UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE CIÊNCIAS E TECNOLOGIA - CCT
DEPARTAMENTO DE SISTEMA E COMPUTAÇÃO - DSC
CURSO DE PÓS-GRADUAÇÃO EM INFORMÁTICA - COPIN

DISSERTAÇÃO DE MESTRADO

VIDEOLIB: UMA BIBLIOTECA DIGITAL DE VÍDEO USANDO METADADOS DUBLIN CORE E MPEG-7

ALEX SANDRO DA CUNHA RÊGO

CAMPINA GRANDE
FEVEREIRO – 2004

ALEX SANDRO DA CUNHA RÊGO

VIDEOLIB: UMA BIBLIOTECA DIGITAL DE VÍDEO USANDO METADADOS DUBLIN CORE E MPEG-7

Dissertação submetida ao Curso de Pós-Graduação em Informática do Centro de Ciências e Tecnologia da Universidade Federal de Campina Grande, como requisito parcial para obtenção do grau de Mestre em Informática (MSc).

Área de Concentração: Ciência da Computação

Linha de Pesquisa: Sistemas de Informações e Banco de Dados

DR. CLÁUDIO DE SOUZA BAPTISTA

(ORIENTADOR)

CAMPINA GRANDE
FEVEREIRO – 2004

RÊGO, Alex Sandro da Cunha

R343V

VideoLib: Uma Biblioteca Digital de Vídeo Usando Metadados Dublin Core e MPEG-7

Dissertação (Mestrado), Universidade Federal de Campina Grande, Centro de Ciências e Tecnologia, Coordenação de Pós-Graduação em Informática, Campina Grande – Paraíba, Fevereiro de 2004.

119 p. Il.

Orientador: Cláudio de Souza Baptista

Palavras-chaves:

1. Banco de Dados
2. Aplicação Web
3. XML
4. XML Schema
5. Indexação de vídeo

CDU – 681.3.07B

VIDEOLIB: UMA BIBLIOTECA DIGITAL DE VÍDEO USANDO METADADOS DUBLIN CORE E MPEG-7

ALEX SANDRO DA CUNHA RÊGO

Dissertação aprovada em 27/02/2004

PROF. CLÁUDIO DE SOUZA BAPTISTA, Ph.D
Orientador

PROF. ULRICH SCHIEL, Dr.
Examinador

PROF. ED PORTO BEZERRA, Dr.
Examinador

CAMPINA GRANDE - PB

“O tempo não é algo que possa voltar para trás. Portanto, plante seu jardim e decore sua alma, ao invés de esperar que alguém lhe traga flores” (Shakespeare).

Dedico este trabalho à minha mãe **Alice,** por estar sempre presente nos mais diversos momentos da minha vida.

AGRADECIMENTOS

Primeiramente a **Deus**, por me dar forças e abrir novos horizontes na minha vida.

A todos os meus **familiares,** pelo incentivo de ingressar no mestrado.

Aos amigos **Philip, Claudivan, Rodrigo César, Eloi e Wesley,** pelos momentos de companheirismo e descontração.

Às amigas **Gizelle, Tatiane e Kátia**, pela companhia, amizade e carinho.

Ao meu orientador **Cláudio Baptista** pela amizade, competência, paciência e conhecimentos transmitidos.

Aos amigos do primeiro apartamento, **Kleber Melo, Alisson, Heronides (Lula), Petrônio (Tota) e Daniel (BA)**, pelo excelente convívio e momentos de divertimento.

A todo o pessoal do LSI (Laboratório de Sistemas de Informações), especialmente a **Rebouças**, **Vilar** e **Elvis**, pelas diversas trocas de idéias.

A CAPES pelo subsídio financeiro.

A todas as pessoas que torcem pelo meu sucesso.

RESUMO

Uma grande quantidade de vídeos pode ser encontrada em formato digital na web, e este número vem crescendo significativamente. A manipulação de vídeos torna-se complexa devido ao grande volume, heterogeneidade semântica interpretada por diferentes usuários e pela carência de padrões adequados de indexação, resultando num esforço maior o trabalho de identificação, busca e gerenciamento dos recursos. Metadados devem ser associados aos vídeos para permitir eficiente identificação, indexação e representação semântica. Os padrões Dublin Core e MPEG-7 fornecem um conjunto de metadados que possibilita a descrição de conteúdo multimídia. Nesta dissertação, apresentamos uma biblioteca digital, denominada VideoLib, que utiliza descrições audiovisuais baseadas em Dublin Core e MPEG-7 para fazer recuperação de conteúdo de vídeo, fornecendo suporte às dimensões temporal e espacial.

Palavras-chave: Banco de Dados; Aplicação Web; XML; XML Schema; Indexação de Vídeo.

ABSTRACT

Recently, a large amount of digital video can be found on the Web, and this number is increasing rapidly. The video manipulation becomes complex due to the great volume, semantic heterogeneity interpreted by different users and by the lack of adequate standards of indexing, resulting in a large effort of identification, search and management of the resources. Metadata must be associated to video in order to enable an efficient identification, indexing and semantic representation of it. Both Dublin Core and MPEG-7 standards provide a metadata set which enables the description of multimedia content. This paper presents a digital video library, known as VideoLib, which utilizes Dublin Core and MPEG-7 audiovisual descriptions to provide video content retrieval, with support to spatial and temporal dimensions.

Keywords: Database; Web Application; XML; XML Schema; Video Index.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE GRÁFICOS

LISTA DE SIGLAS

API – Application Programming Interface

D – Descriptor

DC – Dublin Core

DDL – Data Definition Language

DS – Descriptor Scheme

GUI – Graphical User Interface

HTML – HyperText Markup Language

IEEE – Institute of Electrical and Electronics Engineers

JDK – Java Development Kit

JSP – Java Server Pages

JXQI – Java XQuery API

MPEG – Moving Picture Expert Group

MVC – Model-View-Controller

SGBD – Sistema de Gerência de Banco de Dados

SQL – Structured Query Language

UML – Unified Modeling Language

URI – Universal Resource Identifier

URL – Uniform Resource Locator

W3C – World Wide Web Consortium

XML – eXtensible Markup Language

XSLT – Extensible Stylesheet Language Transformations

SUMÁRIO

Capítulo 1

Introdução

Com o crescimento da quantidade de vídeos que vem sendo disponibilizada sob a forma digital na Web, torna-se evidente a necessidade da utilização de soluções de software que proporcionem uma eficiente criação, armazenamento e recuperação de tais conteúdos audiovisuais. A criação de ferramentas de busca multimídia implica em diversos desafios (SWAIN, 1999). Um deles refere-se à forma que deve ser empregada para representar e indexar conteúdo multimídia. Juntamente com essa questão, deve-se definir qual interface deverá ser apresentada ao usuário, incluindo a forma de consulta à representação audiovisual, e como a lista de resultados deve ser resumida para apresentação.

A manipulação de vídeos torna-se complexa devido ao grande volume (com relação ao tamanho) e à carência de padrões adequados de indexação, resultando num esforço maior o trabalho de identificação, busca e gerenciamento dos recursos (HUNTER, 2002).

