o trabalho, apresentando o projeto AMMO e sua estrutura de gerenciamento de aplicações multimídia. O Capítulo 4 descreve a arquitetura definida para o uso de XML no processo de consulta, assim como os Padrões de Documento XML propostos. Nesse capítulo também é apresentada a implementação do mecanismo de geração automática dos documentos XML. A utilização dos documentos gerados é apresentada no Capítulo 5, que mostra formas de visualização dos documentos XML na Web. O Capítulo 6 conclui o trabalho e discute a importância da abordagem aqui apresentada.

Capítulo 2

XML (*Extensible Markup Language*)

2.1 - Introdução

Linguagens de Marcação são linguagens que utilizam marcas ou rótulos (*tags*) para descrever o conteúdo e estruturar a informação contida em um documento. Um exemplo de Linguagem de Marcação é a HTML, usada para formatar documentos Web.

Esse tipo de linguagem é baseado em SGML (*Standard Generalized Markup Language*) [ISO 1986], uma enorme metalinguagem padrão que surgiu antes da própria Web, e que define linguagens de marcação de documentos. SGML permite aos documentos descrever sua própria gramática, ou seja, especificar o conjunto de *tags* usado no documento e o que essas *tags* representam. Documentos HTML reúnem um subconjunto de *tags* (*tagset*) reduzido, em conformidade com a especificação SGML. Portanto, HTML é uma aplicação de SGML, ou seja, um conjunto fixo de *tags* com utilização e significado definidos.

As primeiras versões de HTML previam apenas a manipulação de texto, e não forneciam elementos para manipulação de imagens, gráficos ou outro tipo de mídia. Para sanar esse problema, foi necessária a criação de um novo padrão da linguagem que especificasse novas *tags* para manipulação de outros tipos de mídia além de texto, gerando uma nova versão da linguagem. A característica de possuir uma gramática fixa fez com que a linguagem fosse alterada várias vezes para se adequar às novas exigências de publicação de dados na Web. A versão de HTML adotada como referência neste trabalho é a 4.01, publicada em Dezembro de 1999.

Já a SGML, a metalinguagem que originou HTML, permite a criação e definição de novos elementos, o que resolveria o problema criado pela rigidez da HTML. Surgiu então uma versão mais enxuta que SGML e mais flexível que HTML: a XML.

Extensible Markup Language (XML) é uma linguagem voltada à descrição e marcação de dados, que foi projetada para ser um subconjunto de SGML, porém com a função específica de publicação de dados na Web, como HTML. Não é uma gramática fixa, como é HTML, mas sim extensível, como é SGML. XML foi criada para descrever o conteúdo dos documentos, e projetada para ser diretamente utilizável na Internet, conforme definição do Grupo de Trabalho XML do *World Wide Web Consortium* (W3C). Sua criação foi, portanto,

inspirada pela SGML, que permite a criação de tipos especializados de documentos; e pela HTML, que é diretamente utilizável na Internet. O projeto de XML procura unir os pontos mais fortes dessas duas linguagens, produzindo documentos que podem ser facilmente descritos, criados e manipulados mas também transferidos e visualizados na Internet.

A seguir a XML é apresentada através da identificação dos principais benefícios decorrentes de seu uso, tais como a forma de estruturação de um documento e a criação de sua DTD (*Document Type Definition* - Definição do Tipo de Documento). Um estudo mais detalhado sobre o padrão XML pode ser encontrado em [Teixeira & Vieira 2001].

2.2 - XML: Uso e Benefícios

Em 1996, um grupo de aproximadamente 80 pesquisadores, juntamente com o *World Wide Web Consortium* (W3C), formou uma equipe de trabalho para desenvolver uma linguagem que complementasse HTML, mas não necessariamente a substituísse.

Uma especificação inicial da linguagem foi divulgada ainda em 1996, com um segundo esboço publicado no início de 1997. Em julho de 1997 a equipe que trabalhava sobre a especificação foi formalizada nos padrões da W3C, e tornou-se um Grupo de Trabalho. O *World Wide Web Consortium* apresentou a XML 1.0 como recomendação no dia 10 de fevereiro de 1998 [Light 1999].

Os objetivos iniciais para o desenvolvimento da linguagem XML, definidos pelo Grupo de Trabalho W3C, foram:

- Ser diretamente utilizável na WWW.
- Suportar uma ampla variedade de aplicações.
- Ser compatível com SGML.
- Tornar fácil a criação de programas para interpretação dos documentos.
- Não possuir opcionais ou apêndices.
- Gerar documentos legíveis e razoavelmente claros.
- Ser ágil para criação de novos documentos.
- Ser de fácil implementação.

XML é extensível, permitindo que os autores de documentos Web criem suas próprias *tags*. Quando um documento XML é interpretado, ele pode trazer consigo a descrição de sua própria estrutura e o que ela significa, através do uso de DTDs (*Document Type Definition*). A presença de uma DTD permite que se controle a adição de novos elementos e estruturas ao documento. XML é flexível o suficiente para ser capaz de descrever qualquer estrutura lógica de texto, seja ele um memorando, uma carta, um livro ou uma enciclopédia.

XML foi desenvolvida para ser altamente expressiva, de fácil aprendizado e implementação. Segundo Bosak [Bosak 1997], XML deve ser utilizada principalmente em quatro amplas categorias:

- Aplicações para Web que acessem bancos de dados heterogêneos.
- Aplicações que distribuam a carga de processo do servidor para o cliente.
- Aplicações que exijam visões diferentes dos mesmos dados, dependendo do cliente.
- Aplicações voltadas para necessidades individuais de usuários.

A versatilidade, portanto, é uma das características marcantes de XML. XML cumpriu seu objetivo principal de estender os recursos para publicação de dados na Web [Sall 1998]. As características de XML trouxeram vários aspectos importantes:

1. Vocabulários de domínio específico: XML é usada para criar *tags* que podem ser entendidas por todos os navegadores com *parsers* (processadores) XML. Vocabulários XML podem estabelecer formatos padrão para envio de documentos e transações.
2. Dados estruturados: Qualquer aplicação compatível com XML pode aproveitar sua organização estruturada para extrair informações de documentos XML com exatidão.
3. Mecanismos de busca: Aumenta a probabilidade de que consultas recuperem arquivos relevantes, devido à informação contextual. Mecanismos de busca utilizando XML podem recuperar uma porção específica do arquivo, e se tornam mais rápidos se as informações de contexto ajudarem a eliminar numerosos resultados irrelevantes.

4. Visualização dos dados é independente da estrutura: O recurso de Folhas de Estilo (abordado na Seção 2.5) é utilizado para descrever como exibir o mesmo dado em diferentes formatos.

O Padrão XML, como um todo, é composto de várias especificações e outros padrões que o compõem. Para entender melhor o Padrão XML e seus documentos, pode-se considerar 2 componentes básicos, que tratam do documento em si e de sua apresentação: a *Linguagem XML*, que é a verdadeira especificação da linguagem, de sua sintaxe e conceitos; e o *XML Style (XSL-Extensible Stylesheet Language)*, que especifica Folhas de Estilo, as quais permitem que os documentos sejam moldados para mostrar os mesmos dados em formatos variados. A próxima Seção detalha a Linguagem XML, que é responsável por caracterizar o padrão XML da forma como foi descrito até aqui, pois é através da linguagem propriamente dita que são implementadas todas as características e vantagens citadas.

2.3 - A Linguagem XML

XML permite a criação de um número ilimitado de diferentes linguagens de marcação para propósitos diversos. O ponto principal é que todas as várias linguagens para propósitos específicos definidas em XML podem ser interpretadas por um pequeno processador simples e padronizado, dentro de qualquer navegador Web.

XML fornece uma sintaxe simples e legível para publicação na Web de qualquer tipo de dados estruturados, incluindo dados inter-relacionados, de uma forma que permite a manipulação e visualização dos dados através de ferramentas simples e padronizadas.

O documento XML é formado de *tags* baseando-se no conceito de documentos compostos por uma série de entidades. Cada entidade pode conter um ou mais elementos lógicos. Cada um desses elementos pode ter certos atributos (propriedades). XML fornece uma sintaxe formal para descrever as relações entre as entidades, elementos e atributos que formam um documento. A Linguagem XML permite a definição das *tags* de forma que essas descrevam o conteúdo do documento, e não sua formatação. O exemplo a seguir mostra um documento XML onde as *tags* foram definidas na DTD (pelo comando `ELEMENT`) para descrever o conteúdo de um memorando.

```
<?xml version="1.0" standalone="yes" encoding="ISO-8859-1"?>
<!DOCTYPE memo[
<!ELEMENT memo(from, to, date, subject?, para+)>
<!ELEMENT para (#PCDATA) >
<!ELEMENT to (#PCDATA) >
<!ELEMENT from (#PCDATA) >
<!ELEMENT date (#PCDATA) >
<!ELEMENT subject (#PCDATA) >
]>
<memo>
     <from> De: Diretor </from>
     <to> Para: Funcionários </to>
     <date>    1 de Setembro   </date>
     <subject> Ref. Feriado  </subject>
     <para> Fica decretado feriado para todos os funcionários,
     de todos os setores, no dia 9 de setembro do corrente ano,
     por motivo de comemoração do aniversário da
     empresa.</para>
</memo>
```

DTD

Este exemplo é exibido no Internet Explorer 5.5 no seguinte formato:

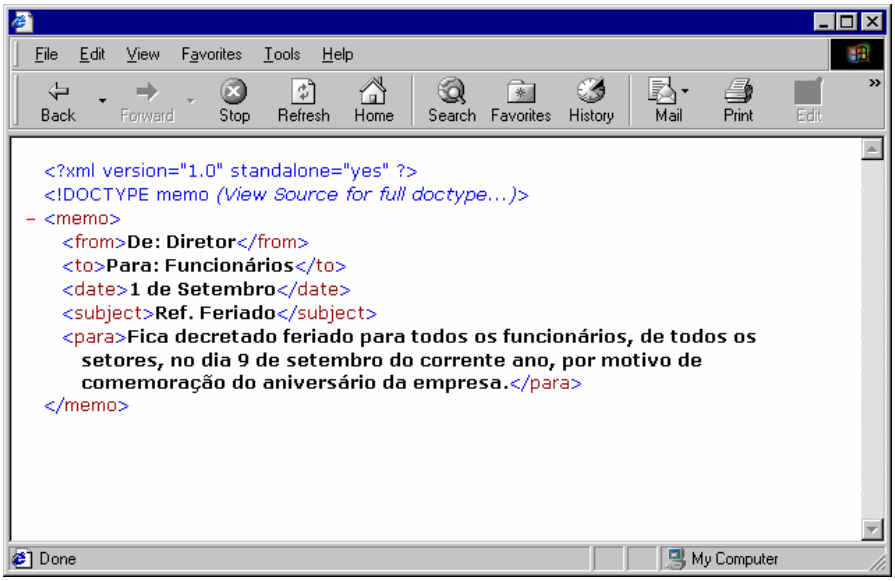


Figura 2.1 - Exemplo de documento XML

A informação de cabeçalho mostra ao *software* cliente como processar o documento XML. A DTD especifica a estrutura lógica de um documento, permitindo definir sua gramática, o que vai possibilitar a um *parser* (analisador) de XML validar o documento. A DTD define também os elementos de um documento e seus atributos, e a relação entre eles. Por exemplo, a DTD do documento da Figura 2.1 especifica que deve haver um parágrafo (`para`) ou mais (indicado pelo símbolo `+`) formando o conteúdo do texto, e deve estar posicionado depois dos elementos de cabeçalho, conforme a ordem especificada na linha:

```
<!ELEMENT memo(from, to, date, subject?, para+)>
```

As DTDs nos documentos XML definem como o documento deve ser escrito e estruturado. Surge então o conceito de *documento XML bem formado*, que seria o documento perfeitamente estruturado. Todos os elementos são posicionados e declarados de forma correta, mas a estrutura lógica não precisa necessariamente acompanhar as regras descritas na DTD. Há ainda o *documento XML válido*, que é o documento XML bem formado que segue todas as regras lógicas descritas na DTD. O documento apresentado na Figura 2.1 é um documento XML válido, já que segue a estrutura especificada em sua DTD interna.

2.3.1 - Componentes do documento XML

Os principais componentes de um documento XML são: *a declaração XML*, que corresponde a linha inicial do documento, sendo compreendida por ferramentas compatíveis com XML; a *declaração de DTD*, que referencia a DTD externa e/ou interna, definindo o padrão do documento; e o documento em si, ou *texto marcado*, onde o conteúdo é descrito através de marcação [Walsh 1997].

Declaração XML

Todo documento XML deve começar com uma declaração para ser apresentado apropriadamente às ferramentas compatíveis com XML. Essa declaração (chamada de "Declaração XML") define, entre outras coisas, a versão de XML que está sendo utilizada. Como a única e atual versão de XML é a 1.0, todas as Declarações XML incluem o texto: `version="1.0"`. Deve declarar também o tipo de caracter que está sendo usado.

Toda Declaração XML deve começar com a seqüência de caracteres: `<?xml`. Para especificar o tipo de caracter em uso, deve-se levar em conta que caracteres são codificados em vários formatos, geralmente dependendo da linguagem usada no documento. A definição de codificação de caracter para indicar, por exemplo, UTF-8 é: `encoding="UTF-8"`.

A seguir tem-se um exemplo de Declaração XML, na qual `standalone` indica a presença ou não de declarações de marcação (DTDs) externas. O valor **yes** para `standalone`, indica que não há DTD externa direta ou indiretamente. Um valor **no** indica que há DTD externa.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

Esta Declaração XML determina que o documento é compatível com XML 1.0, codificado em caracter UTF-8 e que sua estrutura não segue nenhuma DTD externa.

Declaração de DTD

A Declaração de DTD pode apontar para um arquivo externo de uma DTD, e/ou incluir declarações de marcação internamente, sendo mais comum apontar para uma DTD externa e incluir, também, definições internas, o que permite sobrepor ou acrescer declarações internas à DTD externa. Documentos bem formados e documentos XML válidos com declaração `standalone="yes"` são forçados a fornecer as declarações de marcação internamente.

Declarações de DTD obedecem ao formato:

```
<!DOCTYPE NOME TIPO-IDENTIFICADOR IDENTIFICADOR [ DECLARAÇÕES-SUBGRUPO-INTERNO ]>.
```

As declarações internas e seu conteúdo, marcadas pelos caracteres `[` e `]`, são opcionais. Um exemplo válido de Declaração de DTD:

```
<!DOCTYPE EXEMPLO SYSTEM "exemplo.dtd">
```

Essa Declaração de DTD nomeia um tipo de documento como EXEMPLO e aponta para sua definição em um arquivo externo através de um identificador SYSTEM.

Já a declaração a seguir nomeia um tipo de documento e o define internamente entre os caracteres [e].

```
<!DOCTYPE EXEMPLO [
<!ELEMENT EXEMPLO (TITULO,PARA+)>
<!ELEMENT TITULO   (#PCDATA)>
<!ELEMENT PARA     (#PCDATA|PALCHAVE)*>
<!ELEMENT PALCHAVE (#PCDATA)>
]>
```

As duas formas podem ser usadas em conjunto. Definições internas sobrepõem eventuais definições de elementos já existentes.

O Texto Marcado

A marcação descreve e organiza o conteúdo através do uso de várias estruturas XML, como *elementos*, *atributos* e *entidades de referência*. Com relação a sintaxe, marcação é qualquer parte do documento que começa com < ou &. Ela descreve conteúdo, mas não é lida como caracter de dados.

Elementos

Elementos são recipientes de conteúdo para descrever texto. Cada bloco de texto deve ser descrito por um elemento, para definir seu propósito e sua função. Elementos constroem a estrutura de documentos através da descrição explícita de cada porção dele, usando *tags-início* e *tags-fim* para delimitar e descrever seu conteúdo. Obedecem à forma: <NOME> CONTEÚDO </NOME>.

Elementos podem também ter a forma de *tags de elemento vazio*, que não contêm nenhum conteúdo, exceto atributos, e obedecem à forma: <NOME/>.

Todo documento XML começa com um elemento raiz, que não pode ser utilizado em nenhum outro lugar do documento. <HTML> é o elemento raiz em documentos HTML, abrindo e fechando cada documento. Todos os elementos dentro do elemento raiz devem ser

organizados apropriadamente, o que significa que cada elemento filho deve estar dentro de seu elemento pai e que os elementos não podem se misturar uns com os outros.

Atributos

Atributos são uma forma de descrever elementos e seu conteúdo. Eles permitem aos autores de documentos uma descrição detalhada, o que os elementos não fornecem sozinhos. Cada atributo pode conter um ou mais valores, dependendo de sua definição, mas não podem conter sub atributos. Atributos existem dentro de *tags*-início e *tags* de elemento vazio e obedecem ao formato:

```
<NOME ATRIB="valor"> CONTEÚDO </NOME>
```

Entidades

XML possui cinco entidades pré-definidas, que são representadas pelo símbolo `&` seguido do nome e ponto e vírgula, apresentadas na Tabela 2.1 abaixo.

Entidade	Caracter
&amp;	&
&lt;	<
&gt;	>
&apos;	'
&quot;	"

Tabela 2.1 - Entidades pré-definidas para documentos XML

Entidades podem também ser usadas para referenciar ou armazenar qualquer conteúdo do documento, como texto, texto marcado, declarações de marcação, etc. São geralmente usadas para texto e marcação que variam muito ou que são reproduzidos com freqüência. Entidades obedecem ao formato: `&nome;`.

Esta seção demonstrou a estrutura de um documento XML, apresentando seus componentes. A próxima seção aborda a forma como o conteúdo da DTD é escrito para definir as regras que o documento XML deve seguir.

2.3.2 - O conteúdo da DTD

Declarações de marcação formam o conteúdo da DTD, definindo cada bloco do documento, determinando onde esse bloco é válido e como deve ser usado. Declarações de marcação são: *declarações de elementos*, *declarações de atributos*, *declarações de entidades* e *declarações de notações*. Tudo o que é declarado na DTD pode ser aplicado nos documentos na forma de marcação.

Declarações de Elementos

Declarações de Elementos são o mais importante tipo de declaração de marcação, já que definem a estrutura de documentos através da definição dos elementos que o documento pode possuir. Elementos podem ser declarados tendo como conteúdo somente outros elementos, somente texto, ou conteúdo misto, o que inclui texto e elementos. Declarações de Elementos obedecem ao formato: `<!ELEMENT NOME CONTEÚDO>`. Na marcação seguem o formato: `<NOME> CONTEÚDO </NOME>`.

Um outro recurso utilizado para a declaração de elementos é o uso de *indicadores de ocorrência,* que definem o número de vezes que um grupo ou elemento pode ser usado. Há quatro tipos de indicadores de ocorrência, com diferentes graus de opção:

- sem símbolo, define como requerido;
- +, define como requerido e repetível;
- *, define como opcional e repetível, e;
- ?, define como opcional e não repetível.

O exemplo a seguir mostra uma declaração de conteúdo misto, pois pode conter dados do tipo texto (PCDATA - *Parsed Character Data*) ou elementos PALCHAVE. O operador de escolha (caracter `|`) requer que somente uma seção seja interpretada. Poderia ser usado também o operador de seqüência (caracter `,`) que requer que cada seção seja interpretada, na ordem especificada.

```
<!ELEMENT TITULO (#PCDATA|PALCHAVE)*>
```

Essa declaração de marcação pode ser aplicada através da seguinte marcação:

```
<TITULO> Linguagem <PALCHAVE> XML </PALCHAVE> </TITULO>
```

Declarações de Atributos

Declarações de Atributos podem ser consideradas como um mecanismo de metadados, já que descrevem o conteúdo e o comportamento de um elemento. Atributos são usados para descrever propriedades de elementos, como a localização de objetos de dados, a altura e largura de imagens ou tabelas, ou a linguagem do texto. Atributos obedecem ao formato: `<!ATTLIST NOME-ELEMENTO NOME-ATRIBUTO TIPO PADRÃO>`. Sua aplicação na marcação ocorre em *tags*-início ou *tags* de elemento vazio, no formato:

`<NOME-ELEMENTO NOME-ATRIB="valor"> CONTEÚDO </NOME-ELEMENTO>` ou `<NOME-ELEMENTO NOME-ATRIB="valor"/>`.