Bibliotecas digitais[1] têm sido usadas para prover acesso a uma grande quantidade de informações, representadas em diversos formatos digitais, tais como: imagens, vídeos, áudios, mapas, e textos. A utilização de aplicações dessa natureza proporciona as seguintes vantagens em comparação com as bibliotecas tradicionais:

1- Todas as operações são efetuadas através de um computador;
2- O formato digital permite facilidades na manipulação e edição;
3- Um documento pode produzir um número ilimitado de cópias sem perder a qualidade e não se desgasta com o tempo e manuseio;
4- Os dados podem ser recuperados remotamente e em qualquer lugar do mundo;
5- Pode estar disponível 24 horas por dia, 7 dias por semana.

A recuperação eficiente da informação é um dos principais objetivos das bibliotecas digitais. Uma pessoa pode procurar por um determinado recurso de seu interesse, fornecendo parâmetros de entrada, e o mecanismo de busca do sistema deve ser capaz de examinar sua coleção e disponibilizar o conteúdo desejado. Um aspecto muito importante no âmbito das

[1] Arms (ARMS, 2001) define biblioteca digital como uma coleção ordenada de informações com serviços associados, onde a informação é armazenada em formato digital e acessível através de uma rede de comunicação.

bibliotecas digitais diz respeito à utilização de metadados[2] para a descrição de recursos. Metadados são normalmente expressos na forma textual e freqüentemente armazenados separadamente dos objetos que descrevem. Sua utilização não está limitada apenas a documentos textuais, podendo ser usado para descrever informações relativas a diversas mídias.

Com os avanços nas pesquisas relacionadas à compressão de dados, armazenamento e telecomunicações, tornou-se possível o desenvolvimento de bibliotecas digitais de vídeos. Porém, uma biblioteca digital de vídeos perde um pouco do seu valor se ela não apresenta uma compreensão do que está nela contido (CHRISTEL, 1995). Sem esse entendimento semântico, um usuário teria que empregar uma boa quantidade de tempo de observação para determinar se um recurso de seu interesse encontra-se em uma biblioteca digital de vídeos que pode conter centenas de horas de conteúdo.

Assim, novas técnicas são necessárias para organizar e encontrar estas vastas coleções de dados, recuperar as partes mais significativas e efetivamente usá-las (CHRISTEL *et al*, 1996). Para isso, um fator muito importante diz respeito à questão da indexação do vídeo, que vai permitir que o conteúdo da mídia não seja acessado de forma linear, possibilitando o acesso à seqüência de *frames*[3] que atende às necessidades do usuário. Neste caso, faz-se necessário adicionar informações semânticas ao conteúdo audiovisual, tornando-se essencial produzir metadados e associá-los às suas respectivas mídias (CHRISTEL; WACTLAR, 2002; KIM; SONG 2001).

Para facilitar a indexação, um vídeo deve ser decomposto hierarquicamente em seqüências, cenas e *shots*[4] (IANNELLA; HUNTER, 1998). Uma seqüência pode ser definida como uma árvore onde cada nodo corresponde a um intervalo de tempo (cena), que por sua vez é particionado em sucessivos sub-intervalos temporais contínuos (*shots*). Cada cena é composta por um ou mais *shots*, que representam diferentes visões do mesmo evento. Muitas características de baixo nível[5] podem ser extraídas de forma automática, tais como duração, número de *frames*, localização e tamanho do vídeo. Entretanto, a extração de informações e análise semântica constitui, na atualidade, uma tarefa centrada no usuário, visto que a

---

[2] Metadados são "dados estruturados sobre os dados". Essencialmente, metadado é uma forma de unir informações para descrever um item de dado.

[3] Quadros de vídeo.

[4] Um *shot* representa uma captura de imagens ininterrupta obtida por uma câmera em um determinado período de tempo (tomada de cena).

[5] Características que podem ser extraídas diretamente da representação digital, de forma automática, sem a interação humana.

extração automática de semântica ainda é considerada uma tarefa muito complexa para ser exclusivamente executada por computadores (SMEATON, 2000). Deste modo, a aplicação de um padrão de metadados baseado em conteúdo audiovisual simplificaria significativamente o desenvolvimento de sistemas de recuperação capazes de localizar arquivos multimídia.

1.1 Motivação

Embora exista um grande volume de dados multimídia que pode ser encontrado na Web, verifica-se uma carência de sistemas que possam indexar, disponibilizar e permitir a manipulação destes.

Estes sistemas devem ser capazes de recuperar conteúdo audiovisual que exija uma certa interpretação semântica acerca do seu conteúdo, algo do tipo: “Quais os jogos de futebol do Brasil que foram realizados na América do Sul nos primeiros seis meses do ano de 2003?”, “Quais as cenas que possuem um evento de decolagem de um avião?” ou também: “Quais os vídeos disponíveis que contém o cantor Djavan tocando violão?”. O mais importante é que o resultado destas pesquisas permita que apenas trechos de interesse do usuário sejam disponibilizados para visualização, sem que seja necessário percorrer todo o vídeo até encontrar o ponto de seu interesse.

Como podemos notar, existe a necessidade de se adotar uma abordagem em que uma determinada consulta possa ser expressa por meio de palavras-chaves, de maneira que a tradução do conteúdo semântico seja feita pelo sistema e possibilite a recuperação adequada das informações. Para que isto seja possível, é necessário que sejam produzidas descrições textuais – utilizando um padrão de metadados adequado – que possibilitem um eficiente acesso e representação do conteúdo audiovisual. Na modelagem de descrições audiovisuais, a representação semântica do conteúdo e utilização de mecanismos de indexação devem ser considerados.

A característica chave que torna os dados de um vídeo diferentes de outros dados, como texto, imagem e mapa, diz respeito à sua dimensão temporal. Um problema crítico que deve ser levado em conta quando modelam-se anotações audiovisuais, consiste na heterogeneidade semântica que pode surgir devido a diferentes interpretações de informações, em uma cena de um vídeo, por diferentes usuários (AL-KHATIB *et al*, 1999), unido à dificuldade de traduzir o conteúdo codificado para seu equivalente sob a forma textual (YEO;YEUNG, 1997). A pessoa responsável pela produção da descrição, fatalmente

não poderá cobrir todas as possibilidades que venham a ser interpretadas em uma cena, e essa lacuna pode originar inconsistência no processo de busca por dados multimídia.

A extração de conhecimento útil e de forma precisa para a representação de conteúdo audiovisual ainda consiste em uma tarefa extremamente difícil (MELAND *et al* 2003). Obter informações sobre "quem" ou "o quê" está sendo mostrado no vídeo, que eventos ou ações estão acontecendo, etc., exigem interpretações de alto nível[6] e metadados estruturados para descrever a semântica do conteúdo de vídeos.

O padrão de metadados MPEG-7 (ISO/IEC, 2002), desenvolvido pelo grupo MPEG (*Moving Picture Expert Groups*), surgiu com o objetivo de fornecer uma interface para a descrição de dados de conteúdo multimídia que suporte algum grau de interpretação de informações, e que possa ser acessado por dispositivos eletrônicos ou código computacional. MPEG-7 não é direcionado para um tipo de aplicação em particular, porém, os elementos que ele padroniza, dão suporte ao desenvolvimento de uma grande faixa de aplicações.

Problemas na recuperação de conteúdo audiovisual podem ser resolvidos, dentro de suas limitações, através da representação de informações por meio da utilização de metadados do padrão MPEG-7. Ao invés de realizar uma busca manipulando diretamente o objeto codificado, a pesquisa seria feita através do acesso aos metadados que compõem a descrição textual do vídeo. A utilização do padrão Dublin Core (Apêndice C), que fornece um conjunto de elementos projetados para promover a descoberta de recursos eletrônicos[7], pode ser combinado para prover uma maior interoperabilidade.

Atualmente, existem poucas aplicações comerciais que lidam com a recuperação de conteúdo visual utilizando o padrão MPEG-7, até mesmo porque a utilização do padrão ainda vem sendo alvo de muitas pesquisas. Além disso, as aplicações existentes apresentam algumas carências com relação à flexibilidade de combinar diversos metadados no processo de elaboração de consultas, e pela ausência de uma recuperação que possibilite o uso de operações de geoprocessamento, e relacionamentos temporais na seleção de produções audiovisuais.

À medida que aumenta a quantidade de descrições multimídia, torna-se evidente a necessidade de desenvolver soluções que possibilitem a seleção de informações que exprimam relações de tempo, por exemplo, sua data de criação ou disponibilização. Da

[6] Conceitos que necessitam da percepção humana.

[7] Qualquer objeto que tenha identidade e possa ser descrito.

mesma forma, a implementação de um *Gazetteer*[8] facilita a recuperação de informações espaciais utilizando o nome de um lugar geográfico. Diante disso, sistemas com suporte para a realização de consultas espaciais e temporais precisam ser desenvolvidos.

1.2 Objetivos

Este trabalho tem como principal objetivo o desenvolvimento de uma biblioteca digital de vídeo que proporcione uma eficiente indexação e recuperação de conteúdo multimídia, com a modelagem e implementação de uma aplicação Web que torne possível a realização de consultas espaço-temporais em metadados MPEG-7 e Dublin Core.

Nesta dissertação, desenvolvemos uma biblioteca digital de vídeos, denominada VideoLib, que possui as seguintes características:

- Utilização de um SGBD (Sistema de Gerência de Banco de Dados) Objeto-Relacional com suporte às operações espaciais *Inside*, *Outside* e *Overlaps*;
- Apresentação de uma árvore para seleção do local geográfico: a definição da localização geográfica que está sendo pesquisada é representada por uma árvore constituída por diferentes níveis hierárquicos de lugares;
- Suporte a operações temporais: existência de recursos que possibilitam a execução de operações temporais sobre as descrições audiovisuais;
- Interface gráfica que possibilita a elaboração de diferentes formas de pesquisa: o usuário pode combinar o conteúdo de caixas de texto para formular sua consulta e submeter para processamento;
- Indexação de vídeo: os vídeos são segmentados hierarquicamente em cenas e *shots*, possibilitando o acesso a parte de seu conteúdo;
- Visualização dos vídeos: o resultado de uma busca permite que um vídeo seja apresentado na sua totalidade ou por cenas/*shots* específicos.
- Utilização de XQuery (Apêndice D): a recuperação das informações é feita através da aplicação da linguagem de consulta XQuery nas descrições audiovisuais;

[8] Dicionário de termos geográficos utilizados para determinar um lugar geográfico.

- Disponibilidade na Web: a VideoLib pode ser acessada na Web por meio de um *browser*;

1.3 Organização da Dissertação

O restante desta dissertação está estruturado em mais cinco capítulos, organizados da seguinte forma:

No Capítulo 2, apresentamos uma descrição dos principais trabalhos relacionados com a nossa pesquisa.

O Capítulo 3 aborda os aspectos relacionados à fase de análise e projeto da VideoLib. Os requisitos funcionais e não-funcionais, levantamento de casos de uso, diagrama de classes e arquitetura do sistema são apresentados.

O Capítulo 4 relata os detalhes de implementação. Descrevemos as tecnologias utilizadas, o processo de geração de conteúdo, o modelo de descrição audiovisual e a especificação do suporte espaço-temporal.

O Capítulo 5 apresenta exemplos práticos de utilização da VideoLib, com a formulação comentada de consultas e exibição de seus respectivos resultados.

Finalizando, no Capítulo 6, descrevemos as conclusões, principais contribuições e propostas para trabalhos futuros, introduzindo possíveis extensões e alterações da aplicação desenvolvida.

Um maior detalhamento sobre os principais assuntos tecnológicos relacionados ao nosso trabalho está presente nos apêndices.

No apêndice A, discorremos sobre as ferramentas de anotação de vídeo que utilizam metadados MPEG-7 para representar suas informações.

O apêndice B apresenta o esquema de descrição audiovisual planejado pela VideoLib.

O apêndice C aborda o padrão Dublin Core e a semântica de seus 15 elementos de metadados.

O apêndice D expõe uma visão geral sobre a linguagem de consulta XQuery e seus tipos de expressões.

O apêndice E apresenta as principais características do padrão de metadados MPEG-7.

Capítulo 2

Trabalhos relacionados

Neste capítulo, apresentamos os principais trabalhos relacionados com a nossa pesquisa. Já existe um bom número de estudos que envolvem a adoção da conformidade MPEG-7, mas poucos são os trabalhos que lidam com o desenvolvimento de sistemas de recuperação de vídeo através de descrições MPEG-7 e Dublin Core.

Alguns projetos que lidam com ferramentas de busca e bibliotecas digitais de vídeo são apresentados. Para cada trabalho relacionado, delineamos seus pontos principais e suas limitações.

2.1 A Distributed MPEG-7 Video Search

O padrão MPEG-7 fornece um conjunto de ferramentas de descrição que permite descrever um vídeo em termos de sua estrutura (segmentação), características de armazenamento (formato da mídia, codificação), criação e produção (título, data, diretor), conteúdo semântico, além de permitir uma eficiente indexação. Para obter mais informações, consultar o apêndice E.

Setten & Oltmans (SETTEN; OLTMANS, 2001) desenvolveram uma aplicação distribuída de busca audiovisual, voltada para o domínio educacional, que recupera vídeos armazenados em um banco de dados. A proposta envolve as tarefas de produção de conteúdo, que resulta no armazenamento de vídeos e metadados MPEG-7 no banco de dados; e distribuição do conteúdo, a parte mais visível para o usuário final (estudantes e professores) e tida como o principal foco do trabalho.

A produção de conteúdo envolve várias etapas de análise, incluindo transcrição de fala para texto, segmentação do vídeo em *shots*, adição de anotações manuais e extração de *keyframes*[9]. Todo processamento é executado paralelamente, com tarefas distribuídas por diversas companhias em diferentes localizações. O resultado deste processamento distribuído é unido e transposto para uma codificação MPEG-7 mestre.