O exemplo a seguir mostra uma Declaração de Elemento (`ELO`) com a declaração de um atributo (`alvo`) para definir um elemento de ligação.

```
<!ELEMENT ELO (#PCDATA)>
<!ATTLIST ELO alvo CDATA #REQUIRED>
```

Essas declarações podem ser aplicadas na marcação da seguinte forma:

```
<ELO alvo="http://www.dc.ufscar.br"> UFSCar </ELO>
```

Declarações de Entidades

Declarações de Entidades definem dados reusáveis. Entidades podem ser classificadas como *Gerais* ou *de Parâmetro*. Entidades Gerais podem ser interpretadas no documento XML, enquanto Entidades de Parâmetro contêm dados reusáveis somente na DTD. Entidades gerais obedecem ao formato: `<!ENTITY NOME DEFINIÇÃO>`, sendo que sua aplicação na marcação obedece ao formato: `&NOME;`. Entidades de parâmetro obedecem ao formato: `<!ENTITY % NOME DEFINIÇÃO>`, e sua aplicação na DTD é: `%NOME;`.

Um exemplo de Declaração de Entidade Geral, que associa uma *string* com um nome de entidade:

```
<!ENTITY cor "verde">
```

Essa declaração é aplicada na marcação da seguinte forma:

```
<PARA>Minha cor favorita é &cor;.</PARA>
```

As Entidades de Parâmetro possuem uso similar, porém são utilizadas apenas internamente na DTD.

Declarações de Notação

Declarações de Notação descrevem o formato dos objetos de dados. São normalmente usadas para descrever o formato de objetos binários, como arquivos gráficos. Podem também ser usadas para descrever dados do tipo caracter, ou outros formatos. Notações seguem o formato `<!NOTATION NOME SYSTEM ID>`.

A Declaração de Notação abaixo define um formato de dados para uso em outras declarações de marcação.

```
<!NOTATION GIF SYSTEM "GIF Notation">
```

2.4 - XML-Style (XSL - Extensible Stylesheet Language)

As marcações XML não trazem consigo informações sobre a forma de visualização do documento. Para apresentação dos dados, XML apresenta o recurso de *Folhas de Estilo* (*style sheets*), que é o mecanismo que permite que o cliente visualize um mesmo documento de várias formas diferentes. Trocando a Folha de Estilo, o mesmo documento pode mudar a apresentação dos dados contidos na página. É um recurso que traz flexibilidade ao autor e comodidade e maior interatividade ao usuário. Como exemplo de aplicação que ilustra essas vantagens podemos citar a publicação, na Web, de um manual técnico que se adapta ao nível de aprendizagem do usuário. A partir de uma mesma base de texto pode haver um estilo para novatos, diferente do estilo para profissionais.

No documento de memorando mostrado na Figura 2.1, poderíamos aplicar a seguinte Folha de Estilo CSS *(Cascading Style Sheets)* [CSSW3C 1996], armazenada em um arquivo externo ao documento, para alterarmos a sua visualização:

```
PARA {display: inline}
SUBJECT, FROM, TO, DATE {display: block}
FROM {font-size: 1.3em}
DATE {color: #FF0000; font-style: italic}
PARA {font-style: italic; margin: 0.5em}
SUBJECT{text-decoration: underline; font-weight: bold}
```

Essa Folha de Estilo utiliza a propriedade `display` para especificar que os parágrafos (`PARA`) devem ser exibidos sem quebra de linha, e os demais elementos com quebra de linha. As propriedades como `font-size`, `color`, `font-style` e `margin` alteram características como tamanho da fonte, cor, estilo e posição.

O cabeçalho do documento XML deve, então, possuir uma referência a esse arquivo. Isso é feito através da inserção de uma linha com uma instrução de processamento, que indica o arquivo a ser adotado como Folha de Estilo:

```
<?xml version="1.0" standalone="yes"?>
<?XML:stylesheet type="text/css" href="memo.css"?>
<!DOCTYPE memo[
<!ELEMENT memo (to, from, date, subject?, para+) >
<!ELEMENT para (#PCDATA) >
<!ELEMENT to (#PCDATA) >
<!ELEMENT from (#PCDATA) >
<!ELEMENT date (#PCDATA) >
<!ELEMENT subject (#PCDATA) >
<!NOTATION GIF SYSTEM "">
]>
```

O documento XML da Figura 2.1, alterado para utilizar a Folha de Estilo, é exibido no Internet Explorer 5.5 da seguinte maneira:

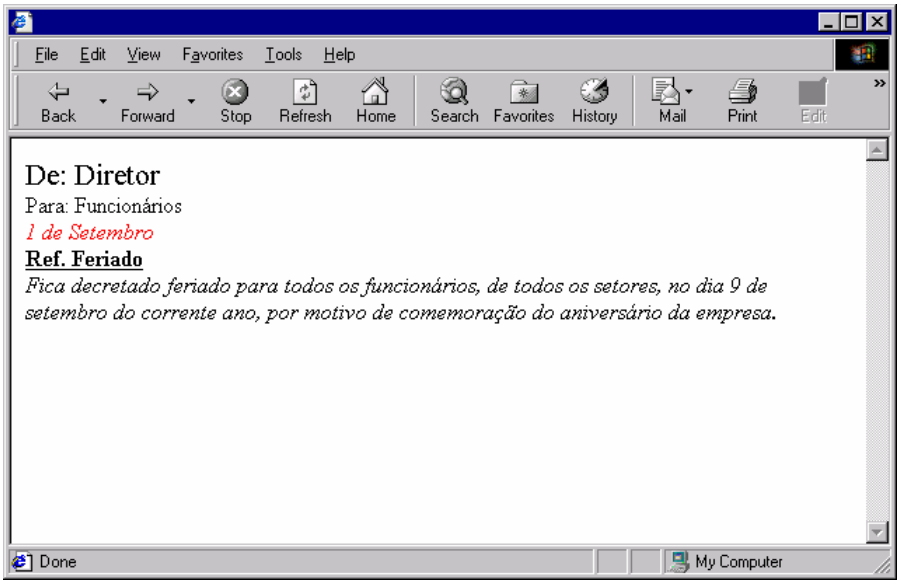


Figura 2.2 - Exemplo de documento XML com Folha de Estilo

A linguagem adotada como padrão inicialmente (e utilizada no exemplo) foi a CSS (*Cascading Style Sheets*, ou Folhas de Estilo em Cascata). Projetada com base em HTML, com a qual também é utilizada, CSS utiliza nomes e identificadores de elementos naturalmente encontrados em HTML para definir a forma de exibição. Outra linguagem de Folhas de Estilo utilizada foi a *Document Style Semantics and Specification Language* (DSSSL) [ISO 1996a], projetada para uso com SGML. DSSSL tem poder para criação de projetos avançados, porém apresenta algumas desvantagens, como uma sintaxe muito difícil de ser aprendida e a falta de uma camada de declaração mais rica.

Pelo fato de não terem sido projetadas com base em XML, algumas limitações surgiram com relação à CSS e DSSSL, e tornou-se necessária a criação de uma linguagem de Folha de Estilos própria para XML, que aproveitasse todo seu potencial. Surgiu então a XSL (*Extensible Stylesheet Language*) [XSLW3C 2001], como meio próprio para Folhas de Estilo XML.

A XSL foi baseada no Padrão DSSSL, combinando o poder do DSSSL com a simplicidade da XML e as vantagens da CSS. A XSL é uma adaptação desses padrões dentro da especificação mais enxuta do XML.

Quando um documento XML é carregado, uma ou mais Folhas de Estilo também são carregadas. A Folha de Estilo XSL pode ser associada com o documento XML de várias formas:

- O documento XML pode trazer consigo a definição da Folha de Estilo.
- Pode ser usado um processador XSL para promover a transformação criando um arquivo de resultado.
- A Folha de Estilo pode ser associada dinamicamente na ferramenta de apresentação.

A decisão sobre qual ou quais Folhas de Estilo carregar pode ficar a cargo do usuário ou pode ser feita através de uma especificação dentro do próprio documento XML. O processamento do documento é feito, então, através do exame da estrutura do documento XML com relação à especificação definida nas Folhas de Estilo ativas. A partir daí é então gerado o *layout* do documento e esse pode ser visualizado.

A XSL possui recursos completos de linguagem, como suporte para cálculos, testes de condições, estruturas de comando e até funções. A especificação da formatação de elementos ou até de caracteres do documento pode ser feita utilizando-se esses recursos.

A XSL fornece total controle sobre todos os elementos, permitindo, por exemplo, que o tamanho da fonte e espaçamento de uma determinada porção do texto sejam calculados através de uma expressão matemática, ou mesmo que uma determinada parte do texto não seja exibida.

2.5 - Considerações Finais

Este capítulo abordou a linguagem XML, focando principalmente dois elementos: o documento XML e as DTDs. O objetivo foi apresentar conceitos presentes no trabalho desenvolvido, como a forma de definição de DTDs e a aplicação dessas DTDs em documentos XML. O próximo capítulo visa contextualizar o trabalho desenvolvido, apresentando o projeto AMMO (*Authoring and Manipulation of Multimedia Objects*) e a forma de gerenciamento das aplicações multimídia armazenadas.

Capítulo 3

Aplicações Multimídia no Projeto AMMO

3.1 - Introdução

O trabalho aqui apresentado tem como objetivo principal a definição de DTDs e o desenvolvimento e implementação de algoritmos para geração automática de documentos XML que representam objetos pertencentes a aplicações multimídia que estão armazenadas em um Banco de Dados Multimídia. Esses objetos são recuperados por consultas feitas através de informações semânticas mantidas sobre os elementos que compõem as aplicações multimídia armazenadas. Os dados recuperados são então representados na forma de documentos XML, tirando proveito das vantagens do padrão XML para apresentação com grande flexibilidade, intercâmbio de informações, alta estruturação e facilidade de manipulação e interpretação de documentos.

A implementação da geração automática dos documentos XML está integrada ao projeto AMMO *(Authoring and Manipulation of Multimedia Objects)*, cujo objetivo é prover um ambiente para suporte à autoria, armazenamento e manipulação de aplicações multimídia baseadas no padrão SMIL *(Synchronized Multimedia Integration Language)*. A base do projeto AMMO provém do projeto SMmD (*Sistemas Multimídia Distribuídos*) (ProTeM-CC II - CNPq) [Teixeira et al. 1995] [Teixeira et al. 1998] através do subprojeto denominado *"Modelagem e Gerenciamento de Objetos Multimídia em Ambientes Distribuídos"*. O trabalho aqui apresentado envolve alguns módulos do Projeto AMMO, apresentados na Figura 3.1.

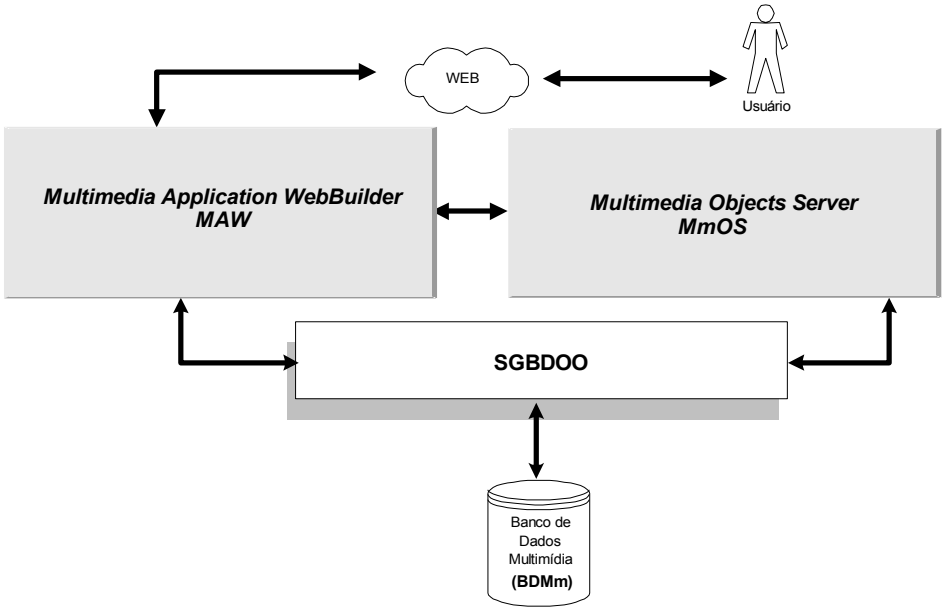


Figura 3.1 - Módulos do Projeto AMMO (*Authoring and Manipulation of Multimedia Objects*)

O ***Multimedia Objects Server (MmOS)*** é responsável por gerenciar as aplicações multimídia armazenadas na base de dados, que é mantida por um ***Sistema Gerenciador de Banco de Dados Orientado a Objetos (SGBDOO)***.

O ***Banco de Dados Multimídia (BDMm)*** e o ***Multimedia Application WebBuilder (MAW)*** representam, respectivamente, a base de dados multimídia propriamente dita e o módulo de autoria e publicação de aplicações multimídia na Web. Devido à sua relevância para este trabalho, o BDMm e o MAW serão abordados com mais detalhes na seção a seguir.

3.2 - O Banco de Dados Multimídia (BDMm)

O projeto AMMO utilizou, inicialmente, o padrão MHEG-5 [ISO 1996b] para representação das aplicações multimídia armazenadas no BDMm. Porém, com o amadurecimento do projeto e sua conseqüente evolução, um dos principais objetivos tornou-se o acesso e a publicação dos dados multimídia através da Web. Visando atender esse objetivo, foi adotado o padrão SMIL.

SMIL é uma proposta da W3C para construção de aplicações multimídia. O padrão SMIL [SMILW3C 1998] foi definido através de XML como um conjunto de marcações que permitem organizar elementos multimídia compondo uma apresentação para ser visualizada na Web. As apresentações criadas com SMIL definem a forma como as mídias devem ser

apresentadas, com relação a sua disposição, aparência, duração e sincronização temporal e espacial.

O padrão SMIL foi inserido no projeto como uma forma de representação das aplicações multimídia armazenadas para que essas fossem acessíveis na Web, o que impulsionou a criação de um novo ambiente dentro do projeto AMMO que manipulasse as aplicações armazenadas nesse novo formato, e permitisse sua integração com os outros módulos do projeto.

O novo ambiente foi denominado MAW (*Multimedia Application WebBuilder*) [Macedo 1999] e, inicialmente, possuía o objetivo específico de conversão das aplicações armazenadas no formato MHEG-5 para aplicações baseadas no padrão SMIL, utilizando o SGBDOO O2 [O2 1995], em plataforma UNIX. Posteriormente a implementação do MAW migrou para o SGBDOO Jasmine (Computer Associates) em plataforma Windows NT, adotando para o Banco de Dados Multimídia um formato baseado no padrão SMIL.

Para a manipulação das aplicações baseadas em SMIL, foi desenvolvida uma estrutura de classes que representa as aplicações multimídia criadas. O primeiro trabalho de modelagem dessa estrutura foi desenvolvido por Figueiredo [Figueiredo 2000], porém o modelo proposto foi reformulado de forma a facilitar sua manipulação. A versão atual da estrutura de classes para armazenar as aplicações multimídia é apresentada em notação UML [OMG 2001] na Figura 3.2, tendo como base a estrutura de documentos definida pelo padrão SMIL 1.0, com a diferença de que o conceito de aplicação multimídia, aqui, é diferente do conceito utilizado pelo padrão SMIL, onde cada documento é considerado uma aplicação multimídia. No MAW um documento SMIL é considerado uma cena dentro da aplicação, fazendo com que uma aplicação multimídia, no ambiente, seja considerada como um conjunto de cenas (ou de documentos SMIL).

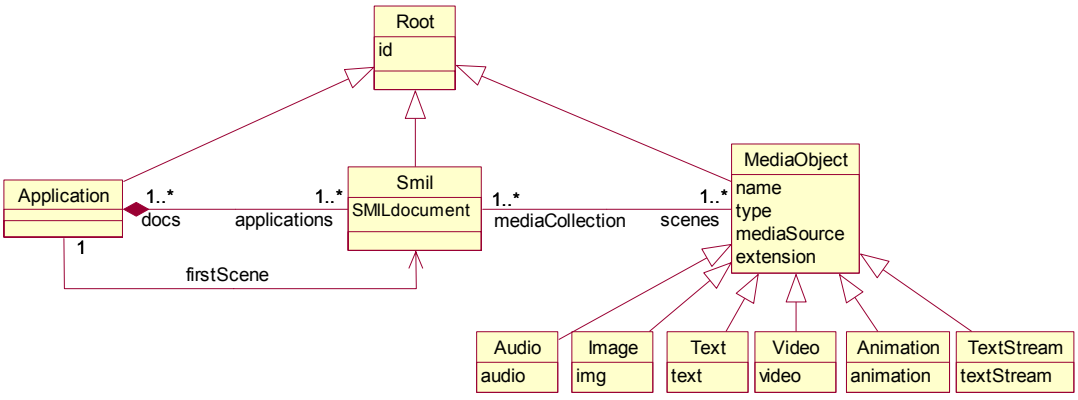


Figura 3.2 - Representação em UML de aplicações multimídia no AMMO

As aplicações multimídia são modeladas como um conjunto de documentos SMIL. Cada documento SMIL representa uma cena da aplicação, e é composto por uma coleção de objetos de mídia, que podem ser de vários tipos (áudio, vídeo, texto, etc.).

Na estrutura apresentada, a classe *Root* mantém identificadores (atributo *id*) para cada objeto das aplicações armazenadas. A classe *Application* representa uma aplicação multimídia armazenada no BDMm através da agregação *docs*, que agrupa os documentos SMIL que formam a aplicação, e da referência *firstScene*, que indica qual documento SMIL representa a cena inicial da aplicação, sendo esta a cena executada quando for solicitada a execução da aplicação completa. A classe *Smil* representa uma cena através do atributo *SMILDocument*, e associa essa cena com uma ou mais aplicações através da associação *applications*. Ainda na classe *Smil*, a relação entre a cena e suas mídias é feita pela associação *mediaCollection*, que referencia todos os objetos de mídia que compõem a cena.

Cada mídia é representada através da classe *MediaObject*, que é especializada (*Audio*, *Image*, *Text*, *Video*, *Animation* e *TextStream*) para reproduzir a forma como o padrão SMIL classifica as mídias componentes de um documento. Através da associação *scenes*, a classe *MediaObject* indica a quais cenas (ou documentos SMIL) essa mídia está relacionada. Além dessa informação, há ainda atributos para indicar o endereço físico em que a mídia se encontra (*mediaSource*), tipo da mídia (*type*), nome (*name*) e extensão do arquivo (*extension*).

O Banco de Dados Multimídia (BDMm) do projeto AMMO possui ainda um conjunto de classes para representar informações semânticas sobre cada mídia armazenada, o que possibilita consulta através de dados semânticos (busca por conteúdo). A forma como as informações semânticas estão estruturadas e mantidas é o assunto da próxima seção.

3.3 - Informações Semânticas das mídias armazenadas

Em um banco de dados multimídia é útil manter não somente os arquivos de mídia, mas também informações sobre seu conteúdo. Pela sua característica predominantemente visual e sensitiva, os dados multimídia tornam-se mais facilmente acessíveis através de suas características de apresentação. Informações semânticas (metadados) permitem maior flexibilidade para o usuário construir consultas [Vieira et al. 2002, Santos et al. 2000].

Conforme já citado, as aplicações multimídia armazenadas no BDMm são tratadas como um conjunto de cenas e essas, por sua vez, possuem um conjunto de mídias que as compõem. Nesse contexto, a mídia é o objeto básico de composição das aplicações,

representando a informação propriamente dita. No momento de recuperação de dados armazenados no BDMm, as mídias tornam-se, portanto, o ponto principal de atenção. Cada mídia está associada a uma estrutura de informações semânticas, possibilitando ao usuário consultar o BDMm através de consultas exatas ou nebulosas baseadas em conteúdo. As informações armazenadas são referentes às características diretamente encontradas nas mídias pelo autor da aplicação, que possui a função de inserir tais informações no momento de criação da aplicação multimídia. As características consideradas relevantes são classificadas como *temas*, *qualificadores* e *associações entre os temas*, fazendo com que a informação semântica armazenada esteja relacionada diretamente com cada mídia através dos temas que a descrevem e as relações entre esses temas. Uma ferramenta de autoria SMIL vem sendo desenvolvida especificamente para o ambiente MAW, prevendo as funcionalidades necessárias ao processo de autoria. Uma primeira versão da ferramenta é apresentada em [Figueiredo 2000].