[9] Imagem representativa de um segmento de vídeo, expressado por uma cena ou *shot*.

As principais funcionalidades providas pelo sistema são: Busca por vídeo com base em informações Dublin Core (DC, 2003), transcrições de áudio e anotações manuais armazenadas em MPEG-7; apresentação da lista de resultados contendo informações Dublin Core e um *keyframe*; visualização dos vídeos constituintes da lista de resultados.

Apesar das funcionalidades descritas, não foi possível realizar uma avaliação sobre o esquema de representação e recuperação de conteúdo, pois este trabalho não expressa detalhes de implementação nem da forma como o usuário pode formular consultas para obter acesso ao conteúdo audiovisual de seu interesse. Além disso, a aplicação não provê suporte à temporalidade dos dados em seu processo de busca. Projetada para ser executada em um *browser* na Internet como uma aplicação ActiveX[10], torna-se necessário que o usuário tenha uma boa conexão (acima de 2 Mbits, segundo informações da própria fonte) para que a demonstração possa ser vista *on-line*, o que não torna a aplicação disponível para a maioria dos usuários.

2.2 Indexing and Retrieval of TV News Programs Based on MPEG-7

A indexação e recuperação de vídeos contendo noticiários televisivos foram, por muito tempo, uma prática manual dominada apenas por especialistas da área. Em Fatemi & Khaled (FATEMI; KHALED, 2001), um sistema de recuperação foi projetado para realizar pesquisas em descrições MPEG-7 que modelam o conteúdo de programas de notícias de televisão, levando em conta a estrutura de composição das notícias durante a fase de consulta e navegação.

O modelo de descrição implementado contém anotações oriundas das seguintes fases:

- Produção: scripts (texto lido pelo apresentador), legendas de teletexto, cobertura gráfica (pessoas envolvidas, lugares) e estrutura hierárquica do programa de notícias, como mostra a figura 1.

[10] ActiveX é uma tecnologia Microsoft que proporciona recursos adicionais de interatividade quando uma página web é acessada através de um *browser*.

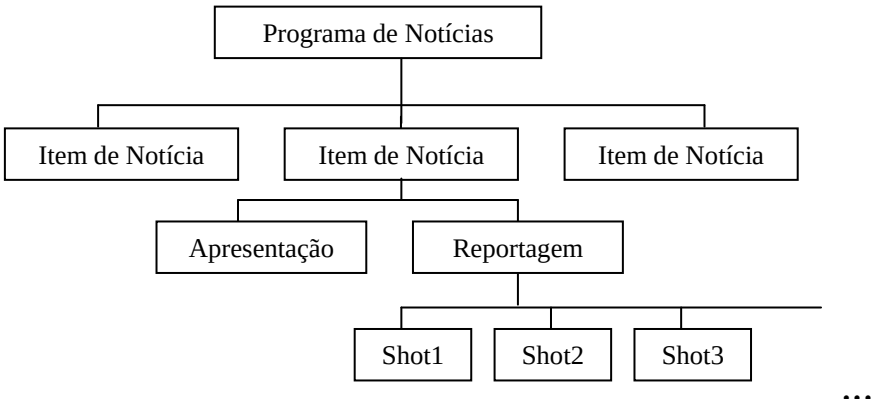

Figura 1. Estrutura hierárquica da modelagem de notícias de TV
Fonte: Fatemi e Khaled (2001).

- Arquivamento: composto por anotações feitas nos *shots* ou grupos de *shots*, consistindo de uma descrição textual do conteúdo audiovisual, juntamente com um conjunto de palavras-chave para indicar pessoas, lugares e conceitos encontrados no vídeo. Outro tipo de anotação é realizada no programa de notícias (figura 1) e nos itens de notícias, com a adição de um texto livre para o evento que está acontecendo;
- Recuperação: durante esta fase, usuários poderiam adicionar *bookmarks*[11] e anotações aos segmentos de vídeo retornados.

Para dar suporte às características necessitadas por este tipo de aplicação, os autores propuseram um modelo de notícias baseado nos *Descriptors* (Ds) e *Description Schemes* (DSs) do MPEG-7, fazendo uso das ferramentas consideradas mais importantes para serem utilizadas no projeto. Todas as descrições audiovisuais são armazenadas em um banco de dados XML nativo. No lado cliente da aplicação, duas interfaces são disponibilizadas: uma para indexação – detecção automática de *shots*, extração de *keyframe* e produção de descrições do conteúdo do vídeo – e outra para recuperação, fornecendo módulos para consulta e *browsing*. O modelo de recuperação implementado utiliza XQL (*XML Query Language*) (XQL, 1999) como linguagem de consulta aos dados.

Por ser uma aplicação que lida com notícias, o volume de informações tende a crescer consideravelmente, de maneira que torne necessário implementar soluções que possibilitem recuperar informações ao longo do tempo, ou relacionados a um local de acontecimento dos

[11] Marcador de endereços favoritos do usuário, armazenados em um arquivo.

fatos. Desta forma, uma das limitações deste trabalho é a carência de consultas envolvendo as dimensões espacial e temporal.

2.3 TV Anytime as an Application Scenario for MPEG-7

Os elementos que MPEG-7 padroniza fornecem suporte para o desenvolvimento de uma ampla faixa de aplicações. Em Pfeiffer e Srinivasan (PFEIFFER; SRINIVASAN, 2000) é demonstrado como o esquema de metadados MPEG-7 pode suprir as exigências requeridas em um cenário de aplicação TV Anytime (TV-ANYTIME, 2003), um fórum internacional que visa padronizar a TV Interativa e incorporar serviços tais como filmes em demanda, home shopping, home banking e educação remota.

O papel dos metadados dentro do TV Anytime se justifica pelas seguintes situações:

- Permite a seleção de conteúdo;
- Captura os itens constituintes para visualização posterior ou processamento;
- Possibilita a navegação do conteúdo remotamente;
- Disponibiliza acesso aos itens do conteúdo.

Como o padrão MPEG-7 fornece uma biblioteca de metadados destinados à descrição genérica de áudio e vídeo, TV Anytime se enquadra perfeitamente em um cenário de aplicação que se aproveita da utilização das ferramentas MPEG-7 para atender suas necessidades (metadados, referência de conteúdo e administração de direitos), sem ter que reinventar um novo conjunto de metadados.

TV Anytime requer a definição de um esquema de descrição específico para discriminar quais componentes serão utilizados para descrever o conteúdo audiovisual, fazendo uso de algumas ferramentas de descrição MPEG-7 consideradas essenciais para formar seu próprio esquema. Módulos independentes realizam a tarefa de validação das instâncias descritivas e se encarregam de possibilitar acesso e busca ao conteúdo audiovisual, de acordo com as informações fornecidas por seus metadados.

Embora este trabalho enfoque alguns conceitos relacionados com a anotação e recuperação de conteúdo audiovisual, selecione *Description Schemes* MPEG-7 para compor um modelo de descrição proprietário e destaque a importância de se fornecer acesso e busca ao conteúdo multimídia, não foi implementado um protótipo para demonstrar na prática o uso das idéias propostas. O escopo da pesquisa está voltado para a demonstração de como uma

coleção de componentes MPEG-7 pode ser utilizada dentro de um cenário de aplicação TV Anytime. Entretanto, serviços e aplicações específicas para TV Anytime devem ser criadas para fazer uso do modelo de descrição proposto.