Para exemplificar a informação semântica que é armazenada, considere a imagem mostrada na Figura 3.3. A imagem contém os temas *Morro*, *Lago*, *Prédio* e *Pássaro*, entre outros. O tema *Morro* pode ser qualificado como *alto*, *sem vegeta*ção, etc.

Figura 3.3 - Exemplo de imagem armazenada no BDMm

Além disso, podem ser encontradas algumas associações entre os temas, tais como: Prédio *próximo* Morro, Morro *atrás* Prédio, Lago *com* Pássaro, entre outras. Essas informações semânticas são armazenadas no BDMm, de acordo com o conjunto de classes ilustrado na parte (a) da Figura 3.4. Esse conjunto de classes está associado à estrutura de classes de aplicações multimídia através de uma referência à classe *MediaObject* da Figura 3.2 (representada na parte (b) da Figura 3.4). Para cada mídia de uma aplicação multimídia pode-se definir informações semânticas para possibilitar a recuperação dessa mídia e, conseqüentemente, as cenas e aplicações relacionadas a ela.

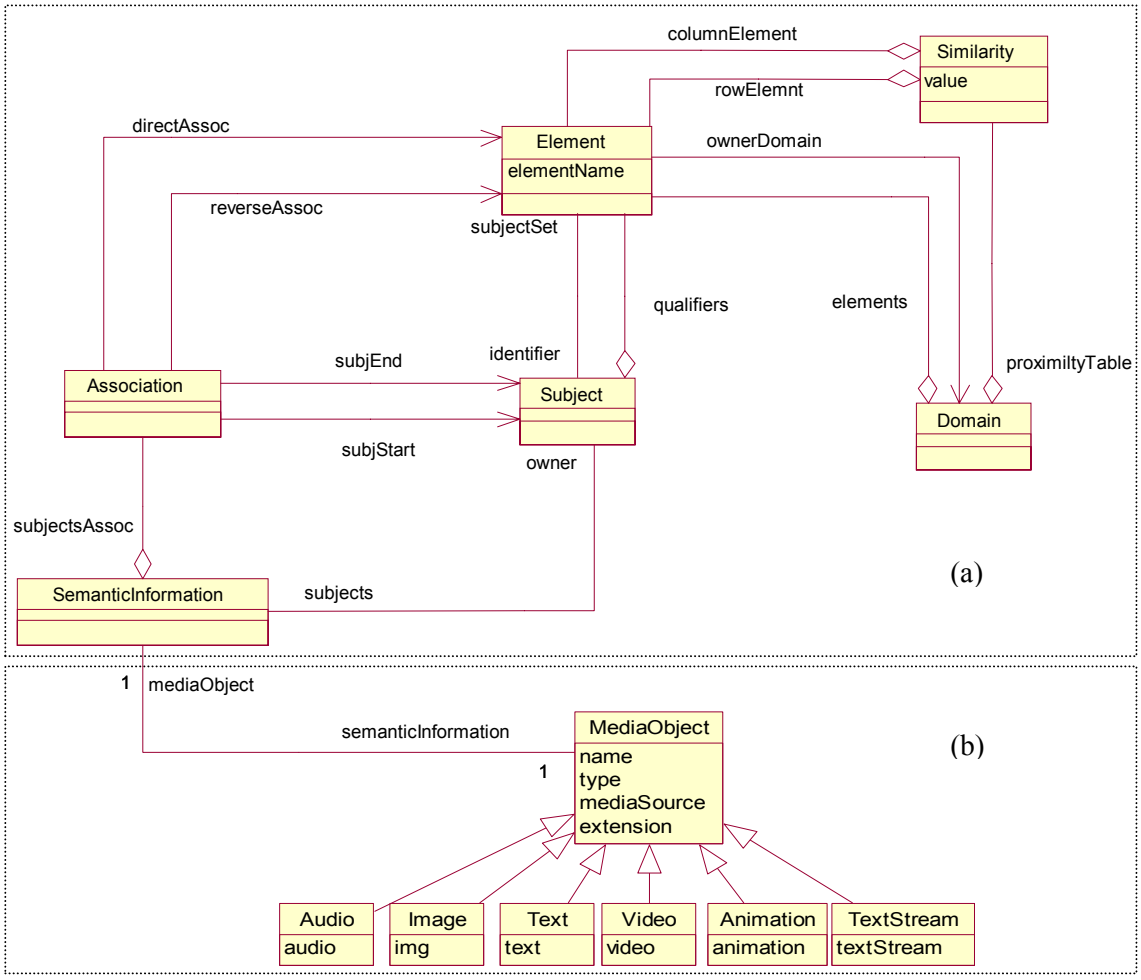


Figura 3.4 - Representação das informações semânticas no AMMO

Na estrutura de representação da informação semântica de uma mídia a classe *SemanticInformation* se relaciona diretamente com a mídia (*MediaObject*) mantendo os temas presentes nas mídias através da associação *subjects.* As associações entre os temas são armazenadas na classe *Association,* que é referenciada pela classe *SemanticInformation* através da agregação *subjectsAssoc*. A classe *Association* representa as associações na forma de associações diretas e associações reversas. As associações diretas são compostas por um tema inicial, indicado pela referência *subjStart,* por um tema final representado pela referência *subjEnd* e um ou mais termos de associação direta (*directAssoc*). As associações reversas são compostas da mesma forma, porém os termos de associação são indicados por *reverseAssoc*. A classe *Subject* mantém, para cada tema, um identificador (*identifier*) e uma lista de adjetivos que o qualifica (*qualifiers*), mantidos na classe *Element*. A classe *Element* define domínio dos temas, qualificadores e associações através da referência *ownerDomain*. A classe *Domain* foi criada para agrupar temas, qualificadores e associações em domínios, estabelecendo relações de proximidade entre eles. Cada domínio mantém vários elementos

equivalentes, que possuem valores que estabelecem a similaridade entre eles. Esses valores são armazenados através da classe *Similarity* e constituem informação importante para o processamento de consultas nebulosas, não sendo objeto de estudo deste trabalho.

Para exemplificar de forma mais prática o armazenamento de aplicações multimídia com informações semânticas no BDMm, a próxima seção apresenta uma aplicação armazenada no BDMm do projeto AMMO, composta de várias cenas e diversos tipos de mídia.

3.4 - Aplicação multimídia armazenada no BDMm

A utilização prática da geração automática de documentos XML, foco deste trabalho, se dá no momento em que uma consulta é realizada e são recuperadas várias mídias, de tipos diferentes, que devem ser apresentadas ao usuário. Para exemplificar essa situação, é apresentada aqui uma aplicação multimídia que aborda o assunto de Robótica. Essa aplicação é composta de várias cenas na forma de documentos SMIL, que é o padrão adotado no Projeto AMMO para representação das aplicações multimídia armazenadas (conforme exposto na seção 3.2). A aplicação é estruturada na forma de um manual visual sobre Robótica, e tem como objetivo apresentar ao usuário os principais conceitos dessa área da tecnologia através de imagens, textos, áudios, vídeos e animações. Devido à grande variedade e quantidade de elementos de mídia envolvidos, essa aplicação mostrou-se adequada para o uso aqui proposto.

A aplicação multimídia denominada “Robótica: Robôs Universais” é composta de uma cena de apresentação (*presentation.smi*), um menu principal (*main.smi*) e cenas que são disparadas através do menu principal, que apresenta quatro opções: Conceitos (*concepts.smi*), História (*history.smi*), Aplicações (*applications.smi*) e Ética (*ethics.smi*). Alguns desses assuntos são representados por uma cena única, enquanto outros possuem várias cenas para detalhamento do assunto. A Figura 3.5 apresenta um exemplo de cena presente na aplicação.

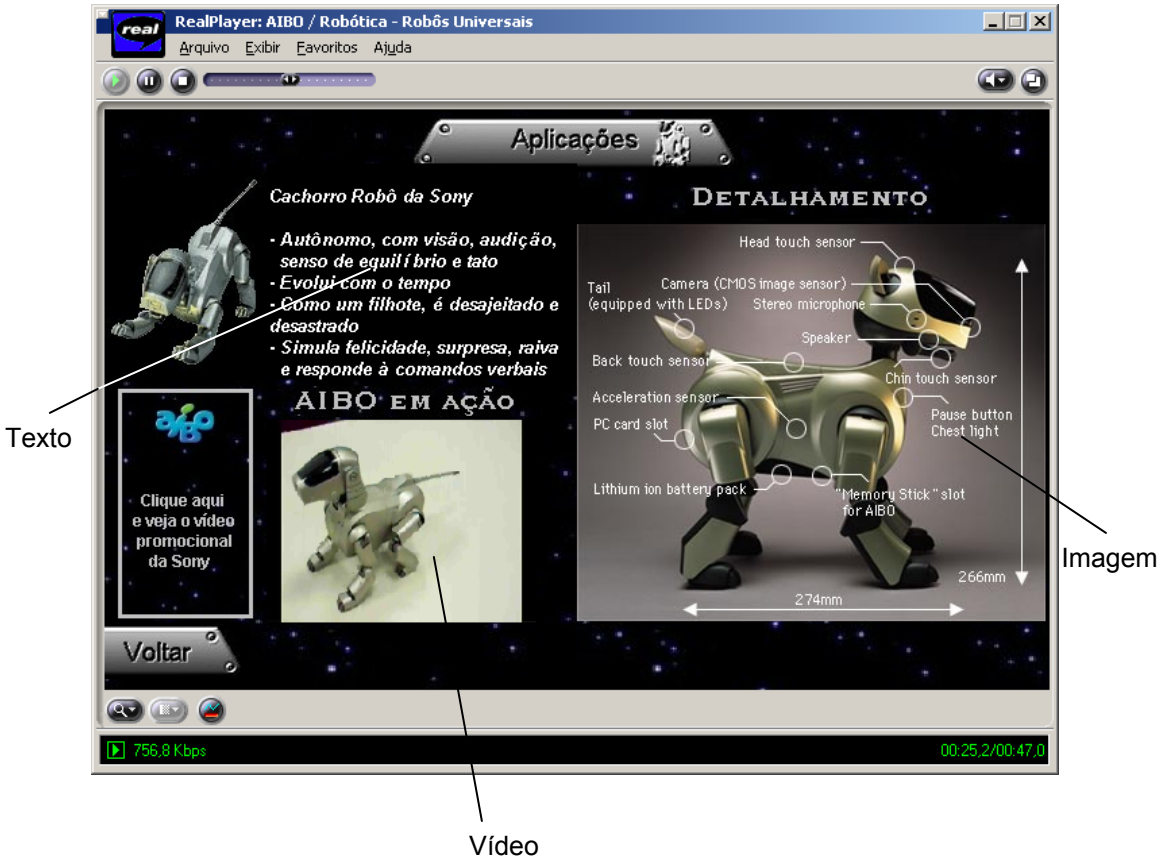


Figura 3.5 - Exemplo de cena com dados multimídia

A aplicação em questão, que contém a cena representada na Figura 3.5, pode ser armazenada no BDMm do projeto AMMO conforme a estrutura previamente apresentada (Figura 3.2). Para instanciação da estrutura de armazenamento da aplicação multimídia são criados vários objetos da classe *MediaObject,* representando cada mídia presente na aplicação. Tomando como exemplo de mídia o vídeo presente na cena apresentada na Figura 3.5, o objeto *MediaObject* referente a ele é instanciado através da atribuição dos dados referentes a mídia, na classe de especialização *Video*:

MediaObject (especialização *Video*)

name: aibo.avi	type: Video
mediaSource: /robotics/video/aibo.avi	extension: avi

A cena, por sua vez, é instanciada como um objeto da classe *Smil*, através da atribuição do nome do arquivo de cena (*appaibo.smi*) ao atributo *SMILDocument*.

Esse mesmo processo é repetido para todas as mídias e cenas da aplicação. Porém, isso ainda não define a aplicação como um todo. Após a instanciação dos objetos de mídia e das

cenas, deve ser feito o relacionamento entre eles. Cada mídia é relacionada às cenas a que pertence através da associação *scenes*, e de forma similar a cena (objeto *Smil*) deve referenciar todos os objetos de mídia que a compõem através da associação *mediaCollection*. No caso de adição posterior de mídias a uma determinada cena o mesmo processo deve ser repetido, ou seja, o objeto de mídia inserido deve ser relacionado a cena que pertence, assim como o objeto de cena (objeto *Smil*) deve incluir o novo objeto através da associação *mediaCollection.* No exemplo apresentado, o objeto *Smil* referente à cena *appaibo.smi* deve referenciar o objeto *MediaObject* referente ao vídeo *aibo.avi,* sendo necessária também a associação inversa desse objeto de mídia para a cena em questão. A cena deve ainda fazer referência às aplicações a que pertence, através da associação *applications*. Para que isso seja possível, o objeto da classe *Application* deve estabelecer a associação *scenes,* que referencia o conjunto de objetos *Smil* que representam as cenas que compõem a aplicação. Deve ser feita também a indicação da cena inicial (*presentation.smi*) através da referência *firstScene*.

Após esse processo, teremos o seguinte panorama no BDMm: um objeto *Application* referencia vários objetos *Smil* representando os arquivos de cenas da aplicação (*presentation.smi*, *main.smi*, *concepts.smi*, *applications.smi*, etc.). Cada objeto *Smil* faz referência a um ou vários objetos *Application* e a objetos *MediaObject* das mídias que o compõem (*aibo.avi*, *aibo.txt*, *details.jpg*, etc.). Os objetos da classe *MediaObject* representam os arquivos de mídia através de suas informações (tipo, extensão, etc.) e referenciam as cenas (*Smil*) em que estão contidas.

Nesse ponto do processo, a aplicação está devidamente armazenada no BDMm, porém não há informações semânticas mantidas sobre as mídias da aplicação para tornar possível a realização de buscas por conteúdo.

Assim como os dados da aplicação foram devidamente armazenados no BDMm, também as informações semânticas devem ser armazenadas de acordo com a estrutura apresentada na Figura 3.4. As informações semânticas são inseridas no momento da autoria, porém não é necessário inserir informações de todas as mídias presentes na aplicação, já que componentes complementares como fundos de tela, rótulos e outros detalhes de composição não representam objetos importantes de consulta.

Para exemplificar a definição de informações semânticas das mídias presentes na aplicação, a Figura 3.6 apresenta outra cena da aplicação multimídia.

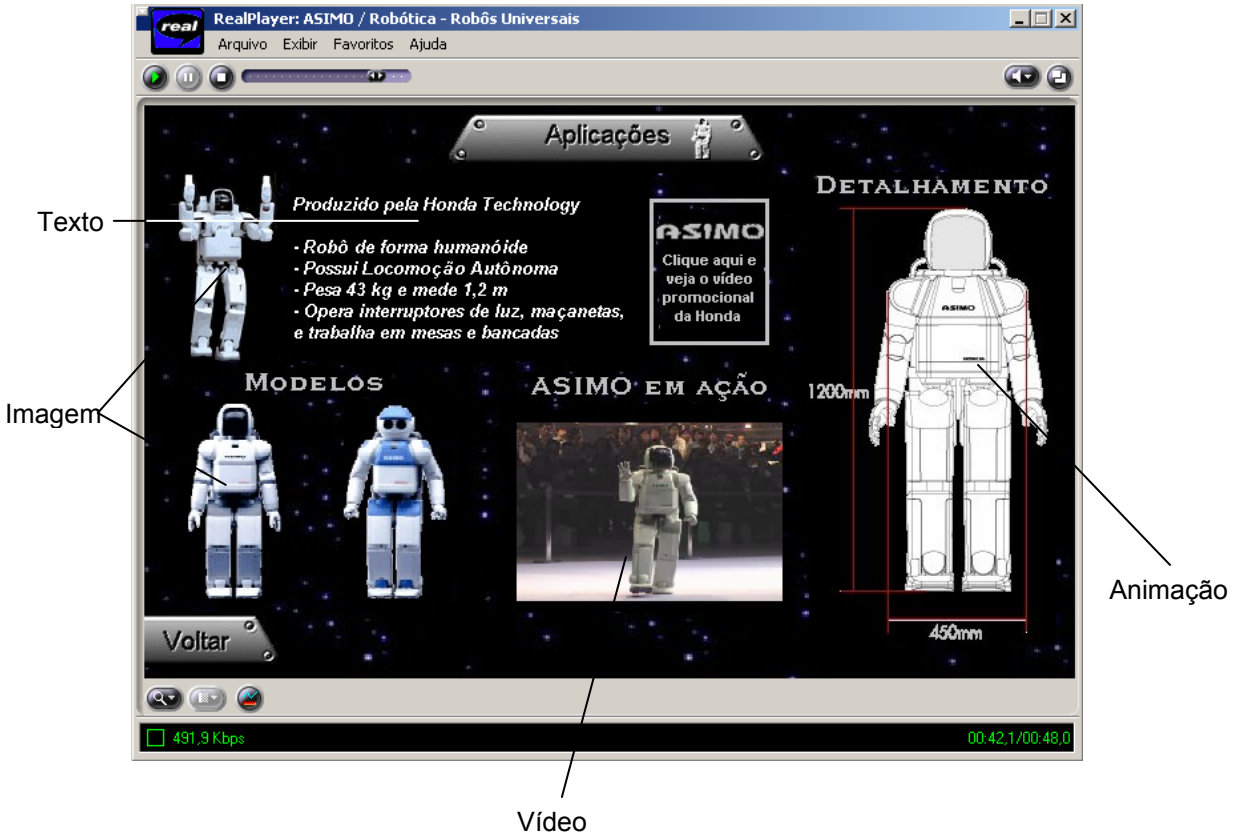


Figura 3.6 - Exemplo de cena com dados multimídia

Na Figura 3.6, as mídias destacadas (com indicação do tipo) são aquelas que apresentam maior relevância para a apresentação do conteúdo da cena e, portanto, serão as tratadas com informações semânticas. Ressalta-se que o processo de definição e inserção das informações semânticas depende do ponto de vista do autor. Todos os dados semânticos inseridos podem receber tratamento de informações imprecisas, pois dependem muito do ponto de vista de cada pessoa quanto à relevância das mídias e dos temas utilizados [Borges 2001].

Na cena da Figura 3.6, é apresentado um vídeo onde um robô se exibe para o público. Dessa forma, nesse vídeo podem ser identificados os temas *Robô* e *Pessoas*. Para qualificar melhor o robô que aparece no vídeo, podemos qualificá-lo como um robô *Branco* e *Humanóide*. Esses dados ainda podem ser relacionados através da percepção de que o robô está *a frente* das pessoas. Chegamos então à seguinte descrição:

Robô branco humanóide a frente de pessoas

tema qualificadores associação direta tema

Formatando esses dados para a estrutura apresentada na Figura 3.4, podemos inserir as seguintes informações semânticas a respeito do vídeo:

Subjects: Robô, Pessoas
Qualifiers: Humanóide, Branco (*Subject Robô)*

Associations:	*subjStart* - Robô Branco Humanóide	*subjStart* - Pessoas
	directAssoc- a frente de	*reverseAssoc*- com
	subjEnd- Pessoas	*subjEnd*- Robô Branco Humanóide

Todos os temas (s*ubjects*), qualificadores (*qualifiers*) e associações (*directAssoc* e *reverseAssoc*) inseridos são instanciados como elementos, sendo objetos da classe *Element*. A composição na forma de temas, qualificadores e associações é feita através da correta referenciação dentro da estrutura de classes. Os elementos inseridos podem ser agrupados em domínios (*Domain*), como por exemplo *Cor* para o qualificador *Branco*, ou *Posicionamento* para a associação *a frente de*.

3.5 - Considerações Finais

Este capítulo apresentou o projeto AMMO e as estruturas definidas em seu ambiente para o armazenamento e manipulação de aplicações multimídia baseadas no padrão SMIL, prevendo a recuperação de mídias através de consultas baseadas em informações semânticas mantidas no Banco de Dados Multimídia. Foi apresentada também uma aplicação multimídia sobre Robótica, e a forma como ela é armazenada no BDMm, com um exemplo de informação semântica descrevendo o conteúdo de uma das mídias que a compõem. Com isso, foi apresentado o contexto onde está inserido o trabalho aqui apresentado, assim como os principais conceitos que o norteiam. O próximo capítulo aborda o processo de consulta e geração dos documentos XML para representação das mídias recuperadas do BDMm do projeto AMMO.

Capítulo 4

Representando dados multimídia com XML

4.1 - Introdução

O primeiro passo para o desenvolvimento deste trabalho foi a definição de DTDs XML que representam os dados multimídia com informações semânticas obtidos como resultado de consultas ao Banco de Dados Orientado a Objetos do MmOS (*Multimedia Objects Server*) do projeto AMMO. A geração automática de documentos XML baseados nessas DTDs também é parte deste trabalho, cujo objetivo final é aprimorar a apresentação dos resultados da consulta via Web e criar um padrão para intercâmbio de informações multimídia com dados semânticos, tirando proveito de vantagens do padrão XML como alta estruturação e facilidade de manipulação e interpretação de dados. Foram implementados também mecanismos que permitem a visualização dos dados de várias formas diferentes, demonstrando as vantagens e a flexibilidade de publicação de dados XML com relação aos dados multimídia que são objeto deste trabalho. Essa abordagem permite aproveitar as vantagens existentes no modelo do projeto AMMO, criando documentos Web que apresentam dados multimídia e informações semânticas associadas. No projeto AMMO, a consulta ao Banco de Dados é feita via Web, e os dados obtidos como resposta à consulta realizada são automaticamente transformados em conteúdo do documento XML, e só então são apresentados ao usuário. Como o padrão XML torna a formatação de visualização independente do documento, é possível construir inúmeras formas de visualização dos dados consultados com base em vários parâmetros, como, por exemplo, tipo de usuário, recursos disponíveis para apresentação, ou formato de documento desejado.

A utilização de documentos XML como parte do processo de consulta proporciona ao ambiente a independência de plataforma e implementação, assim como torna mais flexível o intercâmbio dos dados consultados e sua apresentação, características importantes na manipulação de dados multimídia. A arquitetura usada no processo de consulta é representada a seguir, na Figura 4.1:

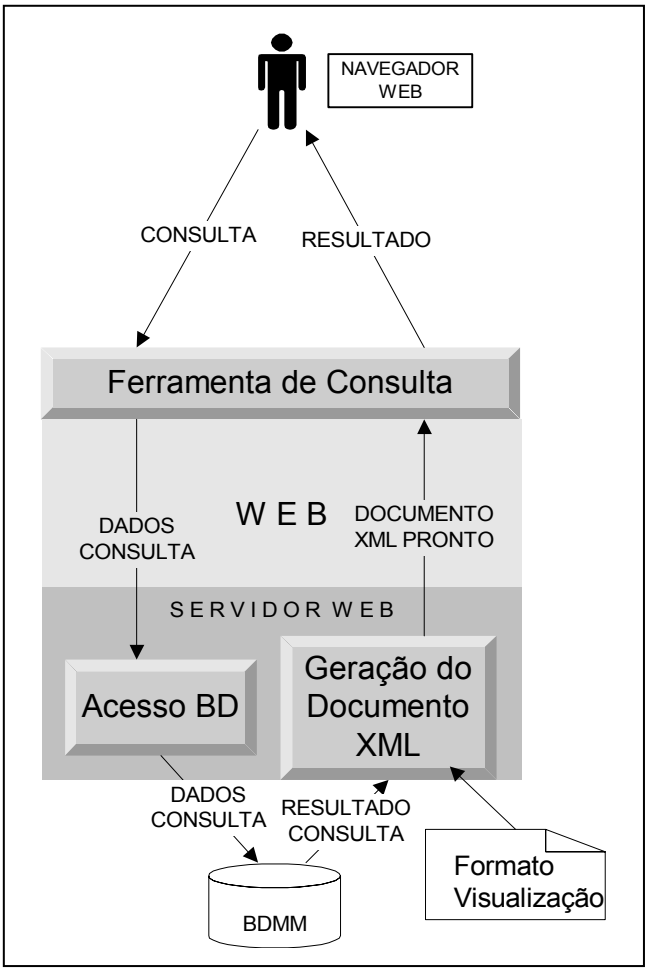


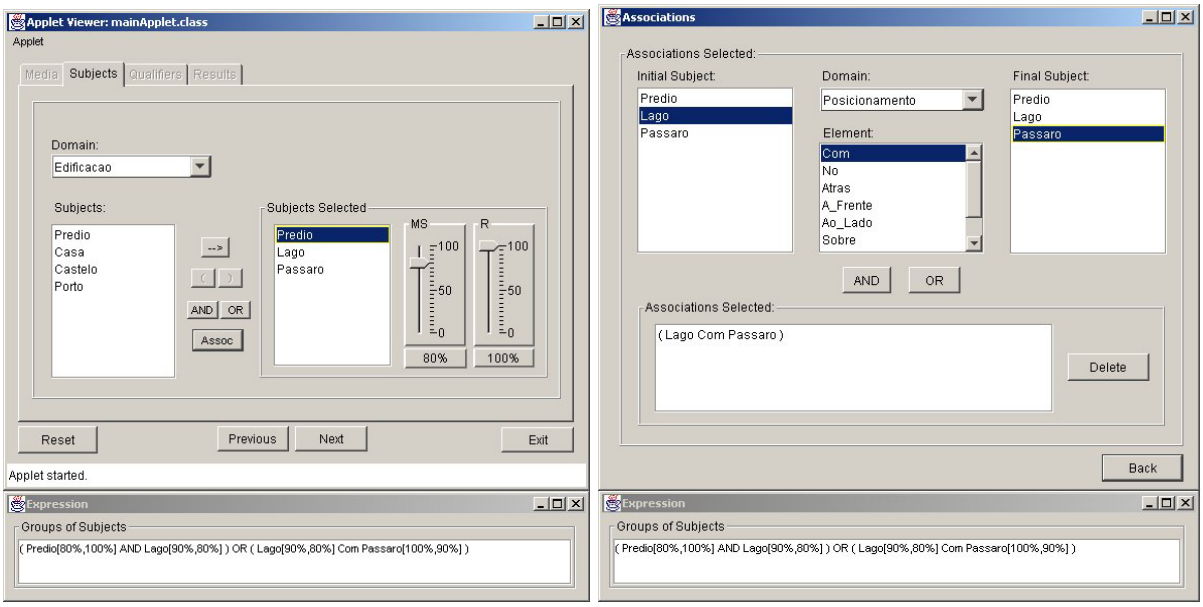
Figura 4.1 - Arquitetura do Processo de Consulta

Com base nessa arquitetura, o processo de consulta pode ser dividido conforme os passos que são executados desde a entrada da expressão de consulta até a apresentação dos resultados ao usuário, a saber: *Elaboração da Expressão de Consulta*, *Recuperação dos Dados*, *Geração do documento XML* e, por fim, *Formatação e Apresentação dos Resultados*. O presente trabalho tem como foco principal a definição de padrões (DTDs) nos quais o documento XML está baseado e, conseqüentemente, os dois últimos passos: *Geração do Documento XML,* detalhado nas Seções 4.4 a 4.6, e *Apresentação dos Resultados,* detalhado no Capítulo 5. Os passos anteriores (*Elaboração da Expressão de Consulta* e *Recuperação dos Dados*) já foram desenvolvidos [Borges 2001] e são apresentados brevemente nas próximas seções.

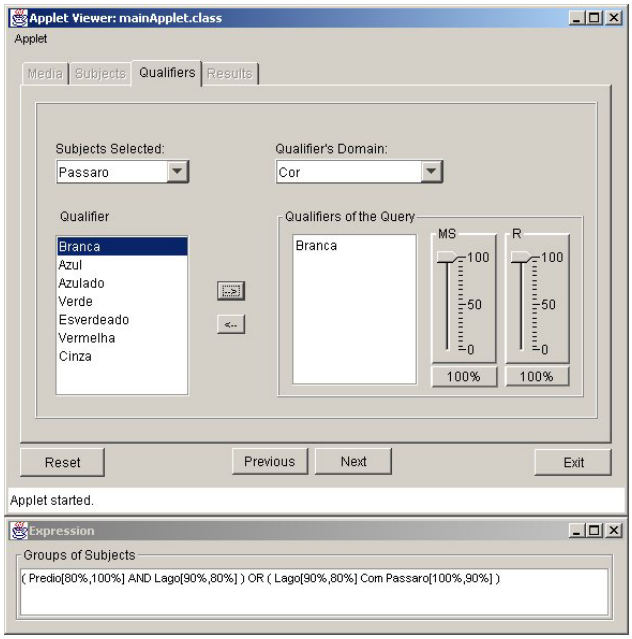
4.2 - Elaboração da Expressão de Consulta

A consulta ao BDMm do projeto AMMO é feita através de sentenças que envolvem dados semânticos das mídias. Portanto, a primeira etapa para recuperação dos dados deve ser a elaboração da expressão de consulta a ser submetida ao BDMm. A construção da sentença de consulta pelo usuário, para recuperação das mídias, é feita através do *Sistema de Consultas Nebulosas*, que permite realizar consultas envolvendo as informações semânticas agrupadas através de conectores *AND* e *OR*. O Sistema de Consultas Nebulosas pode ser acessado por qualquer navegador compatível com *applets* Java (Figura 4.2). Um exemplo de consulta seria: "*Recupere as imagens que possuem Prédio Alto e Lago Azul ou Lago Azul com Pássaro Branco*". No ambiente, o usuário pode informar os temas desejados, no caso "*Prédio*", "*Lago*" e "*Pássaro*" (Figura 4.2 - I), os qualificadores para cada um dos temas, como "*Alto*", "*Azul*" e "*Branco*" (Figura 4.2 - II), e as associações entre os temas (Figura 4.2 - III). Adicionalmente, pode informar a similaridade mínima (MS) desejada para cada tema e qualificador, e a importância de cada um deles para a consulta, através da definição de um grau de relevância (R). Se o usuário não estabelecer graus de similaridade, a consulta tem o comportamento de uma busca exata. É obtido como resultado da consulta o conjunto de mídias que satisfazem a consulta elaborada com os parâmetros definidos.

A interface e utilização da ferramenta são descritas em detalhes em [Borges 2001].



(I) (II)



(III)

Figura 4.2 - Sistema de Consultas Nebulosas

4.3 - Recuperação dos Dados

Após a construção da expressão de consulta, essa deve ser submetida ao MmOS para recuperação das mídias que a satisfazem. Para avaliar a expressão, o mecanismo desenvolvido realiza duas etapas. A primeira efetua uma pré-seleção das mídias armazenadas com base nos temas envolvidos. Na segunda etapa são realizados cálculos para verificar quais dentre as mídias pré-selecionadas realmente satisfazem os parâmetros estabelecidos pela expressão de consulta.

Como o processo é realizado via Web, a recuperação dos dados Multimídia é executada no Servidor Web, que deve conter o MmOS ou ao menos estar conectado a ele. Quando os dados são enviados pela ferramenta de consulta, via WWW, o Servidor Web abre uma conexão com o BDMm criando uma nova sessão para recuperação dos dados. A partir desse ponto o mecanismo de avaliação da expressão de consulta é ativado, buscando e recuperando através de metadados as mídias que satisfazem a consulta em questão. Os dados obtidos são organizados na forma de uma coleção de objetos, onde cada objeto representa uma mídia recuperada.

Nesse ponto do processo, a consulta construída através do Sistema de Consultas Nebulosas já foi interpretada, e os dados que a satisfazem já foram recuperados e organizados na forma de uma coleção. Porém, os resultados obtidos ainda não foram apresentados ao usuário e não há nenhuma informação de como isso deve ser feito.

A forma como os dados serão apresentados deve ser extremamente flexível, já que a consulta pode ter sido realizada por diferentes tipos de usuários, com objetivos variados, que vão desde consultas para visualização até intercâmbio com outros módulos do projeto. O padrão SMIL, que é usado como modelo para o armazenamento das aplicações, é adequado para autoria e manipulação das aplicações multimídia, porém seus elementos definem a forma de apresentação das mídias na tela, de forma padronizada, não possuindo a flexibilidade desejada para apresentação das mídias recuperadas do BDMm. O uso de XML permite que as mídias sejam apresentadas fora do contexto da cena, se assim for desejado, além de adeqüar a forma de apresentação a parâmetros como tipo de usuário e recursos multimídia disponíveis, com padrões de documento baseados diretamente nas estruturas do BDMm do projeto AMMO, aproveitando as informações semânticas e todas as vantagens existentes no ambiente. Para prover a flexibilidade necessária ao resultado da consulta é gerado automaticamente um documento XML que representa os dados recuperados. O processo de criação do documento XML envolve vários passos, que são discutidos em detalhes na próxima seção.

4.4 - Criação do Documento XML

A criação dos documentos XML é a parte mais importante de um sistema de informação baseado em XML, já que esse documento será a base de disponibilização e apresentação dos dados. A definição dos tipos de documentos a serem criados deve levar em conta vários fatores, como propósito do documento, tipo de dado a ser representado e formas

de apresentação [Pimentel et al. 2000]. Para que a criação automática desses documentos seja satisfatória, vários passos devem ser seguidos até se chegar à instância do documento em si:

1. Definição do modelo de dados a ser representado pelos documentos.
2. Criação da DTD referente ao modelo adotado.
3. Recuperação dos dados que servirão de conteúdo ao documento XML a ser instanciado.
4. Implementação da geração automática dos documentos, baseado nas fases anteriores.

No desenvolvimento apresentado aqui, o modelo de dados a ser utilizado é aquele implementado no projeto AMMO para representação de objetos de mídia com informações semânticas sobre seu conteúdo, e é sobre esse modelo que deve ser construída a DTD que irá definir a estrutura dos documentos XML criados.

4.4.1 - Processo de Desenvolvimento da DTD

Uma Definição de Tipo de Documento (DTD) serve como gramática para o documento XML, especificando a sua estrutura lógica e servindo como um esquema para os dados representados pelo documento. Define também os elementos de um documento e seus atributos, e a relação entre eles. Documentos XML baseados em uma mesma DTD podem diferir significativamente em termos de conteúdo, mas serão reconhecíveis como um tipo de documento por causa da estrutura ou marcação similar que é utilizada.

Conforme discutido no Capítulo 2, documentos XML podem ser *válidos* ou *bem formados*. Documentos XML que estão em conformidade com uma DTD são considerados *válidos*. Documentos *bem formados* não possuem DTD e, portanto, não podem ser validados quanto à sua estrutura. Esse tipo de documento pode ser validado apenas quanto à sua sintaxe, de acordo com as regras do padrão XML. Porém, em aplicações XML baseadas em Bancos de Dados, é importante que o documento XML seja fiel à estrutura do modelo de dados utilizado. Para que isso seja possível, deve-se possuir uma DTD definida de acordo com a estrutura do Banco de Dados a ser representado para que essa DTD sirva de base aos documentos XML gerados e esses possam ser considerados *válidos*. Segundo Marchal [Marchal 2000], os principais benefícios do uso da DTD são:

- O processador XML pode impor a estrutura, conforme definida na DTD.
- A aplicação tem acesso à estrutura do documento.
- A DTD fornece informações para que o processamento do documento seja correto.
- A DTD pode declarar valores padrão ou fixos para atributos.

Portanto, o uso de documentos XML *válidos* é importante para a correta representação do modelo de dados implementado, e mais importante ainda é a definição da DTD, que servirá como base para validação dos documentos. A DTD deve ser definida de forma a representar o modelo de dados conforme a visão desejada, prevendo o melhor aproveitamento possível do documento criado.

Elementos X Atributos

Criar uma DTD significa, basicamente, definir quais são os elementos de um documento XML, como eles se relacionam e quais os atributos desses elementos. Porém, nesse momento surge uma dúvida: na DTD, o que deve ser definido como elemento e o que deve ser definido como atributo? Embora pareça uma decisão simples, principalmente quando a DTD é baseada em um modelo de dados já existente, essa questão é motivo de debates na comunidade XML, e representa um ponto importante a ser definido no processo de criação da DTD [Marchal 2000]. Veja o exemplo do Quadro 4.1, com duas formas de representação de dados sobre pessoas:

<pessoa>
  <nome> Carlos Alberto </nome>
  <sobrenome> Alves </sobrenome>
  <idade>25</idade>
  <data_nasc>09/09/76</data_nasc>
</pessoa>
	<pessoa
	  nome=”Carlos Alberto”
	  sobrenome=”Alves”
	  idade=”25”
	  data_nasc=”09/09/76”
	/>

Quadro 4.1 - Representação de dados em XML

No primeiro exemplo, cada característica de *pessoa* foi definida como um elemento, enquanto o segundo exemplo define os mesmos dados como atributos do elemento <*pessoa*>. Os dois exemplos do Quadro 4.1 representam corretamente uma tabela ou classe de um Banco

de Dados, com seus respectivos campos ou atributos. Os blocos das DTDs referentes a esses exemplos são apresentados no Quadro 4.2.

<!ELEMENT pessoa (nome,sobrenome, idade,data_nasc)> <!ELEMENT nome CDATA #REQUIRED> <!ELEMENT sobrenome CDATA #REQUIRED> <!ELEMENT idade CDATA #REQUIRED> <!ELEMENT data_nasc CDATA #REQUIRED>	<!ELEMENT pessoa EMPTY> <!ATTLIST pessoa nome CDATA #REQUIRED Sobrenome CDATA #REQUIRED idade CDATA #REQUIRED data_nasc CDATA #REQUIRED>

Quadro 4.2 - DTDs para definição de dados XML

É difícil definir qual das duas formas seria melhor, já que ambas cumprem o objetivo de representar os dados corretamente. Em alguns casos, a escolha entre atributos ou elementos é simplesmente uma questão de estética, enquanto em outros casos, depende do tipo de aplicação a qual o documento XML se presta. De qualquer forma, algumas práticas são consenso e, na maioria dos casos, podem ser seguidas na definição do documento XML:

Devem ser usados elementos:

- quando a informação faz parte do elemento pai.
- para validar estruturas complexas.
- quando a informação possui uma estrutura própria.

Devem ser usados atributos:

- quando a informação é inerente, e não parte, do elemento pai.
- para validar tipos de dados simples.
- para propriedades auxiliares, que estão relacionadas com um dado principal.

Há também algumas vantagens e desvantagens em cada caso que podem ser levadas em conta para decisão da abordagem a ser adotada. Atributos podem ter um valor padrão e são definidos através de tipos, o que muitas vezes facilita o mapeamento de propriedades de objeto para atributos. Com relação ao processamento e manipulação dos documentos, atributos têm a vantagem de estabelecerem um relacionamento forte com o elemento pai que os contém, o que facilita o processamento desse elemento pai e de todas suas propriedades. Elementos, contudo, são naturalmente organizados como uma árvore, oferecendo a

possilidade de estruturação (que os atributos não possuem), além de possuírem formatação de ocorrência através de caracteres como * ou +.

Portanto, existem vantagens e desvantagens nas duas formas de representação, e não há normas rígidas que definem o que deve ser usado em cada caso. Cabe ao projetista visualizar todo o contexto da aplicação e escolher a abordagem que torne o documento XML o mais útil possível.

Para representação dos dados multimídia obtidos como resultado às consultas do projeto AMMO, o mapeamento foi feito relacionando-se as *classes* da estrutura de dados com *elementos* da DTD, sendo que as *propriedades* (ou atributos) das classes foram mapeadas como *atributos* dos elementos criados. Essa abordagem foi adotada levando em conta que os relacionamentos entre as classes formam uma estrutura complexa, que fica mais bem representada através da estruturação dos elementos na DTD. As propriedades das classes são representadas na forma de atributos dos elementos no documento XML. Dessa forma, cada classe da estrutura de dados possui seu elemento correspondente no documento XML, e esses elementos, por sua vez, estão relacionados na DTD conforme a estrutura de dados mantida no BDMm.

Visão do Banco de Dados

Definidas todas as características conceituais, o primeiro passo na implementação de uma DTD deve ser a definição do elemento raiz, que vai englobar o documento todo. Porém, muitas vezes o modelo de dados a ser representado não possui um elemento raiz explicitamente definido, ou mesmo que possua, esse pode não representar exatamente a estrutura desejada no documento XML. Geralmente a representação completa de um modelo de dados na forma de documentos XML exige a criação de mais de uma DTD para que seja possível representar todas as visões e relacionamentos presentes no Banco de Dados.