2.4 Considerações finais

A forma como os elementos MPEG-7 serão utilizados para responder às consultas do usuário está fora do escopo do padrão, ou seja, a maneira como o conteúdo deverá ser examinado não será sempre a mesma; por exemplo, um vídeo poderá ser consultado e filtrado usando palavras, material visual, etc. É de responsabilidade do sistema de recuperação combinar os dados da consulta com os metadados presentes da descrição MPEG-7.

A VideoLib se assemelha com os trabalhos previamente mencionados por se tratar de uma aplicação que realiza pesquisas em metadados MPEG-7. O principal diferencial deste trabalho fundamenta-se na sua capacidade de poder realizar consultas espaço-temporais baseada em metadados MPEG-7 e Dublin Core. No que diz respeito à dimensão temporal, o resultado de uma busca pode ser filtrado realizando-se operações relacionais com datas de acordo com o operador temporal utilizado. As consultas espaciais podem ser efetivadas com o auxílio de um *Gazetteer* implementado pela VideoLib, e através da execução de operações espaciais (*Inside, Outside, Overlaps*) sob o metadado correspondente à localização geográfica de cada descrição audiovisual, tarefa esta que torna-se possível devido a assistência de um SGBD com suporte espacial.

Para auxiliar no processo de busca, a VideoLib disponibiliza uma interface gráfica que permite combinar informações Dublin Core e MPEG-7 na etapa de formulação da consulta. O resultado do processamento da consulta permite a visualização do conteúdo dos vídeos retornados, seja na sua totalidade ou correspondente apenas ao trecho de interesse do usuário. O uso de metadados Dublin Core no esquema de descrição da VideoLib (Apêndice B) tem por objetivo facilitar a aplicação de uma linguagem de consulta XML, visto que seus elementos correspondentes no padrão MPEG-7 são embutidos em um baixo nível hierárquico, implicando em uma maior complexidade em termos de recuperação. Maiores informações sobre o modelo de descrição desenvolvido, que compreende o uso conjunto de metadados Dublin Core e MPEG-7, são encontradas no capítulo 4, seção 4.1, subseção 4.1.3.

Capítulo 3

Análise e Projeto

No capítulo 2 fizemos um levantamento sobre os principais trabalhos relacionados com a nossa pesquisa. Neste capítulo, apresentamos as circunstâncias que serviram de fundamento para a construção da VideoLib, uma biblioteca digital de vídeos que realiza a recuperação de conteúdo audiovisual utilizando metadados MPEG-7 e Dublin Core. Como visto no capítulo anterior, os trabalhos que lidam com recuperação de informações por meio de anotações MPEG-7, não disponibilizam um suporte espaço-temporal. Diante desta necessidade, descrevemos nessa parte da dissertação, os aspectos de análise e projeto da VideoLib, visando propor um modelo capaz de suprir essa lacuna.

Na seção 3.1, abordamos o levantamento dos requisitos funcionais e não funcionais que devem ser atendidos pelo sistema. Na seção 3.2, apresentamos o diagrama de casos de uso para o processo de busca de conteúdo multimídia. Na seção 3.3, elaboramos o diagrama de classes do sistema, contendo a definição do papel de cada classe, seus principais métodos e atributos. Na seção 3.4, detalhamos a arquitetura do sistema.

3.1 Análise da VideoLib

Para criar o software de uma aplicação, é necessária uma descrição do problema e de seus requisitos, ou seja, determinar o objeto de discussão e o que o sistema deve fazer (LARMAN, 2000). Nesta seção, apresentaremos os requisitos funcionais e não funcionais que a VideoLib deve atender.

3.1.1 Requisitos Funcionais

Os requisitos funcionais identificam as funcionalidades que devem ser atendidas pelo sistema, determinando “o que” deve ser tomado em consideração. A seguir, apresentamos nove requisitos da VideoLib:

RF1 – Disponibilizar o sistema de consultas na Web

Com o constante crescimento da Web, e pela carência de sistemas que recuperem conteúdo multimídia, existe a necessidade de tornar o sistema disponível o maior tempo possível. Desta forma, qualquer usuário conectado à Internet pode acessar o sistema e realizar suas consultas a qualquer momento, usando um *browser*[12].

RF2 – Permitir a elaboração de diferentes consultas

Uma produção audiovisual recebe anotações sobre diferentes tipos de informações, tais como dados referentes ao conteúdo bibliográfico (data de produção, criador, título, descrição, etc.) e sobre trechos resultantes do processo de segmentação de vídeo, necessárias para implementar o mecanismo de indexação.

Diante das informações que podem ser anexadas a um vídeo, existe a importância de prover diferentes formas de elaboração de consultas, realizando combinações entre os tipos de informações disponíveis para recuperar um conteúdo de interesse.

Na VideoLib, as consultas serão submetidas na forma direta, através do fornecimento de palavras-chave no termo de pesquisa. Portanto, o sistema deve permitir que o usuário selecione os elementos correspondentes às suas necessidades, juntamente com seus respectivos valores, e submeta para processamento.

RF3 – Disponibilizar o resultado da busca

O resultado da submissão de uma consulta será apresentado ao usuário através de uma lista com informações de produção de cada vídeo selecionado, o resumo do conteúdo audiovisual e um *link* para visualização da mídia.

[12] Software utilizado para navegação e exibição de páginas web.

RF4 – Visualização do conteúdo audiovisual

Como a VideoLib manipula conteúdo audiovisual, os itens provenientes da lista de resultados serão visualizados através de um *player*[13]. A exibição do vídeo poderá ser feita de duas maneiras: na forma completa, tendo acesso a todo o conteúdo da mídia, ou apenas o trecho correspondente ao segmento de vídeo que atende os interesses do usuário.

RF5 – Expandir o resultado de uma pesquisa

A segmentação de um vídeo produz uma estrutura hierárquica em forma de árvore composta por cenas e *shots.* Uma consulta pode gerar resultados em que apenas o mais alto nível hierárquico seja disponibilizado. Assim, cada vídeo retornado deve prover a funcionalidade de expandir seu resultado a fim de encontrar informações mais específicas sobre seu conteúdo.

O sistema deve permitir a expansão da estrutura de um vídeo selecionado, para que se tenha acesso às cenas e *shots* que o constituem.

RF6 – Tornar disponível a utilização de operadores espaciais e temporais

Um vídeo pode conter informações relacionadas a um lugar geográfico ou data de criação/divulgação. A partir do momento que o sistema manipula estes tipos de informações, evidencia-se a importância de implementar mecanismos que forneçam o suporte de operações sobre tais dados.

O sistema deve ser capaz de permitir a realização de operações sobre os dados espaciais, utilizando operadores que explorem as características espaciais de um determinado objeto. Da mesma forma, diferentes operações temporais devem ser aplicadas em intervalos de tempo, definidos por um período compreendido entre duas datas. Tais funcionalidades permitem aos usuários o refinamento dos dados que são retornados em uma consulta.