Dessa forma, é importante que o objetivo da criação do documento XML esteja bem definido desde o início do processo, para que o documento seja realmente útil quando for instanciado. Relacionamentos complexos e dados importantes na estrutura do Banco de Dados podem gerar elementos muito complexos ou mesmo inúteis, dificultando a geração e a interpretação do documento XML, comprometendo sua eficácia dentro da aplicação. Por outro lado, muitas vezes elementos do documento XML que não possuem correspondentes no

modelo de dados, ou mesmo conteúdos redundantes para alguns elementos, podem tornar o documento XML mais amigável quanto à manipulação e visualização.

Para representação dos objetos recuperados do MmOS do projeto AMMO, a visão considerada está focada na consulta realizada via Web, onde os objetos de mídia propriamente ditos tornam-se o ponto central de atenção, já que são eles os dados relacionados às informações semânticas que servem de base para a consulta. Como o objetivo do documento XML é representar os dados obtidos como resultado de consultas, a DTD que o define terá como elemento central o objeto de mídia. Portanto, o documento XML é formatado por elementos da estrutura do BDMm com base na classe *MediaObject*, que é a classe que representa os objetos obtidos como resultado, e a partir dessa classe os demais dados são relacionados e representados. Nesse sentido, as cenas e aplicações aparecem no documento XML como parte dos dados referentes ao objeto de mídia recuperado, sendo que para cada objeto de mídia recuperado são representadas as cenas e aplicações às quais esse objeto pertence, mantendo a relação de cada mídia com suas cenas e aplicações, caso seja necessário recuperá-las. Não é necessário, como resultado da consulta, representar todos os objetos de mídia que formam uma cena, como está no modelo de dados, já que o documento XML está focado nas mídias obtidas como resultado à consulta. A definição desse documento, ou seja, sua DTD, é apresentada a seguir.

4.4.2 - A DTD *MMDATA*

A DTD criada está especificada de acordo com as normas do padrão XML, e define uma linguagem de marcação denominada *MMDATA*, conforme o código apresentado no Quadro 4.3.

```
<!-- ====================== Início da DTD MMDATA ==================== -->
<!-- ==================== Declarações de Entidades ================== -->

<!ENTITY % mediatype "(Audio|Image|Text|Video|Animation|TextStream)">

<!-- ======== Declarações de Elementos e Listas de Atributos ======== -->

<!ELEMENT mmdata (mediaobject)*>
<!ELEMENT mediaobject (scene+)>
<!ATTLIST mediaobject
      name        CDATA       #IMPLIED
      type        %mediatype; #REQUIRED
      mediasource CDATA       #REQUIRED
      extension   CDATA       #IMPLIED>
<!ELEMENT scene (application)+>
<!ATTLIST scene
      documentsmil CDATA      #REQUIRED>
<!ELEMENT application EMPTY>
<!ATTLIST application
      firstscene  CDATA       #REQUIRED>
<!-- ========================== Fim da DTD ========================== -->
```

Quadro 4.3 - Código da DTD MMDATA

A DTD criada define, inicialmente, uma entidade chamada de `mediatype`. Essa entidade servirá como base para definição de atributos que referenciam os tipos de mídia que podem estar presentes no BDMm.

Em XML, assim como nas linguagens de marcação em geral, o documento é marcado por um elemento principal (*root element*), que o caracteriza. Dessa forma, documentos HTML sempre são iniciados pela marcação `<HTML>` e encerrados com `</HTML>`, documentos SMIL são marcados por `<SMIL>` e `</SMIL>`, e assim por diante. No Padrão de Documento definido aqui, a marcação principal é `mmdata`, cujo nome indica que o documento representará o resultado da consulta, que é um conjunto de dados multimídia. Esse elemento engloba todo o documento e, portanto, conterá todas as informações sobre as mídias, representando o conjunto de mídias como um todo. O resultado da consulta pode ser um conjunto com várias mídias, mas também pode ser nulo no caso da consulta não encontrar nenhuma mídia que satisfaça os requisitos informados. Sendo assim, o conteúdo do elemento é definido como

(mediaobject)*, com o sinal * indicando que pode haver zero ou mais elementos mediaobject.

O elemento mediaobject, por sua vez, vai representar as informações sobre cada objeto de mídia recuperado como resposta à consulta realizada. A construção desse elemento na DTD indica que ele tem como conteúdo uma ou mais (sinal +) cenas (scene). Essa construção segue a lógica da estrutura do BDMm, em que cada mídia está associada a pelo menos uma cena, sendo que a mesma mídia pode também estar sendo referenciada por outras cenas.

Os atributos de cada elemento, que são definidos através de ATTLIST, representam os atributos de cada classe conforme a definição no BDMm. A maioria dos atributos dos elementos é definida como obrigatório (#REQUIRED) e com conteúdo do tipo texto (CDATA - *Character Data*).

Os atributos do elemento mediaobject, que representam os atributos da classe *MediaObject*, são:

- name - contém o nome do elemento de mídia, conforme definido na autoria.
- type - define o tipo de mídia referente ao objeto sendo representado. Seu valor é definido como mediatype, fazendo referência à entidade criada no início da DTD para representar os tipos de mídia suportados no BDMm.
- mediasource - contém o caminho físico (*path*) do arquivo de mídia.
- extension - Extensão ou tipo do arquivo de mídia. Define qual padrão multimídia o arquivo obedece, como JPEG ou BMP para imagens, e AVI ou MPEG para vídeos. O termo IMPLIED indica que o atributo pode ser omitido, diferente dos demais atributos do elemento, que são obrigatórios (REQUIRED).

Todos os demais elementos da DTD seguem as convenções de sintaxe descritas para o elemento mediaobject, e que são parte do padrão XML. Da mesma forma, todos os elementos são logicamente definidos de acordo com a estrutura implementada no BDMm do projeto AMMO.

O elemento scene é definido de forma a estar associado a um ou mais elementos application, já que uma cena pode ser referenciada por uma ou mais aplicações do BDMm.

Por fim, o elemento application é vazio (EMPTY), sendo caracterizado através dos atributos que o compõem.

4.4.3 - Usando a DTD *MMDATA*

No contexto do trabalho aqui apresentado, as consultas realizadas pelo Sistema de Consultas Nebulosas terão seus resultados transformados automaticamente em documentos XML em conformidade com a DTD *MMDATA*. Como exemplo, vamos considerar a aplicação apresentada anteriormente (Seção 3.4) e a situação onde um usuário interessado em obter material sobre Robótica realizou uma consulta com o objetivo de recuperar dados a respeito de Robôs Humanóides, que caracterizam uma área de estudo da Robótica, e obteve como resultado (por exemplo) o conjunto de mídias apresentado no Quadro 4.4.

MediaSource	***Type***
asimo.txt	Text
asimo.jpg	Image
asimomodel1.jpg	Image
whatisrobot.wav	Audio
asimo.mpg	Video

Quadro 4.4 - Mídias recuperadas (exemplo)

O conteúdo do documento XML é construído a partir dos dados obtidos como resultado da consulta. O algoritmo apresentado no Quadro 4.5 transforma os dados de cada mídia em seu correspondente no formato XML. Note que o algoritmo não inclui a marcação `<mmdata>`, que caracteriza o documento, por tratar apenas um objeto de mídia de cada vez. A construção do documento XML completo será discutida a seguir.

```
Inserir tag-início do elemento de mídia (<mediaobject>)
Recuperar propriedades do objeto de mídia
Inserir propriedades da mídia como atributos (name, type, mediasource e extension)
Recuperar conjunto de cenas que a mídia referencia
Para cada cena recuperada faça
    Inserir tag-início do elemento de cena (<scene>)
      Inserir propriedade da cena como atributo (documentsmil)
    Recuperar aplicações a que a cena pertence
    Para cada aplicação recuperada faça
      Inserir tag do elemento vazio de aplicação (<application/>)
      Inserir propriedade da aplicação como atributo (firstscene)
    Fim-Para
    Inserir tag-fim do elemento de cena (</scene>)
Fim-Para
Inserir tag-fim do elemento de mídia (</mediaobject>)
```

Quadro 4.5 - Algoritmo para geração de dados XML (DTD MMDATA)

Nesse ponto estão formatadas em XML as referências a todos os objetos obtidos como resultado da consulta, assim como a informação sobre cada elemento de mídia, e a qual cena e aplicação pertencem.

Para visualização do resultado, é criado o documento XML apresentado no Quadro 4.6. O cabeçalho, a indicação da DTD *MMDATA* como padrão para o documento e a *tag* raiz `<mmdata>` são elementos fixos para todos documentos criados, que têm conteúdo gerado automaticamente conforme o algoritmo apresentado no Quadro 4.5, representando o resultado da consulta. Dessa forma, o documento XML gerado mantém um padrão naturalmente direcionado à Internet e facilmente intercambiável, facilitando a manipulação dos dados multimídia obtidos como resposta à consulta.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<!DOCTYPE mmdata SYSTEM "mmdata.dtd">
<mmdata>
      <mediaobject name="asimo.txt" type="Text"
                  mediasource="./robotics/text/asimo.txt">
                  <scene documentsmil="./robotics/appasimo.smi">
                    <application firstscene="./robotics/presentation.smi"/>
                  </scene>
      </mediaobject>
      <mediaobject name="asimo.jpg" type="Image"
                 mediasource="./robotics/images/asimo.jpg" extension="jpg">
                  <scene documentsmil="./robotics/appasimo.smi">
                    <application firstscene="./robotics/presentation.smi"/>
                  </scene>
                  <scene documentsmil="./robotics/presentation.smi">
                    <application firstscene="./robotics/presentation.smi"/>
                  </scene>
      </mediaobject>
      <mediaobject name="asimomodel1.jpg" type="Image"
           mediasource="./robotics/images/asimomodel1.jpg" extension="jpg">
                  <scene documentsmil="./robotics/appasimo.smi">
                    <application firstscene="./robotics/presentation.smi"/>
                  </scene>
      </mediaobject>
      <mediaobject name="whatisrobot.wav" type="Audio"
                  mediasource="./robotics/audio/whatisrobot.wav"
                  extension="wav">
                  <scene documentsmil="./robotics/concepts.smi">
                    <application firstscene="./robotics/presentation.smi"/>
                  </scene>
      </mediaobject>
      <mediaobject name="asimo.mpg" type="Video"
                  mediasource="./robotics/video/asimo.mpg" extension="mpg">
                  <scene documentsmil="./robotics/appasimo.smi">
                    <application firstscene="./robotics/presentation.smi"/>
                  </scene>
      </mediaobject>
</mmdata>
```

Quadro 4.6 - Documento XML representando os dados multimídia recuperados

4.5 - Acrescentando Dados Semânticos ao documento XML

Uma das principais características do BDMm do projeto AMMO é o fato de serem mantidos dados semânticos de cada mídia armazenada. Esses dados permitem a busca e recuperação das mídias a partir de informações sobre seu conteúdo, proporcionando alta reutilização e melhor aproveitamento do conteúdo do BDMm.

O Quadro 4.7 mostra a definição dos dados semânticos referentes às mídias recuperadas conforme exemplo apresentado no Quadro 4.4.

Mídia	Dados Semânticos
Name: asimo.txt *Type*: Text	*Subjects*: Robô, Especificação *Qualifiers*: Humanóide, Autônomo (*Subject Robô)* *Associations:* *subjStart:* Especificação *directAssoc:* de *subjEnd:* Robô Humanóide Autônomo
Name: asimo.jpg *Type*: Image	*Subjects*: Robô *Qualifiers*: Humanóide, Branco (*Subject Robô)*
Name: asimomodel1.jpg *Type*: Image	*Subjects*: Robô *Qualifiers*: Humanóide, Branco (*Subject Robô)*
Name: whatisrobot.wav *Type*: Audio	*Subjects*: Cientista, Robôs, Humanos *Associations:* *subjStart:* Robôs / Cientista *directAssoc:* e / define *subjEnd:* Humanos / Robôs
Name: asimo.mpg *Type*: Video	*Subjects*: Robô, Pessoas *Qualifiers*: Branco, Humanóide (*Subject Robô)* *Associations:* *subjStart:* Robô Branco Humanóide / Pessoas *directAssoc:* a frente de / *reverseAssoc:* com *subjEnd:* Pessoas / Robô Branco Humanóide

Quadro 4.7 - Dados semânticos mantidos no BDMm do Projeto AMMO

Os dados semânticos armazenados podem ser aproveitados em outras atividades além da recuperação de dados. A acessibilidade, por exemplo, é um fator importante a ser considerado na apresentação de dados multimídia e que pode ser beneficiado pela utilização dos dados semânticos como incremento para visualização dos dados multimídia recuperados.

Através dos dados semânticos das mídias, pode-se representar as mídias recuperadas pela sua descrição, proporcionando uma alternativa para aqueles usuários que porventura sofram de algum tipo de restrição para execução ou visualização de determinado tipo ou formato de mídia, como a falta de um *player* apropriado ou mesmo de equipamentos multimídia. Mesmo para aqueles usuários que possuam todos os recursos disponíveis, a apresentação da descrição do conteúdo da mídia pode ser uma forma mais simples de refinamento da busca ou mesmo de otimização de tempo para análise de um resultado muito extenso, permitindo analisar os dados obtidos sem a necessidade de execução ou visualização completa de cada mídia recuperada.

Devido à quantidade de informações contidas em um arquivo de dados multimídia, seu tamanho geralmente é muito grande. Portanto, a velocidade de recuperação e apresentação das mídias é um fator crucial na manipulação de aplicações multimídia. Esse fator pode ser otimizado através da utilização das informações semânticas para descrever seu conteúdo, de forma textual, permitindo ao usuário que escolha as mídias que julgar realmente interessantes, para que só então sejam carregadas e apresentadas.

Portanto, as mídias selecionadas e representadas através de um documento XML devem trazer consigo suas informações semânticas, prevendo a utilização desses dados na apresentação do resultado da consulta. Para tanto, o documento XML gerado automaticamente após a recuperação dos dados deve incluir também em seu conteúdo os dados semânticos de cada mídia recuperada.

As próximas seções descrevem como os dados semânticos são representados e a forma como esses dados podem ser inseridos no documento XML descrito anteriormente.

4.5.1 - A DTD *SEMANTICDATA*

A DTD criada para representação das informações semânticas está baseada na estrutura de dados exposta no Capítulo 3, mais especificamente na estrutura do BDMm que modela as informações semânticas mantidas no banco (Figura 3.4 parte (a)). Essa estrutura tem como elemento principal a classe *SemanticInformation*, que serve de raiz para as demais classes da estrutura de dados semânticos. Das subclasses da estrutura, a classe *Similarity* não é representada por se tratar de uma classe criada com o objetivo de manter informações a respeito da similaridade entre elementos armazenados no BDMm, sendo importante para recuperação correta dos dados, mas não apresentando importância fundamental para visualização ou representação dos dados recuperados. A classe *Domain* tem como função agrupar tipos de elementos, e também não é representada no documento XML. A classe *Element* não é representada diretamente, mas os valores recuperados dessa classe são representados individualmente na DTD (como conteúdo dos temas (*subjects*)). Portanto, a DTD de dados semânticos não representa a estrutura toda, mas sim apenas os dados que interessam ao escopo da aplicação desenvolvida. Nesse sentido, foram representadas na DTD as informações de temas da mídia, seus qualificadores e associações. O código da DTD criada é apresentado no Quadro 4.8.

```
<!-- ==================== Início da DTD SEMANTICDATA ================== -->
<!-- ========= Declarações de Elementos e Listas de Atributos ========= -->

<!ELEMENT semanticdata (semantic)*>
<!ELEMENT semantic (subject*,association*)>
<!ELEMENT subject (identifier,qualifier*)>
<!ELEMENT identifier (#PCDATA)>
<!ELEMENT qualifier (#PCDATA)>
<!ELEMENT association (subjstart,subjend)>
<!ATTLIST association
          direct        CDATA       #REQUIRED
          reverse       CDATA       #IMPLIED>
<!ELEMENT subjstart (subject)>
<!ELEMENT subjend (subject)>
```

Quadro 4.8 - Código da DTD SEMANTICDATA

A DTD *SEMANTICDATA* define um elemento raiz `semanticdata`, que servirá como base para documentos XML de dados semânticos. Como os dados semânticos são utilizados como complemento às informações dos objetos de mídia recuperados, o elemento `semanticdata` dá lugar ao elemento `mmdata` (raiz do documento gerado para conter o resultado da consulta) nos documentos XML gerados, tendo, portanto, importância apenas conceitual na definição da DTD.

Para realizar o relacionamento dos dados semânticos com os objetos de mídia é definido o elemento `semantic`, que agrupa as informações semânticas podendo ser associado a cada mídia recuperada para construção de um documento XML completo. O elemento `semantic` possui como conteúdo dois elementos: `subject` e `association`, representando, respectivamente, os temas e associações referentes à mídia em questão. O símbolo `*` mantém a relação indicada na estrutura do BDMm, representando o fato de que cada objeto *SemanticInformation* está associado a nenhum ou vários objetos *Subject* e *Association*.

O elemento `subject` representa cada tema associado a mídia, e possui como conteúdo um identificador (`identifier`) e nenhum ou vários qualificadores (`qualifier`), que formam os temas relacionados a mídia. Os elementos `identifier` e `qualifier` possuem como conteúdo dados do tipo texto (PCDATA - *Parsed Character Data*), que são os valores recuperados da classe *Element* no BDMm e que formam o tema.

O elemento `association`, por sua vez, possui como conteúdo os elementos `subjstart` (tema inicial) e `subjend` (tema final), para representar as associações de temas que descrevem a mídia. Os atributos `direct` e `reverse` contêm os valores recuperados do BDMm para formação das associações direta e reversa entre os temas representados pelos elementos `subjstart` e `subjend`.

O conjunto de elementos representados pela DTD *SEMANTICDATA* representa uma visão do banco de dados do projeto AMMO, com o objetivo de fornecer dados alternativos para visualização do resultado da consulta. Para cada mídia recuperada, será associado um elemento `semantic` e, consequentemente, os elementos que o compõem. A geração automática da representação XML dos dados semânticos de cada mídia é feita através do algoritmo do Quadro 4.9, que está dividido em 2 módulos. O algoritmo representa a geração dos dados semânticos relacionados a cada mídia recuperada do BDMm.

```
Recuperar o objeto de informação semântica (SemanticInformation)
Inserir tag-início do elemento de dados semânticos (<semantic>)
Recuperar conjunto de temas da informação semântica
Para cada tema recuperado faça
    Executar procedimento de recuperação de dados do tema (Quadro 4.9 - Módulo 2)
Fim-Para
Recuperar conjunto de associações da informação semântica
Para cada associação recuperada faça
    Inserir tag-início do elemento de associação (<association>)
    Recuperar elemento (Classe Element) de associação direta
    Inserir valor do elemento recuperado como atributo (direct)
    Se há associação reversa
      Recuperar elemento (Classe Element) de associação reversa
      Inserir valor do elemento recuperado como atributo (reverse)
    Fim-se
    Recuperar tema inicial da associação
      Inserir tag-início do elemento de tema inicial (<subjstart>)
      Executar procedimento de recuperação de dados do tema (Quadro 4.9 - Módulo 2)
      Inserir tag-fim do elemento de tema inicial (</subjstart>)
    Recuperar tema final da associação
      Inserir tag-início do elemento de tema final (<subjend>)
      Executar procedimento de recuperação de dados do tema (Quadro 4.9 - Módulo 2)
      Inserir tag-fim do elemento de tema final (</subjend>)
    Inserir tag-fim do elemento de associação (</association>)
Fim-Para
Inserir tag-fim do elemento de dados semânticos (</semantic>)
```

Quadro 4.9 - Módulo 1: Algoritmo principal para geração de dados XML (DTD SEMANTICDATA)

```
Inserir tag-início do elemento de tema (<subject>)
Recuperar elemento (Classe Element) identificador do tema
Inserir tag-início do identificador do tema (<identifier>)
Inserir valor do elemento recuperado como conteúdo
Inserir tag-fim do identificador do tema (</identifier>)
Recuperar conjunto de elementos (Classe Element) qualificadores do tema
Para cada qualificador recuperado faça
    Inserir tag-início do qualificador do tema (<qualifier>)
    Inserir valor do elemento recuperado como conteúdo
    Inserir tag-fim do qualificador do tema (</qualifier>)
Fim-Para
Inserir tag-fim do elemento de tema (</subject>)
```

Quadro 4.9 - Módulo 2: Procedimento auxiliar na geração de dados XML (DTD SEMANTICDATA)

Um exemplo de documento XML contendo dados semânticos é apresentado no Quadro 4.10. O cabeçalho, a indicação da DTD *SEMANTICDATA* como padrão para o documento e a *tag* raiz `<semanticdata>` são elementos fixos para todos documentos desse tipo.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<!DOCTYPE semanticdata SYSTEM "semanticdata.dtd">
<semanticdata>
      <semantic>
            <subject>
                  <identifier>Robô</identifier>
                  <qualifier> Branco </qualifier>
                  <qualifier> Humanóide </qualifier>
            </subject>
            <subject>
                  <identifier>Pessoas</identifier>
            </subject>
            <association direct="a frente de"  reverse="com">
                  <subjstart>
                    <subject>
                        <identifier>Robô</identifier>
                        <qualifier> Branco </qualifier>
                        <qualifier> Humanóide </qualifier>
                    </subject>
                  </subjstart>
                  <subjend>
                     <subject>
                              <identifier> Pessoas </identifier>
                     </subject>
                  </subjend>
            </association>
      </semantic>
</semanticdata>
```

Quadro 4.10 - Documento XML representando dados semânticos

Conforme já citado, dificilmente serão criados documentos contendo apenas dados semânticos, já que esses dados descrevem objetos de mídia que também devem estar presentes no documento XML. Portanto, o algoritmo apresentado no Quadro 4.9 foi desenvolvido prevendo sua utilização como parte do processo de geração do documento XML. Com o mesmo objetivo, a DTD *SEMANTICDATA* foi criada de forma a ser associada com os dados multimídia, tornando-se um módulo com o objetivo específico de descrição de dados semânticos, conforme a estrutura proposta no projeto AMMO. Por ser um módulo independente, os dados semânticos tornam-se opcionais na representação dos dados

multimídia recuperados, já que sua DTD pode ser incluída ou não no documento XML dependendo do interesse da aplicação. Além disso, definida dessa forma a DTD *SEMANTICDATA* pode ser utilizada como padrão de representação de dados semânticos para qualquer documento XML, de qualquer padrão, e não apenas para dados multimídia. A utilização dessa DTD associada a outros padrões de documento (como *MMDATA*) pode ser feita através do conceito de *namespaces*, exposto a seguir.

4.5.2 - *Namespaces*

Inicialmente, documentos XML eram desenvolvidos isoladamente. Uma DTD era desenvolvida para uma determinada aplicação, e os elementos definidos nessa DTD seriam específicos a essa aplicação. Há poucos recursos de reutilização de elementos, exceto, talvez, recortando e colando DTDs antigas nas novas.

Graças aos *namespaces*, é possível desenvolver elementos reutilizáveis, ou seja, elementos que podem ser reaproveitados em vários documentos, através da associação de um padrão específico aos elementos. Isso proporciona a extensão de documentos XML, pois significa que DTDs definidas como *namespaces* podem aumentar os elementos existentes e rotular com clareza quem é o responsável pela extensão. Isso evita conflitos de nomes e é uma maneira de permitir a reutilização de estruturas padrão.

O exemplo do Quadro 4.11 utiliza *namespaces* para reaproveitar uma definição padrão de elementos gráficos em um documento XML.

```
<?xml version=”1.0”?>
<!DOCTYPE graphics SYSTEM "datagraphics.dtd">
<graphics xmlns:graph=”http://www.url.com/graphics/1.0”>
       <picture file=”rectangle.pix”>
            <graph:rectangle>
                  <graph:coords><graph:x> 10 </graph:x>
                  <graph:y> 50 </graph:y> </graph:coords>
                  <graph:height> 10 </graph:height>
                  <graph:width> 30 </graph:width>
            </graph:rectangle>
       </picture>
</graphics>
```

Quadro 4.11 - Exemplo de utilização de *namespaces*

O elemento `graphics` é o elemento raiz do documento XML, que ainda possui o elemento `picture`. O prefixo `graph:` precedendo os outros elementos indica que esses elementos são definidos através da utilização de um *namespace.* No exemplo, o prefixo que associa os elementos à DTD reaproveitada foi `graph`. A definição do prefixo (e do *namespace*) é feita pela linha:

```
<graphics xmlns:graph="http://www.url.com/graphics/1.0">
```

A declaração associa um URI (*Uniform Resource Identifier*) a um prefixo. Normalmente, o tipo de URI associada é um URL (*Uniform Resource Locator*), que é o padrão de endereços utilizado na Internet. Como URLs são garantidamente exclusivos, não há como haver duplicação de identificação nos prefixos. Com isso, mesmo DTDs que possuam elementos com nomes comuns podem ser utilizadas como *namespaces*, pois identificarão unicamente cada elemento através do prefixo.

Portanto, *namespaces* XML são um mecanismo para identificar de modo não ambíguo DTDs diferentes daquela usada como base para o documento XML. O uso desse recurso permite forte reutilização de dados, propiciando a criação de padrões de representação e diminuindo a ambigüidade.

No contexto deste trabalho, a DTD *SEMANTICDATA* será associada a documentos XML em conformidade com a DTD *MMDATA* através de *namespaces.* Porém, nada impede a utilização do modelo de dados semânticos definido aqui em larga escala, com outros tipos de documentos XML. A forma como os documentos XML são criados, contendo tanto os dados das mídias recuperadas quanto suas informações semânticas, é o assunto da próxima seção.

4.6 - Criando um documento XML completo

Com base nas DTDs desenvolvidas e nos algoritmos definidos, torna-se possível a criação automática de documentos XML que representem os dados multimídia (com informações semânticas) obtidos como resposta às consultas realizadas ao BDMm do projeto AMMO, conforme a arquitetura proposta no início deste capítulo (Figura 4.1).

O documento final criado contém todas as informações recuperadas, ou seja, os dados referentes às mídias com suas respectivas informações semânticas. Para tanto, as duas DTDs desenvolvidas são utilizadas, assim como todos os algoritmos definidos, o que torna

necessária a definição de um novo algoritmo que englobe todas as informações e procedimentos necessários.

O Quadro 4.12 apresenta o algoritmo final para criação do documento XML.

```
Inserir cabeçalho XML (<?xml ..... ?>)
Declarar DTD principal (<!DOCTYPE ..... >)
Inserir tag-início do elemento raiz (<mmdata>)
Se dados semânticos serão inseridos então
    Definir namespace (xmlns:sem="semantic.dtd")
Fim-Se
Para cada mídia recuperada faça
    Inserir tag-início do elemento de mídia (<mediaobject>)
    Recuperar propriedades do objeto de mídia
    Inserir propriedades da mídia como atributos (type, name, mediasource e extension)
    Se dados semânticos serão inseridos então
        Executar procedimento de recuperação dos dados semânticos (Quadro 4.9)
    Fim-Se
    Recuperar conjunto de cenas que a mídia referencia
    Para cada cena recuperada faça
        Inserir tag-início do elemento de cena (<scene>)
        Inserir propriedade da cena como atributo (documentsmil)
        Recuperar aplicações a que a cena pertence
        Para cada aplicação recuperada faça
            Inserir tag do elemento vazio de aplicação (<application/>)
            Inserir propriedade da aplicação como atributo (firstscene)
        Fim-Para
        Inserir tag-fim do elemento de cena (</scene>)
    Fim-Para
    Inserir tag-fim do elemento de mídia (</mediaobject>)
Fim-Para
Inserir tag-fim do elemento raiz (</mmdata>)
```

Quadro 4.12 - Algoritmo final para criação do documento XML

O Quadro 4.13 apresenta um documento XML completo criado conforme o algoritmo apresentado. O documento criado está em conformidade com a DTD *MMDATA*, e usa o recurso de *namespaces* para incluir dados semânticos das mídias. O prefixo "`sem`" referencia elementos em conformidade com a DTD *SEMANTICDATA*, conforme definido no elemento raiz do documento. O documento apresentado utiliza o mesmo conjunto de mídias e dados semânticos utilizado como exemplo anteriormente (Quadro 4.7), porém, para simplificar a explicação, não contém todas as mídias apresentadas no exemplo.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<!DOCTYPE mmdata SYSTEM "mmdata.dtd">
<mmdata xmlns:sem="semantic.dtd">
<mediaobject name="whatisrobot.wav" type="Audio"
                  mediasource="./robotics/audio/whatisrobot.wav"
                  extension="wav">
      <sem:semantic>
            <sem:subject>
                  <sem:identifier> Cientista </sem:identifier>
            </sem:subject>
            <sem:subject>
                  <sem:identifier> Robôs </sem:identifier>
            </sem:subject>
            <sem:subject>
                  <sem:identifier> Humanos </sem:identifier>
            </sem:subject>
            <sem:association direct="e">
                  <sem:subjstart> <sem:subject>
                        <sem:identifier> Robôs </sem:identifier>
                  </sem:subject> </sem:subjstart>
                  <sem:subjend> <sem:subject>
                        <sem:identifier> Humanos </sem:identifier>
                  </sem:subject> </sem:subjend>
            </sem:association>
            <sem:association direct="define">
                  <sem:subjstart> <sem:subject>
                        <sem:identifier> Cientista </sem:identifier>
                  </sem:subject> </sem:subjstart>
                  <sem:subjend> <sem:subject>
                        <sem:identifier> Robôs </sem:identifier>
                  </sem:subject> </sem:subjend>
            </sem:association>
      </sem:semantic>
      <scene documentsmil="./robotics/concepts.smi">
            <application firstscene="./robotics/presentation.smi"/>
      </scene>
</mediaobject>
<mediaobject name="asimo.mpg" type="Video"
            mediasource="./robotics/video/asimo.mpg"
            extension="mpg">
      <sem:semantic>
```