[13] Componente de software responsável pela exibição de conteúdo audiovisual.

RF7 – Permitir a diversificação de conteúdo

A aplicação deverá ser capaz de trabalhar com recuperação audiovisual de qualquer categoria de vídeo (shows musicais, comerciais de TV, telejornalismo, etc.), sem que para isso seja causado qualquer impacto na arquitetura da aplicação.

RF8 – Implementar o Gazetteer

A referência espacial deve ser feita através do nome de um lugar geográfico. As informações sobre os lugares geográficos devem estar organizadas em forma de árvore seguindo uma hierarquia geográfica, tais como: continentes, países e estados. Esta organização facilita o entendimento e a navegação entre as localizações geográficas.

O sistema deve possuir um esquema capaz de estruturar, manter e fornecer acesso ao conteúdo geográfico manipulado pela aplicação, da mesma forma que deve ser flexível para suportar a adição de novos locais geográficos ou realizar possíveis mudanças na árvore.

RF9– Indexação e manutenção da Biblioteca Digital

Qualquer vídeo que venha a fazer parte da biblioteca digital deve passar pelo processo de indexação, atividade esta que consiste na decomposição estrutural da mídia em cenas e *shots.* Juntamente com esta tarefa, são realizadas anotações específicas sobre cada intervalo temporal que uma cena ou *shot* representa.

A VideoLib utilizará de recursos que facilitem a realização da indexação de vídeo na composição de suas descrições audiovisuais, fornecendo o suporte necessário para a efetivação de inserção, remoção e alteração do conteúdo visando tornar mais prático o trabalho de manutenção.

3.1.2 Requisitos Não-Funcionais

Os requisitos não-funcionais, um conjunto de atributos considerados essenciais para um bom desenvolvimento/desempenho do sistema, estão em destaque nas linhas que seguem:

RNF1 – Tipo de interface desejada

O sistema será construído para utilização na Web, permitindo seu acesso por meio de qualquer computador que esteja conectado à Internet. Assim, sua interface deve ser apresentada por meio de um *browser*.

RNF2 – O sistema deve ser de fácil manutenção

O sistema deverá planejar uma arquitetura flexível que necessite de poucas atualizações quando forem necessárias mudanças nas regras que regem a aplicação. Para que não haja acoplamento direto dos objetos da lógica da aplicação com a interface do usuário, deve-se utilizar o padrão MVC (*Model-View-Controller)* (GAMMA *et al,* 2000). A seção 4.1, do capítulo 4, fornece maiores informações sobre este padrão.

RNF3 – A interface deve ser de fácil utilização

Na submissão de consultas, não deverá ser exigido conhecimento sobre qualquer linguagem de acesso aos dados, seja ela SQL (*Structured Query Language*) ou XQuery. Deste modo, a interface com o usuário deverá abstrair detalhes internos[14] de implementação – no que se refere ao uso destas linguagens – e proporcionar um fácil manuseio, com entrada de dados facilitada pelo teclado e mouse, de forma a simplificar a interação do usuário com o sistema.

RNF4 – Usuários do sistema

Usuários conectados à Internet poderão utilizar as funcionalidades do sistema. Usuários leigos ou experientes poderão acessar a interface e selecionar os itens de interesse para a formulação de suas consultas, pois os campos que constituem as entradas de dados são de fácil entendimento.

[14] Detalhes internos que se refiram à forma de elaboração de instruções para execução de consultas.

RNF5 – Independência de Plataforma

Com a diversificação de plataformas operacionais existentes, torna-se conveniente desenvolver aplicações que sejam transportáveis, ou seja, que possam ser executadas em diferentes computadores (independente de sua arquitetura e sistema operacional) sem que haja necessidade de reescrever programas. Por utilizar a linguagem de programação Java, como base para a implementação da lógica do negócio, esta transportabilidade é viável devido a disponibilização do software para diversas plataformas.

3.2 Diagrama de Casos de Uso

A modelagem do diagrama de casos de uso é uma técnica utilizada para melhorar a compreensão dos requisitos de um software (LARMAN, 2000). Um caso de uso é um documento narrativo que descreve a seqüência típica de eventos de um ator[15] usando um sistema para completar um processo (JACOBSON *et al,* 1992).

O modelo de casos de uso é representado por um diagrama composto por atores, casos de uso e o relacionamento entre eles, tendo a finalidade de oferecer uma rápida compreensão acerca da interação entre seus elementos constituintes, combinado com a descrição de cada um deles.

Um método usado para identificar os casos de uso é baseado em atores. Nesse método, são identificados os atores relacionados ao sistema. Para cada ator, são identificados os processos que eles iniciam ou dos quais eles participam. Partindo desta idéia, apresentamos na figura 2, o diagrama de caso de uso para o processo de submissão de uma consulta na VideoLib. Neste diagrama, os atores são representados pelo usuário do sistema e Anotador.

[15] Ator é qualquer entidade externa ao sistema que interage com o mesmo. Um ator estimula o sistema com eventos de entrada ou recebe algum resultado de processamento do mesmo (LARMAN, 2000).

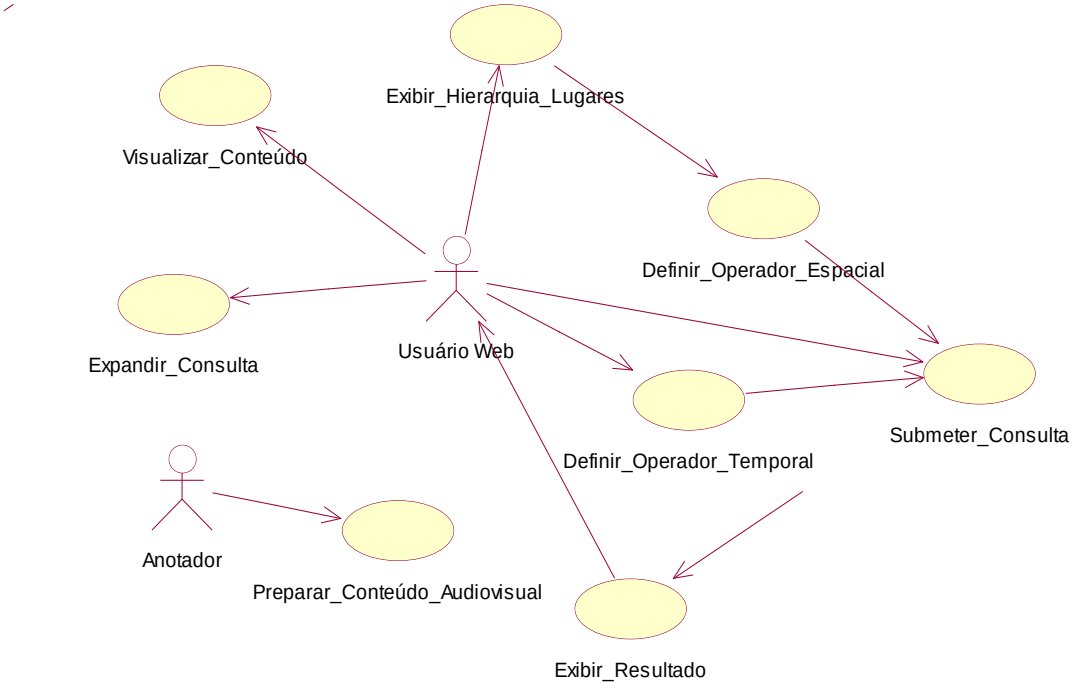

Figura 2. Diagrama de Caso de Uso

De acordo com a figura 2, observamos que o ator (Usuário Web) estimula o sistema com eventos de entrada ou recebendo algo do mesmo. Esse processo envolve a interação do usuário com os casos de uso.