```
            <sem:subject>
                  <sem:identifier>Robô</sem:identifier>
                  <sem:qualifier> Branco </sem:qualifier>
                  <sem:qualifier> Humanóide </sem:qualifier>
            </sem:subject>
            <sem:subject>
                  <sem:identifier>Pessoas</sem:identifier>
            </sem:subject>
            <sem:association direct="a frente de" reverse="com">
                  <sem:subjstart> <sem:subject>
                        <sem:identifier> Robô </sem:identifier>
                        <sem:qualifier> Branco </sem:qualifier>
                        <sem:qualifier> Humanóide </sem:qualifier>
                  </sem:subject> </sem:subjstart>
                  <sem:subjend> <sem:subject>
                              <sem:identifier> Pessoas </sem:identifier>
                  </sem:subject> </sem:subjend>
            </sem:association>
      </sem:semantic>
      <scene documentsmil="./robotics/appasimo.smi">
            <application firstscene="./robotics/presentation.smi"/>
      </scene>
</mediaobject>
<!-- Outras mídias recuperadas -->
</mmdata>
```

Quadro 4.13 - Exemplo de documento XML com dados semânticos

4.7 - RDF na manipulação de metadados na Web

No contexto deste trabalho, foram usados documentos XML puros baseados nas DTDs apresentadas (*SEMANTICDATA* e *MMDATA*), mas nada impede a utilização dessas DTDs como modelo de metadados para documentos RDF. O padrão RDF (*Resource Description Framework*) [RDFW3C 1999] foi criado pelo W3C para representação de informações que descrevem dados disponibilizados na Web, ou seja, tem como objetivo principal representar metadados sobre recursos Web. Tem como principal motivação prover facilidade para processamento automático desses metadados, para definir um mecanismo comum de descrição de informação sobre qualquer domínio. Como fornece um meio de representação de metadados, permite troca entre aplicações sem perda de significado e, por ser desenvolvido

como padrão para uso comum, incentiva a disponibilidade de *parsers* e ferramentas de processamento compatíveis entre si. No Apêndice B são descritos documentos RDF e sua sintaxe.

Capítulo 5

Visualizando dados multimídia com XML

5.1 - Introdução

Conforme discutido no Capítulo 2, a representação de dados em XML provê alta flexibilidade para sua visualização na Web, além de XML ser um padrão altamente intercambiável. A estrutura do documento XML, que traz consigo informações a respeito de seu conteúdo e não de sua visualização, prevê a utilização de recursos como Folhas de Estilo XSL, interação com HTML, CSS ou DSSSL para formatação de apresentação, o que permite que o usuário visualize um mesmo documento com formatos diferentes e possa selecionar o conteúdo a ser apresentado. Dessa forma, o documento final a ser carregado pelo navegador Web pode possuir representações diferentes com relação à formatação e também aos dados apresentados.

Portanto, a utilização de documentos XML como parte do processo de consulta ao BDMm do projeto AMMO ganha importância no momento de visualização do resultado das consultas. As próximas seções detalham as vantagens proporcionadas quanto à apresentação dos objetos de mídia e manipulação das informações semânticas, apresentando resultados obtidos através da construção de Folhas de Estilo XSL que visam proporcionar flexibilidade ao ambiente.

5.2 - Utilização do documento XML

Associada ao uso de dados semânticos, a flexibilidade de XML traz vantagens importantes para visualização dos dados multimídia. Como a apresentação satisfatória de dados multimídia depende de fatores como tipo de usuário, de recursos tecnológicos disponíveis (de *software* e *hardware*) ou mesmo de opções de navegação do usuário, a acessibilidade se torna um fator importante na publicação dos dados [Brusilovsky 1996, Rousseau et al. 1999].

Devido à diversidade de perfis de usuários possíveis para sistemas disponíveis via Web, são comuns problemas como falta de elementos de *software* (*plug-ins*, *players*,

bibliotecas, etc.) ou de *hardware* para visualização de elementos como vídeo, áudio e até mesmo imagens. Uma alternativa para contornar essas restrições é a representação das mídias pela sua descrição, através da manipulação dos dados semânticos contidos no documento XML. Além de contornar problemas tecnológicos, a descrição do conteúdo da mídia melhora o tempo de carga de resultados preliminares muito extensos sem a necessidade de execução ou visualização de cada mídia recuperada, permitindo que o usuário refine a busca baseando-se em dados semânticos textuais.

A melhora também ocorre em consultas com resultados muito extensos. Uma das características de arquivos de objetos multimídia diz respeito à necessidade de grandes áreas de armazenamento, pois são arquivos que contêm muitas informações e, conseqüentemente, ocupam muito espaço. Essa característica torna a velocidade de preparação e execução dos arquivos multimídia um fator muito importante no momento da apresentação. O uso de XML com dados semânticos permite opções de visualização que agilizam esse processo, o que pode ser muito vantajoso. Com a apresentação textual dos dados semânticos, o usuário pode analisar o conteúdo de cada mídia recuperada e escolher a apresentação completa de cada mídia conforme seu interesse, evitando a carga simultânea e controlando qual mídia deve ser apresentada em determinado momento.

O resultado também pode ser formatado para se adequar às necessidades do usuário quanto ao uso do sistema. A partir do documento XML podem ser gerados dados estatísticos constando nomes de arquivos, quantidade de mídias recuperadas, catálogos textuais para impressão ou outros formatos de apresentação dos resultados que podem ser úteis ao usuário, sem necessidade de execução dos arquivos de mídia.

Portanto, a flexibilidade inerente aos documentos XML faz com que o documento gerado automaticamente para representar as mídias recuperadas do BDMm seja uma base para geração de visualizações com formatos variados. Porém, o documento XML também pode ser mantido em seu estado original, sem formatação de apresentação, prevendo sua utilização apenas como padrão de documento para intercâmbio entre aplicações, ou mesmo como fonte de dados para outras aplicações. No âmbito do projeto AMMO, pode servir como base para integração das várias ferramentas e módulos componentes do ambiente, proporcionando a definição e implementação desses módulos independente de plataforma ou ambiente de programação, utilizando a DTD do documento XML como base para manipulação e troca das informações.

Ainda no contexto do AMMO, a facilidade de aplicação de várias formas de visualização também vem contribuir para o poder de adaptação do ambiente às constantes

variações de ferramentas e plataformas, notadamente na Web. Além disso, permite também que haja alterações na forma de disponibilização e apresentação dos dados conforme a evolução do projeto, utilizando o mesmo documento XML como base.

A próxima seção apresenta exemplos práticos de visualização dos documentos XML ilustrando algumas das vantagens citadas para apresentação dos dados multimídia tirando proveito de suas informações semânticas. Os exemplos apresentados foram desenvolvidos para uso no ambiente MAW do projeto AMMO como parte do processo de publicação na Web do resultado de consultas baseadas em conteúdo, e utiliza Folhas de Estilo XSL para efetuar a filtragem e formatação dos dados XML.

5.3 - Formas de visualização

O documento XML pode ser transformado através da utilização de Folhas de Estilo XSL, cujo código mapeia e filtra os dados XML para um novo formato. A transformação do documento pode ser realizada no cliente, através de referência no documento XML ou na ferramenta de apresentação; ou ainda no servidor, através do uso de um processador XSL que cria fisicamente um novo arquivo com o documento transformado.

Para os documentos XML gerados automaticamente com resultados de consulta ao BDMm, a forma direta de visualização é transformá-los em um documento HTML que apresente todas as mídias recuperadas em um navegador Web, possibilitando a ativação da cena e da aplicação correspondente. A Figura 5.1 demonstra esse formato de visualização, utilizando o mesmo conjunto de mídias e dados semânticos utilizado como exemplo anteriormente (Quadro 4.7).

XSL é uma dentre as várias formas possíveis de formatação de apresentação de documentos XML. O código das Folhas de Estilo XSL utilizadas encontra-se no Apêndice A. Detalhes sobre o código XSL podem ser encontrados em [XSLW3C 2001].

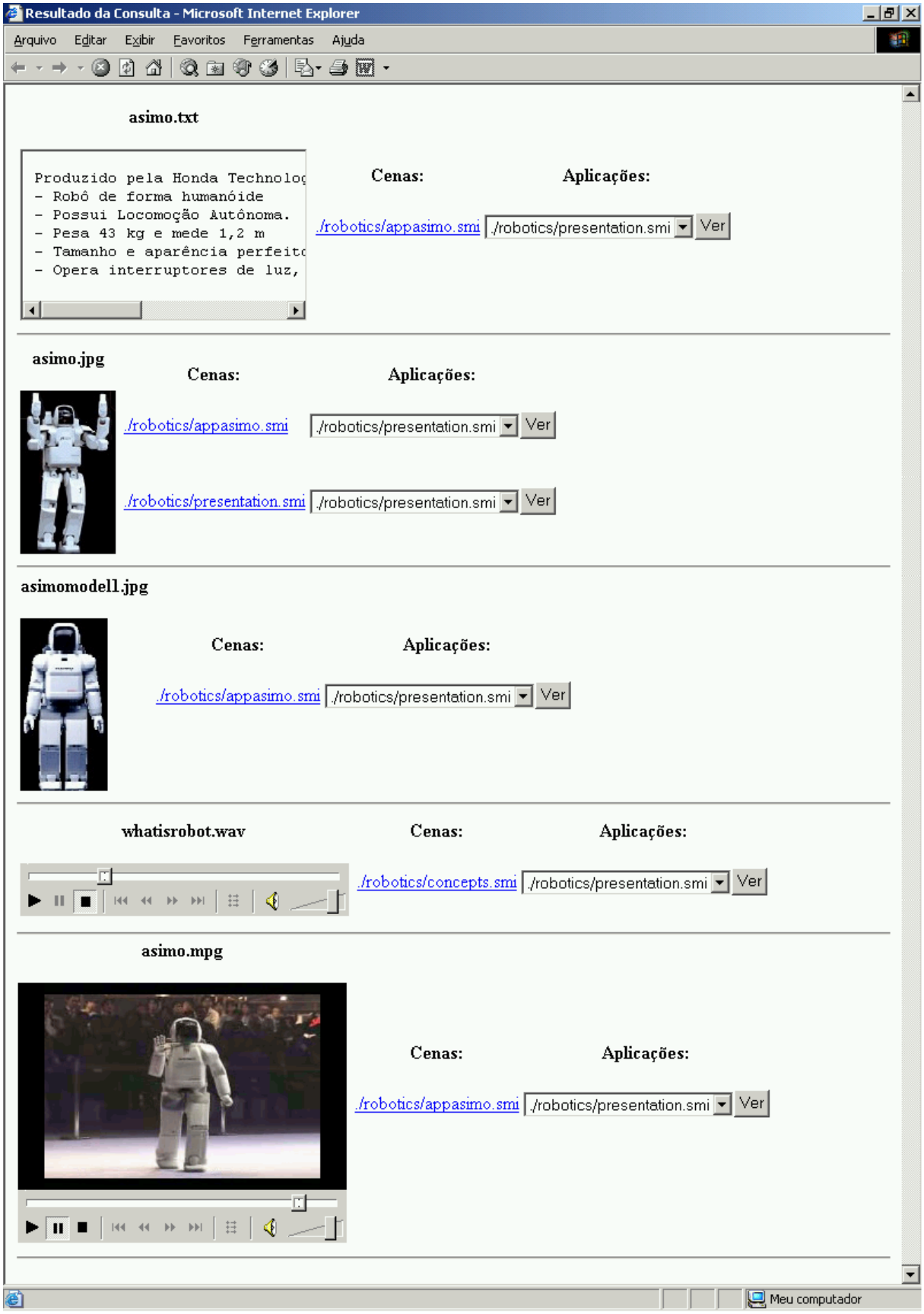


Figura 5.1 - Visualização do documento XML

No exemplo da Figura 5.1, cada mídia é completamente carregada, permitindo sua visualização. Os dados semânticos são ignorados nesse momento, sendo que as informações adicionais presentes são o nome do arquivo (acima da mídia recuperada) e as cenas e aplicações correspondentes, que também podem ser visualizadas. No caso da visualização da cena, outro documento XML é gerado automaticamente contendo os dados da cena em questão para que seja apresentada uma tela com os dados semânticos de todas as mídias constantes da cena. Essa tela servirá de apoio no caso do usuário não possuir todos os recursos necessários para execução das mídias presentes na cena, como no caso em que um usuário sem equipamento de áudio executa uma consulta para recuperação de imagens e, a partir de uma das imagens recuperadas, executa uma cena da aplicação. Se essa cena possuir algum elemento de áudio, o usuário terá acesso à descrição desse elemento, mesmo que não possa ouvi-lo.

A Figura 5.2 apresenta um exemplo de cena sendo visualizada a partir de uma consulta realizada. Note que o navegador Web apresenta os dados semânticos das mídias enquanto a cena é executada.

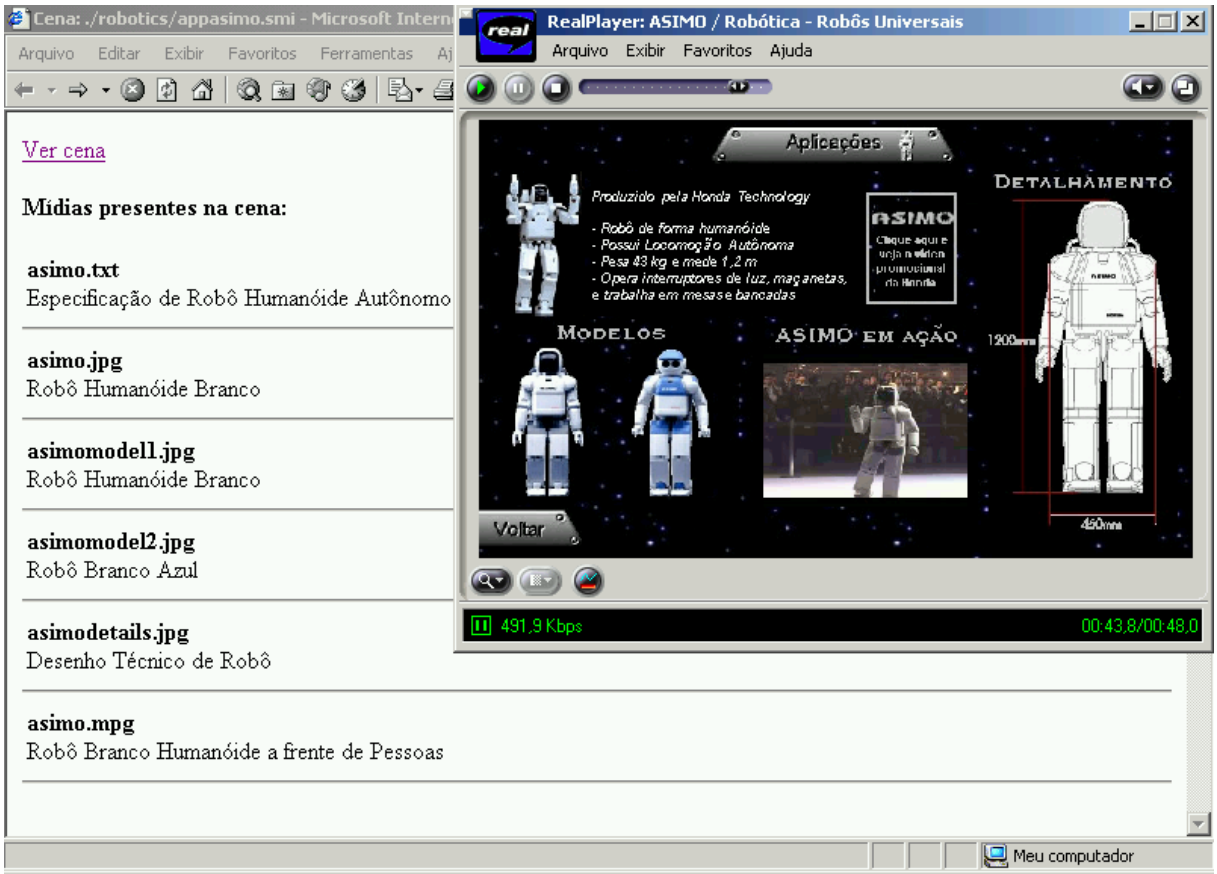


Figura 5.2 - Visualização da cena a partir da consulta

Os dados semânticos podem ser utilizados também para evitar o carregamento desnecessário de vários elementos de mídia simultaneamente, ficando a cargo do usuário selecionar, através da descrição, a mídia que deseja visualizar. Essa forma de visualização é apresentada na Figura 5.3.

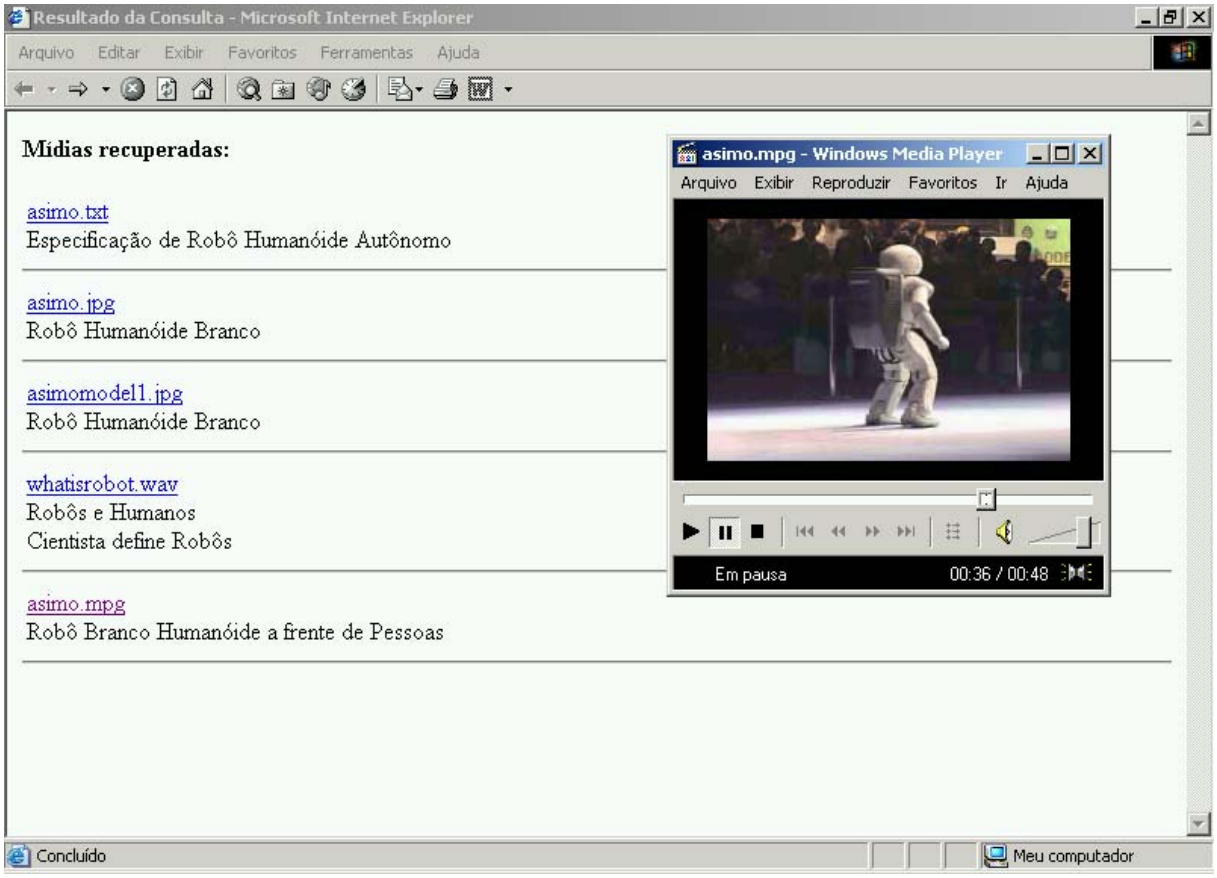


Figura 5.3 - Visualização das mídias controlada pelo usuário

A Figura 5.3 demonstra uma forma de visualização diferenciada para as mídias apresentadas na Figura 5.1, exibindo apenas sua descrição e permitindo que o usuário visualize os elementos um a um, conforme demonstrado com o elemento de vídeo *asimo.mpg* na Figura 5.3. Como exemplo, esse formato não apresenta *links* para as cenas e aplicações, mas isso não impede que esses dados sejam acrescentados ao formato de visualização.

Outras formas de visualização podem ser disponibilizadas, com objetivos diferentes dos apresentados até aqui, como por exemplo o formato da Figura 5.4, que apresenta um catálogo para impressão dos resultados encontrados. Vale lembrar que as Folhas de Estilo XSL também podem ser usadas para transformação de documentos XML em outros formatos, como PDF (*Portable Document Format*) ou RTF (*Rich Text Format*).

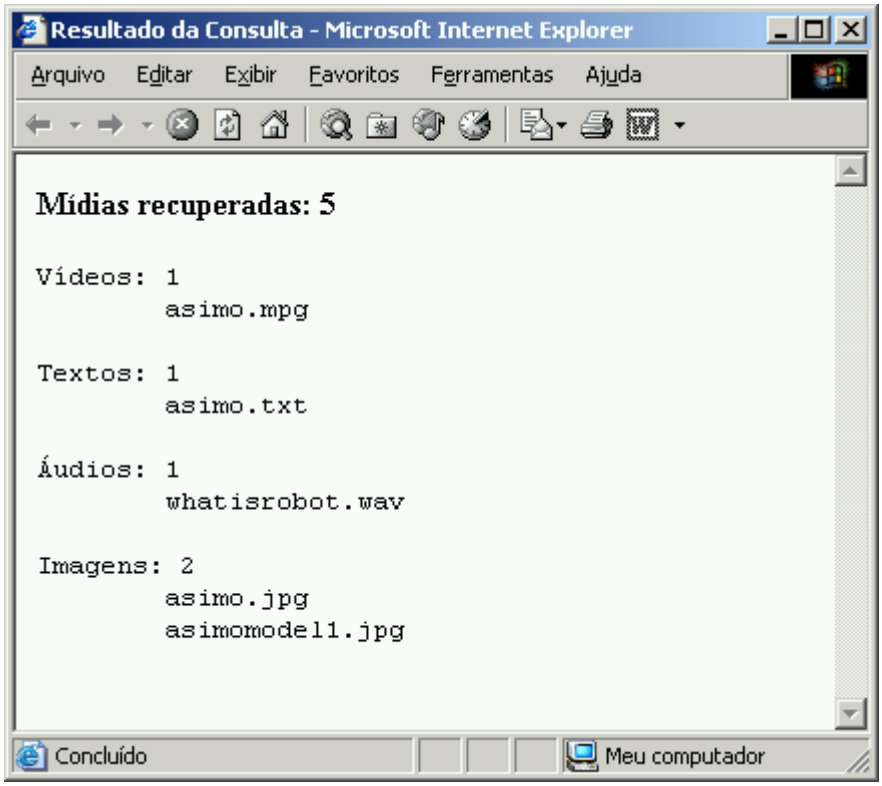


Figura 5.4 - Visualização textual do documento XML

A formatação do documento XML permite inúmeras opções de visualização, com filtragem de dados ou visualização diferenciada para conteúdos diferentes, como, por exemplo, permitindo que textos e imagens sejam exibidos normalmente enquanto áudios e vídeos sejam apenas descritos através de suas informações semânticas. O uso e a forma de apresentação do documento XML vai depender de vários fatores, porém é importante notar que caso uma nova formatação de visualização seja necessária, basta que uma nova Folha de Estilo seja acrescentada às opções de visualização, sem haver necessidade de alteração no processo de construção e manipulação dos documentos XML.

5.4 - Considerações Finais

Este capítulo apresentou formas de visualização do documento XML que foram definidas através do uso de Folhas de Estilo XSL. Os exemplos apresentados visam ilustrar a flexibilidade de XML e demonstrar como as informações semânticas podem ser aproveitadas para prover ao ambiente MAW do projeto AMMO a característica de acessibilidade, que é muito importante para adequação de todo o sistema de manipulação de dados multimídia a diferentes tipos de usuário.

Capítulo 6

Conclusões

6.1 - Resultados Alcançados

No desenolvimento deste trabalho, foram desenvolvidas DTDs XML para representação de informações multimídia com informações semânticas, aproveitando as vantagens do padrão XML, que é voltado diretamente para a Web, para aprimorar a apresentação de dados multimídia recuperados e proporcionar acessibilidade ao ambiente MAW *(Multimedia Application WebBuilder)* do Projeto AMMO.

Para incorporação dos recursos XML ao ambiente MAW foram desenvolvidas as seguintes atividades:

- Implementação de uma arquitetura para utilização de documentos XML para publicação na Web de dados obtidos a partir de um Banco de Dados.
- Mapeamento do BDMm para documentos XML naturalmente utilizáveis na Web.
- Desenvolvimento de DTDs para manipulação de dados multimídia com informações semânticas.
- Implementação de algoritmos para geração automática de documentos XML.

6.2 - Contribuições

O trabalho aqui apresentado traz uma nova perspectiva ao ambiente MAW e a todo o Projeto AMMO, com o uso de DTDs que podem servir como base para integração das várias ferramentas e módulos componentes do projeto, proporcionando a definição e implementação desses módulos independentes de plataforma ou ambientes de programação, utilizando as DTDs XML como base para manipulação e troca de informações. Além disso, a arquitetura utilizada para publicação de dados multimídia na Web através de XML se mostra flexível, já que XML não é um padrão proprietário e pode ser adaptado a vários padrões de base de dados graças à sua natureza extensível.

A flexibilidade para construção de formatos de visualização também contribui para a acessibilidade no processo de apresentação dos dados multimídia, que se constitui em uma característica muito importante para o uso satisfatório do ambiente via Web. Além disso, permite também que haja alterações na forma de disponibilização e apresentação dos dados conforme a evolução do projeto, utilizando um documento XML como base.

Recursos como *namespaces* foram utilizados para generalizar o uso das DTDs desenvolvidas. Os algoritmos implementados para criação automática dos documentos XML permitem que todo o processo de formatação e publicação dos dados seja implementado de forma dinâmica, gerando documentos prontos para a Web. Essa abordagem permite que o trabalho aqui apresentado seja abrangente o bastante para contemplar novas tecnologias e contextos que vêm sendo continuamente acrescentados ao projeto AMMO, como, por exemplo, o desenvolvimento de um ambiente para suporte à Educação a Distância [Seno & Vieira 2000, Seno 2001, Seno & Vieira 2002].

6.3 - Trabalhos Futuros

O desenvolvimento deste trabalho mostrou que os recursos XML apresentados podem ser expandidos para uma utilização mais abrangente dentro do Projeto AMMO. Para tanto, é necessária a total integração dos recursos implementados com os demais módulos do ambiente MAW e o estudo de novas extensões com objetivos mais amplos. Com isso, podem ser vislumbrados os seguintes aspectos a serem aplicados:

- Enriquecimento da arquitetura definida visando abranger consultas realizadas a partir de qualquer módulo do Projeto AMMO, com características particulares.
- Definição de novas DTDs para abranger todas as visões possíveis do BDMm.
- Estudo de interface para definição eficiente das formatações de visualização.
- Implementação de seleção automática de Folha de Estilo conforme perfil do usuário.
- Geração automática de Folhas de Estilo para adequação a novos perfis.
- Estudo de modelagem e uso dos metadados no padrão RDF.

Referências Bibliográficas

Aigrain, P.; Zhang, H.J.; Petkovic, D. (1996). **Content Based Representation and Retrieval of Visual Media: a State of the Art Review.** Multimedia Tools and Applications, vol. 3, September.

Bartlett, P. G. (2001). **Single Source Publishing: XML for Multimedia Data Exchange.** In ContentWire, November 07. URL: http://www.content-wire.com/Home/Index.cfm. [ONLINE Jan/2002].

Berners-Lee, T. (1996). **The World Wide Web: Past, Present and Future**. URL: http://www.w3.org/People/Berners-Lee/1996/ppf.html. [ONLINE Jan/2002].

Borges Jr., S. R. (2001). **Consultas Nebulosas Baseadas em Informações Semânticas em um Banco de Dados Multimídia.** Dissertação de Mestrado – PPGCC-DC – UFSCar, São Carlos, São Paulo, Brasil.

Bosak, J. (1997). **XML, Java, and the Future of the Web.** World Wide Web Journal, Vol. 2, No.4, pp. 219-227, O'Reilly.

Bosak, J.; Bray, T. (1999). **XML and the Second-generation Web.** Scientific American, May. URL: http://www.scientificamerican.com/1999/0599issue/0599bosak.html. [ONLINE Jan/2002].

Bryan, M. (1997). **An Introduction to the Extensible Markup Language (XML)**. SGML Centre. URL: http://www.personal.u-net.com/~sgml/xmlintro.htm. [ONLINE Jan/2002].

Brusilovsky, P. (1996). **Methods and Techniques of Adaptive Hypermedia.** Journal of User Modeling and User Adapted Interaction, Vol. 6, pp. 87-129.

Bulterman, D.C.A. (1997). **Models, Media and Motion: Using the Web to Support Multimedia Documents.** In Proceedings of Multimedia Modeling 97, pp. 227-246, Singapore.

Cody, W. F.; Haas, L. M.; Niblack, W.; Arya, M.; Carey, M. J.; Fagin, R.; Lee, D.; Petkovic, D.; Schwarz, P. M.; Thomas, J.; Roth, M. T.; Williams, J. H.; Wimmers, E. L. (1995). **Querying Multimedia Data from Multiple Repositories by Content: The Garlic Project.** In Proceedings of the International Conference on Visual Database Systems (VDB), pp. 17-35, Lausanne, Switzerland.

CSS W3C - World Wide Web Consortium Style Activity (1996). **Cascading Style Sheets (CSS) 1.0 Specification.** W3C Recommendation, URL: http://www.w3.org/TR/REC-CSS1.

Figueiredo, J. M. (2000). **Desenvolvimento de um Ambiente para Autoria e Manipulação de Aplicações Multimídia na World Wide Web.** Dissertação de Mestrado – PPGCC-DC – UFSCar, São Carlos, São Paulo, Brasil.

van Harmelen, F.; Fensel, D. (1999). **Practical Knowledge Representation for the Web.** In Proceedings of the IJCAI'99 Workshop on Intelligent Information Integration, Stockholm, Sweden. URL:http://www.cs.vu.nl/~frankh/postscript/IJCAI99-III.html. [ON LINE Jan/2002]

ISO - International Organization for Standardization (1986). **Information Processing – Text and Office Systems – Standard Generalized Markup Language (SGML)**. ISO/IEC 8879.

ISO - International Organization for Standardization (1996a). **Information Processing – Text and Office Systems – Document Style Semantics and Specification Language (DSSSL)**. ISO/IEC 10179.

ISO - International Organization for Standardization (1996b). **Information Technology Coding of Multimedia and Hypermedia Information, Part 5: Support for Base-Level Interactive Applications**. ISO/IEC 13522-5.

Kleiner, C.; Lipeck, U. W. (2001). **Automatic Generation of XML DTDs from Conceptual Database Schemas.** In Proceedings of WEBDB 2001, pp. 396-405, Vienna, Austria.

Light, R. (1999). **Iniciando em XML.** São Paulo, Makron Books.

Mace, S.; Flohr, U.; Dobson, R.; Graham, T. (1998). **Weaving a Better Web.** Byte Magazine. Vol. 23, N. 3, pp. 58-68.

Macedo, M. R. (1999). **MHEG-5/SMIL: Um Ambiente de Conversão entre os Padrões.** Dissertação de Mestrado – PPGCC-DC – UFSCar, São Carlos, Brasil, 136 pgs.

Marchal, B. (2000). **XML - Conceitos e Aplicações.** São Paulo, Berkeley Brasil.

O2 Technology (1995). **O2 Beginner´s Guide, release 4.6.**

OMG - Object Management Group (2001). **Unified Modeling Language (UML) Specification.** URL: http://www.omg.org/technology/documents/formal/uml.htm. [ONLINE Jan/2002].

Ozden, B.; Rastogi, R.; Silberschatz, A. (1997). **Multimedia Support for Databases.** In Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 1-11, Tucson, Arizona.

Pimentel, M.G.C.; Pires, D.F.; Teixeira, C.A.C. (2000). **XRot: Um Roteiro para a Autoria de Aplicações Interoperáveis para a Web.** Anais do VI Simpósio Brasileiro de Sistemas Multimídia e Hipermídia, pp. 195-212, Natal, Brasil.

Rakow, T.C.; Neuhold, E.J.; Lohr, M. (1995). **Multimedia Database Systems - the Notion and the Issues.** In Proceedings of BTW, Springer-Verlag, pp. 1-29, Berlin, Germany.

RDF W3C - RDF Core Working Group of the World Wide Web Consortium (1999). **Resource Description Framework Model and Syntax Specification.** W3C Recommendation, URL: http://w3.org/TR/REC-rdf-syntax/.

Rousseau, F.; Macías, J. A. G.; Lima, J. V.; Duda, A. (1999). **User Adaptable Multimedia Presentations for the WWW.** Computer Networks, Vol. 31, N. 11, pp. 1273-1290.

Sall, K. (1998). **XML - Structuring Data for the Web: an Introduction.** URL: http://www.wdvl.com/Authoring/Languages/XML/Intro/. [ON LINE Jan/2002]

Santos, M..T. P.; Vieira, M.T.P.; Figueiredo, J.M. (1997a). **Extensão de um Banco de Dados de Objetos MHEG-5 para Suportar Busca por Conteúdo.** Anais do XII Simpósio Brasileiro de Banco de Dados, pp. 107-121, Fortaleza, Brasil.

Santos, M.T.P., Santos, F.C., Vieira, M.T.P. (1997b). **A MHEG-5 Objects Server for Multimedia Applications.** In Proceedings of the ICAST 97 & ICMIS 97, pp.283-289, Schaumburg, Illinois, USA.

Santos, M.T.P.; Vieira, M.T.P. (1998). **Sistema de Recuperação de Informações de um Servidor de Objetos MHEG-5**. Anais do IV Simpósio Brasileiro de Sistemas Multimídia e Hipermídia, pp. 249-260, Rio de Janeiro, Brasil.

Santos, M.T.P.; Vieira, M.T.P.; Borges, S.R.; Figueiredo, J.M.; Fornazari, F.P.; Biajiz, M. (2000). **Semantic Infomation Search Facilities for MHEG-5 and SMIL Applications.** In proceedings of FQAS 2000, pp. 315-325, Warsaw, Poland.

Seno, W. P.; Vieira, M.T.P. (2000). **Uma arquitetura de um Ambiente de Educação a Distância.** Relatório Técnico RT-DC 001/2001 – Departamento de Computação – UFSCar, São Carlos, São Paulo, Brasil.

Seno, W. P. (2001). **Modelo e base de dados para automatização do planejamento e execução de cursos em ambientes de EAD.** Dissertação de Mestrado – PPGCC-DC – UFSCar, São Carlos, São Paulo, Brasil.

Seno, W. P.; Vieira, M.T.P. (2002). **DE-MAW - Um Ambiente de Educação a Distância.** Revista Brasileira de Informática na Educação - RBIE, Volume 10, N. 1, pp. 21-29.

SMIL W3C - Synchronized Multimedia Working Group of the World Wide Web Consortium (1998). **Synchronized Multimedia Integration Language (SMIL) 1.0 Specification.** W3C Recommendation, URL: http://www.w3.org/TR/REC-smil.

Teixeira, C.A.C.; Vieira, M.T.P.; Araújo, R.B.; Trevelin, L.C.; Pimentel, M.G.C.; Sena, C.V.G. (1995). **Arquitetura do Projeto SMmD.** Anais do I Workshop em Sistemas Hipermídia Distribuídos, pp. 30-39, São Carlos, Brasil.

Teixeira, C.A.C.; Pimentel, M.G.C.; Vieira, M.T.P.; Santos, M.T.P.; Abrão, I.C.; Barrére, E.; Baldochi Jr, L. (1998). **SMmD – A MHEG-5 Based Distributed Multimedia System.** In Proceedings of the ProTeM-CC – Projects Fase II, pp. 239-260, Belo Horizonte, Brazil.

Teixeira, E. C.; Vieira, M. T. P. (2001). **XML: Criação de Documentos e Integração com Bancos de Dados.** Relatório Técnico RT-DC 003/2001 – Departamento de Computação – UFSCar, São Carlos, São Paulo, Brasil.

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, pp. 154-159, Toulouse, France.

Vieira, M.T.P; Biajiz, M.; Santos, M.T.P.; Ribeiro, L.C.; Fornazari, F.P. (1999). **Metadata for Content-Based Search on an MHEG-5 Multimedia Objects Server.** In Proceedings of the Third IEEE Metadata Conference, Maryland, USA. URL: http://computer.org/proceedings/meta/1999/papers/32/MVieira.html. [ON LINE Jan/2002]

Vieira, M.T.P.; Biajiz, M.; Borges, S.R.; Teixeira, E.C.; Santos, F.G.; Figueiredo, J.M. (2002). **Content-Based Fuzzy Search in a Multimedia Web Database.** In Proceedings of Intelligent Exploration of the Web, Springer-Verlag. (To appear)

XML W3C - World Wide Web Consortium XML Activity (1998). **Extensible Markup Language (XML) 1.0.** W3C Recommendation, URL: http://www.w3.org/TR/REC-xml.

XSL W3C - Extensible Style Sheet Working Group of the World Wide Web Consortium (2001). **Extensible Stylesheet Language (XSL) 1.0.** W3C Recommendation, URL: http://www.w3.org/TR/xsl/.

Walsh, N. (1997). **A Technical Introduction to XML.** World Wide Web Journal, Vol.2, No.4 , O'Reilly.

Apêndice A

1 - Exemplos de Código das Folhas de Estilo XSL utilizadas

1.1 - Folha de Estilo utilizada no exemplo apresentado no Quadro 5.1

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"
                xmlns:sem="semantic.dtd">
  <xsl:output method="html" indent="yes"/>

<xsl:template match="mmdata">
<html>
<head>
       <script type="text/javascript">
              function irPara(nome)
              {
                     PageIndex=nome.select.selectedIndex
                     if (nome.select.options[PageIndex].value != "none")
                     {
                            location = nome.select.options[PageIndex].value
                     }
              }
       </script>
       <title> Resultado da Consulta </title>
</head>
<body>
       <xsl:apply-templates select="mediaobject"/>
</body>
</html>
</xsl:template>

<xsl:template match="mediaobject">
     <table>
       <tr>
            <td>
              <center><b><xsl:value-of select="@name"/></b></center><br/>
              <xsl:if test="self::node()[@type='Image']">
              <img><xsl:attribute name="src">
                     <xsl:value-of select="@mediasource"/></xsl:attribute></img>
              </xsl:if>
              <xsl:if test="self::node()[@type='Video']">
                   <xsl:if test="self::node()[@extension='avi']">
                   <embed ShowControls="true" AutoStart="false">
                     <xsl:attribute name="src">
                     <xsl:value-of select="@mediasource"/></xsl:attribute>
                   </embed>
                   </xsl:if>
                   <xsl:if test="self::node()[@extension='rm']">
                   <embed  type="audio/x-pn-realaudio-plugin"  CONTROLS="ImageWindow"
                     HEIGHT="240" WIDTH="320" AUTOSTART="true" NOJAVA="true">
                     <xsl:attribute name="src">
                            <xsl:value-of select="@mediasource"/></xsl:attribute>
                   </embed>
                   </xsl:if>
              </xsl:if>
              <xsl:if test="self::node()[@type='Audio']">
                   <embed ShowControls="true" AutoStart="false">
                     <xsl:attribute name="src">
                            <xsl:value-of select="@mediasource"/></xsl:attribute>
                   </embed>
```

```
              </xsl:if>
              <xsl:if test="self::node()[@type='Text']">
                     <iframe scrolling="auto" width="250" height="150">
                     <xsl:attribute name="src">
                           <xsl:value-of select="@mediasource"/></xsl:attribute>
                     <xsl:text>   </xsl:text>
                     </iframe>
              </xsl:if>
            </td>
            <td>
              <table>
              <tr><th>Cenas:</th><th>Aplicações:</th></tr>
              <xsl:apply-templates select="scene"/>
              </table>
            </td>
      </tr>
     </table>
<hr></hr>
</xsl:template>

<xsl:template match="scene">
       <tr>
       <td valign="center">
       <a> <xsl:attribute name="href">
              <xsl:value-of select="@documentsmil"/></xsl:attribute>
       <xsl:value-of select="@documentsmil"/></a>
       </td>
       <td valign="center">
       <br/>
       <form> <xsl:attribute name="name">
              <xsl:value-of select="generate-id()"/></xsl:attribute>
       <select name="select">
       <xsl:for-each select="application">
              <option>
              <xsl:attribute name="value">
                     <xsl:value-of select="@firstscene"/>
              </xsl:attribute>
              <xsl:value-of select="@firstscene"/>
              </option>
       </xsl:for-each>
       <input type="button" name="b1" value="Ver"> <xsl:attribute name="onclick">
       irPara(<xsl:value-of select="generate-id()"/>)</xsl:attribute>
       </input>
       </select>
       </form>
       </td>
       </tr>
</xsl:template>

</xsl:stylesheet>
```

1.2 - Folha de Estilo utilizada no exemplo apresentado no Quadro 5.2

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"
                xmlns:sem="semantic.dtd">
  <xsl:output method="html" indent="yes"/>

<xsl:template match="mmdata">
<html>
<head>
       <title>Cena:<xsl:value-of select="mediaobject/scene/@documentsmil"/></title>
</head>
<body>
       <a> <xsl:attribute name="href">
       <xsl:value-of select="mediaobject/scene/@documentsmil"/></xsl:attribute>
       Ver cena </a><br/>
       <h4>Mídias presentes na cena: </h4>
       <xsl:apply-templates select="mediaobject"/>
</body>
</html>
</xsl:template>

<xsl:template match="mediaobject">
     <table>
       <tr>
            <td>
              <b><xsl:value-of select="@name"/></b><br/>
              <xsl:apply-templates select="sem:semantic"/>
            </td>
       </tr>
     </table>
<hr></hr>
</xsl:template>

<xsl:template match="sem:semantic">
             <xsl:for-each select="sem:association">
             <xsl:value-of select="sem:subjstart/sem:subject/sem:identifier"/>
                    <xsl:for-each select="sem:subjstart/sem:subject/sem:qualifier">
                          <xsl:text> </xsl:text><xsl:value-of select="."/>
                    </xsl:for-each>
      <xsl:text> </xsl:text> <xsl:value-of select="@direct"/><xsl:text> </xsl:text>
                    <xsl:value-of select="sem:subjend/sem:subject/sem:identifier"/>
                    <xsl:for-each select="sem:subjend/sem:subject/sem:qualifier">
                          <xsl:text> </xsl:text><xsl:value-of select="."/>
                    </xsl:for-each>
                    <br/>
             </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Apêndice B

1. Representando recursos Web com expressões RDF

O modelo do padrão RDF foi baseado na forma direta de expressão de informações em linguagem natural. Como exemplo, a frase a seguir fornece informações a respeito da página Web do Programa de Pós-Graduação em Ciência da Computação da UFSCar (PPG-CC / UFSCar):

A página http://www.dc.ufscar.br/posgrad/ tem como responsável o Coordenador do PPG-CC / UFSCar.

Nessa expressão podemos identificar três elementos: o *recurso* que vai ser descrito (a página), a *propriedade* que vai caracterizá-lo (o responsável), e o *valor* dessa propriedade (Coordenador PPG-CC). Para identificar o recurso foi usada sua URL, que é um identificador único. O termo "responsável" identifica a propriedade abordada, e "Coordenador PPG-CC" identifica o valor da propriedade "responsável".

No padrão RDF cada parte da expressão recebe um nome particular. O termo que indica o recurso descrito é chamado de *sujeito*. O identificador da característica descrita pela expressão é o *predicado*. O valor da propriedade é chamado de *objeto*. Então, tomando a expressão em Português usada como exemplo, temos:

Sujeito *Predicado* *Objeto*

A página http://www.dc.ufscar.br/posgrad/ tem como responsável o Coordenador do PPG-CC / UFSCar.

Através desse conceito, as expressões podem ser modeladas como um digrafo (grafo direcionado) rotulado, onde os nós descrevem recursos que podem ser rotulados com URIs, strings literais ou vazios, e os arcos conectam os nós e são rotulados com URIs. Cada arco direcionado pode ser descrito como uma tupla ternária formada pelo sujeito (no início do arco), pela propriedade do arco e pelo objeto (no fim do arco).

O grafo RDF para a expressão usada como exemplo, com uma URI definida para a propriedade “responsável” é:

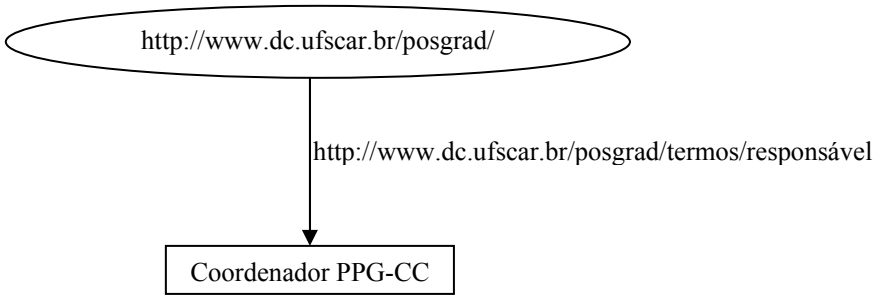


Figura B1 - Representação de dados RDF em grafo

Este grafo corresponde à seguinte representação em tupla ternária:

<http://www.dc.ufscar.br/posgrad/>
< http://www.dc.ufscar.br/posgrad/termos/responsável>
“Coordenador PPG-CC”.

Para codificar os grafos em XML, os nós e arcos são transformados em elementos e atributos, e as URIs são escritas em XML através de *namespaces*. As URIs que rotulam os nós sujeito são armazenadas como valores de atributos em XML. Os nós rotulados por strings literais (os quais são sempre nós objeto) se tornam conteúdo textual de elementos ou valores de atributos.

O documento RDF/XML correspondente à expressão usada como exemplo é:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:ppgcc="http://www.dc.ufscar.br/posgrad/termos/">
     <rdf:Description rdf:about="http://www.dc.ufscar.br/posgrad/">
           <ppgcc:responsavel>Coordenador PPG-CC</ppgcc:responsavel>
     </rdf:Description>
</rdf:RDF>
```

Quadro B1 - Documento RDF/XML sobre página http://www.dc.ufscar.br/posgrad/

Como o documento apresentado (Quadro B1) é um documento XML, a primeira linha apresenta a declaração XML. A linha seguinte inicia um elemento `rdf:RDF`, seguido pela declaração de *namespace* XML que especifica que todas as marcações neste conteúdo

prefixadas com `rdf:` são parte desse *namespace*, identificado pela URI `http://www.w3.org/1999/02/22-rdf-syntax-ns#`, que é a fonte para os termos específicos de RDF.

Logo após a declaração de *namespace* para o prefixo `rdf:`, há a declaração de outro *namespace* XML, desta vez para o prefixo `ppgcc:`, que é associado ao nome de *namespace* `http://www.dc.ufscar.br/posgrad/termos/`. Este *namespace* é a fonte para os termos específicos definidos no *site* do PPG-CC/UFSCar.

Após a declaração do documento XML e das declarações de *namespaces,* é iniciada a descrição RDF. O elemento `rdf:Description` indica o início de uma descrição, e segue definindo o recurso a ser descrito usando o atributo `rdf:about` para especificar sua URI. O elemento `<ppgcc:responsavel>` que aparece em seguida contém o valor "`Coordenador PPG-CC`", que é o valor da propriedade `autor` na expressão de exemplo. O elemento `responsavel` está aninhado dentro do elemento precendente `rdf:Description`, indicando que se aplica ao recurso especificado no conteúdo do elemento `rdf:Description`. Finalmente, as últimas linhas indicam o fim dos elementos `rdf:Description` e `rdf:RDF`.

No exemplo adotado, foram descritos recursos com URIs já definidas, sendo que a referência ao recurso foi feita através do atributo `rdf:about`. Porém, existem situações em que o recurso a ser descrito não possui uma URI própria, como no caso em que queremos referenciar apenas parte de um documento. Adotando como exemplo a mesma URI usada anteriormente (http://www.dc.ufscar.br/posgrad/), são construídas expressões para descrever os pesquisadores envolvidos no PPG-CC/UFSCar. Essas descrições, em RDF, compõem um documento identificado pela URI http://www.dc.ufscar.br/posgrad/pesquisadores. Um exemplo desse documento contendo apenas uma dessas descrições é:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:ppgcc="http://www.dc.ufscar.br/posgrad/termos/">
      <rdf:Description rdf:ID="Pesq1">
            <ppgcc:nome>Marina</ppgcc:nome>
            <ppgcc:sobrenome>Teresa Pires Vieira</ppgcc:sobrenome>
            <ppgcc:titulacao>Doutora</ppgcc:titulacao>
      </rdf:Description>
</rdf:RDF>
```

Quadro B2 - Documento RDF/XML sobre página http://www.dc.ufscar.br/posgrad/pesquisadores

Este formato de documento é similar aos anteriores por representar, da mesma forma, as propriedades (nome, sobrenome e titulação) de um recurso sendo descrito. Entretanto, o elemento `rdf:Description` do Quadro B2 tem um atributo `rdf:ID` ao invés de um atributo `rdf:about` (que é usado no exemplo do Quadro B1). O atributo `rdf:ID` indica a descrição de um novo recurso identificado por um identificador simples, ao invés de estarmos nos referindo a um recurso existente através de sua URI. O identificador deve ser único dentro do recurso no qual ele está definido (neste caso, http://www.dc.ufscar.br/posgrad/pesquisadores).

Definido dessa forma, outros documentos RDF podem referenciar um pesquisador do PPG-CC/UFSCar concatenando o identificador (`Pesq1`) a URI base da página Web, formando a URI `http://www.dc.ufscar.br/posgrad/pesquisadores#Pesq1`. Se houver necessidade de, por exemplo, agrupar em uma mesma página dados dos pesquisadores envolvidos em projetos do Grupo de Banco de Dados, isso pode ser feito da seguinte forma:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:gbd="http://www.dc.ufscar.br/gbd/termos/">
      <rdf:Description
       rdf:about="http://www.dc.ufscar.br/posgrad/pesquisadores#Pesq1">
            <gbd:instituicao>UFSCar</gbd:instituicao>
            <gbd:email>marina@dc.ufscar.br</gbd:email>
   </rdf:Description>
</rdf:RDF>
```

Quadro B3 - Documento RDF/XML com referência a identificador externo

No exemplo do Quadro B3, foi usado o elemento `rdf:Description` com um atributo `rdf:about`, cujo valor é a URI do pesquisador definido na descrição RDF do Quadro B2.

Os exemplos apresentados aqui mostram não apenas como funciona o padrão RDF, mas sim suas possibilidades, ilustrando um dos princípios básicos de RDF, que é o de que qualquer recurso existente na Web pode ser descrito com base em um padrão comum.