A seguir, apresentamos a descrição dos casos de uso no formato alto nível[16] (para todos os casos de uso) e na forma expandida, que segundo Larman (2000, p. 70), fornece um melhor detalhamento das funcionalidades do processo.

Caso de Uso : **Definir_Operador_Temporal**
Ator(es) : Usuário Web
Descrição : Para consultas que envolvem a dimensão temporal, o usuário deve especificar o período de tempo desejado e selecionar o operador temporal de interesse.
Referência : RF6

[16] Descrição de um processo de forma breve, usualmente, em duas ou três sentenças.

Caso de Uso : **Exibir_Hierarquia_Lugares**

Ator(es) : Usuário Web

Descrição : Para consultas que envolvem a dimensão espacial, o usuário navega em uma árvore composta por uma hierarquia de lugares geográficos.

Referência : RF1, RF2 e RF8

Seqüência Típica de Eventos

Ação do ator	Resposta do Sistema
1. Este caso de uso começa quando o usuário necessita obter a relação de localizações geográficas disponíveis. Assim, em outra janela, o usuário navega em uma árvore hierarquicamente composta de localizações geográficas e seleciona o item de interesse.	
	2. Após a confirmação da localização geográfica selecionada, o valor correspondente é transportado para a interface principal.

Caso de Uso : **Definir_Operador_Espacial**

Ator(es) : Usuário Web

Descrição : Com a definição do lugar geográfico a ser pesquisado, o usuário seleciona o operador espacial que será utilizado na consulta.

Referência : RF6

Caso de Uso : **Submeter_Consulta**

Ator(es) : Usuário Web

Descrição : O usuário fornece os valores nas caixas de texto referentes às informações de interesse e submete a consulta.

Referência : RF1, RF2 e RF6

Seqüência Típica de Eventos

Ação do ator	Resposta do Sistema
1. Este caso de uso começa quando o usuário acessa o sistema via *Web* à procura por vídeos ou cenas de vídeo de seu interesse. O usuário digita a palavra-chave, simples ou composta, na caixa de texto correspondente ao termo a ser pesquisado. Cada caixa de texto corresponde a um termo e possui semântica distinta com relação aos outros elementos. Mais de uma caixa de texto pode receber palavra-chave. Para utilizar o suporte temporal, o usuário deve selecionar o operador temporal de interesse (*During*, *After* ou *Before*) e fornecer o período de tempo através de datas. Estas atividades correspondem às ações ligadas ao caso de uso ”Definir_Operador_Temporal”. Caso a consulta faça uso do suporte espacial, relacionado com o caso de uso “Definir_Operador_Espacial”, é necessário que seja fornecida a localização geográfica de referência, e o tipo de operação espacial desejado (*Inside, Overlaps* e *Outside*).	
2. Usuário submete a consulta.	3. Verifica quais elementos receberam valores na interface e transforma o resultado da interação em uma instrução a ser processada pelo sistema.
	4. Executa a instrução de consulta e seleciona os itens de interesse.
	5. Aciona “Exibir_Resultado”

Caso de Uso : **Exibir_Resultado**

Ator(es) : Sistema

Descrição : Após o processamento da consulta, o resultado final com os itens selecionados é apresentado ao usuário.

Referência : RF3

Caso de Uso : **Visualizar_Conteúdo**

Ator(es) : Usuário Web

Descrição : Com a apresentação do resultado da consulta, o usuário tem a possibilidade de visualizar os vídeos retornados, seja na sua totalidade ou em trechos específicos.

Referência : RF4

Caso de Uso : **Expandir_Consulta**

Ator(es) : Usuário Web

Descrição : Os vídeos retornados, pelo processo de submissão da consulta, podem ser expandidos em nível de cenas ou *shots.*

Referência : RF5

Seqüência Típica de Eventos

Ação do ator	**Resposta do Sistema**
1. Este caso de uso começa quando o resultado da consulta é exibido ao usuário e o mesmo deseja expandir o resultado para o nível de cenas ou *shots.*	
2. Usuário deseja obter a relação de cenas ou *shots* relacionados ao item escolhido.	3. Obtém o identificador do item de resultado escolhido e seleciona todas as cenas ou *shots* que o constituem
	4. Um novo resultado é gerado e apresentado ao usuário, com a lista de cenas ou *shots* resultantes do processamento.

Caso de Uso : **Preparar_Conteúdo_Audiovisual**

Ator(es) : Anotador[17]

Descrição : Todo vídeo pertencente à biblioteca digital deve passar por um processo de análise para que possa ser segmentado e realizado as anotações textuais sobre o seu conteúdo.

Referência : RF9

Seqüência Típica de Eventos

Ação do ator	Resposta do Sistema
1. Este caso de uso começa quando um novo vídeo é selecionado para fazer parte da biblioteca digital. O anotador utiliza ferramentas de suporte para realizar a detecção automática de *shots*, extração de *keyframe* e acréscimo de anotações manuais, iniciando, desta forma, o processo de indexação. A indexação de um vídeo consiste basicamente nos seguintes passos (IANNELLA; HUNTER, 1998): • Segmentar o vídeo hierarquicamente em cenas e *shots*. Uma cena é um intervalo temporal que representa diferentes visões de um mesmo evento. Sua composição é formada por um ou mais *shots*. Um *shot* é uma seqüência contínua de *frames* capturada por uma câmera de vídeo. Uma seqüência, que não foi definida na etapa de indexação da VideoLib por não exigir uma segmentação em três níveis, é composta por uma ou mais cenas relacionadas.	

[17] Profissional responsável pela produção das descrições audiovisuais.

Seqüência Típica de Eventos

Ação do ator	Resposta do Sistema
• Descrever a mídia, iniciando com a descrição das informações bibliográficas[18] do conteúdo audiovisual (data de produção, criador, título, etc.); • Descrever cada cena, fornecendo um identificador (*id*), um resumo textual, valores para o tempo inicial e final, ou em se tratando de frame, seus valores de início e fim; • Descrever cada *shot*, fornecendo um identificador (*id),* um resumo textual, valores inicial e final para o tempo ou *frame* e determinação do *keyframe* (imagem representativa de um segmento de vídeo, expressada por uma cena ou *shot*).	
3. O resultado da atividade de indexação é transposto para um padrão de metadados que possibilite a representação de conteúdo audiovisual.	
	4. Após a finalização da produção de conteúdo, o arquivo resultante do processo de anotação é introduzido no repositório de dados do sistema para que o mesmo seja pesquisado quando solicitações por conteúdo multimídia forem requisitadas.

[18] Informações acerca do conteúdo que está sendo descrito.

3.3 Fase de Projeto – Diagrama de Classes

Nas seções 3.1 e 3.2, fizemos um levantamento dos requisitos e definimos o diagrama de casos de uso. Esta análise enfatiza uma investigação do problema a ser resolvido, de como uma possível solução pode ser montada para satisfazer os objetivos. Nesta seção, estaremos focalizando os aspectos de projeto da VideoLib, ou seja, a solução lógica adotada para que o sistema atenda aos requisitos pré-estabelecidos.

O diagrama de classes (figura 3) fornece uma visão das classes, interfaces e seus relacionamentos como componente de software do sistema. A notação UML *(Unified Modeling Language)* (BOOCH, 1999) foi utilizada para elaborar o diagrama de classes da VideoLib.

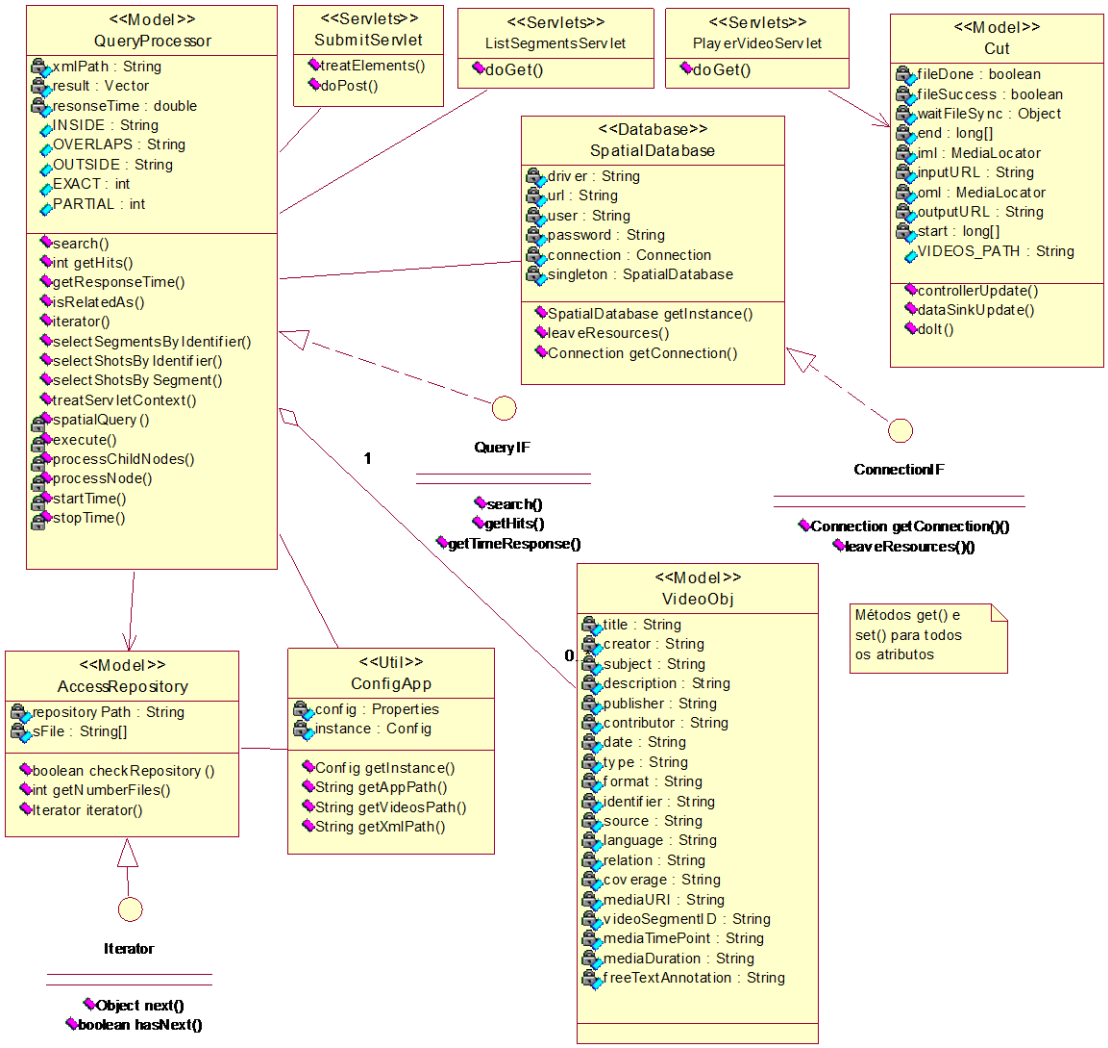

Figura 3. Diagrama de Classes da VideoLib

A seguir, descrevemos as classes do diagrama organizadas por pacote[19], fornecendo uma visão geral dos seus objetivos e dos principais métodos:

3.3.1 Pacote *Database*

- **SpatialDatabase:** Esta classe, responsável pela conexão do sistema com o banco de dados espacial, tem a função de implementar o *Gazetteer*. Sua estrutura baseia-se na implementação do *Design Pattern*[20] *Singleton* (GAMMA *et al*, 2000), que tem o objetivo de assegurar que uma classe possua uma única instância e que esta instância seja facilmente acessível. Através desta única instância, os usuários poderão requisitar conexões com o banco de dados. Os atributos pertencentes a esta classe possuem as seguintes definições:
  - driver: nome do *driver* utilizado para conexão com o SGBD;
  - url: *string* de conexão com o banco de dados;
  - user: identificação do usuário para acesso ao SGBD;
  - password: senha utilizada para validar a identificação do usuário;
  - connection: conexão estabelecida com o banco de dados;
  - singleton: instância de SpatialDatabase necessária para garantir um único objeto da classe.

  Seus métodos são:
  - getInstance(): método de classe que obtém a instância de SpatialDatabase;
  - leaveResources(): libera os recursos obtidos por uma conexão;
  - getConnection(): obtém uma conexão com o banco de dados.

- **ConnectionIF:** Esta interface disponibiliza métodos responsáveis pela obtenção de uma conexão e pela liberação dos recursos alocados. O uso desta interface permite, a cada classe que a implemente, realizar um diferente tratamento na forma de abrir e liberar recursos de uma conexão, propiciando a prática de polimorfismo.

---

[19] Pacotes são mecanismos utilizados pela UML (*Unified Modeling Language*) que tem a finalidade de ilustrar o agrupamento de elementos semanticamente interligados.

[20] São soluções genéricas para problemas que ocorrem com freqüência, de forma que tais soluções possam ser reusadas diversas vezes em situações diferentes (GAMMA *et al*, 2000).

3.3.2 Pacote *Util*

- **ConfigApp:** a VideoLib necessita de uma forma flexível de se obter acesso ou modificar alguns parâmetros de configuração do sistema, sem que para isso seja necessário modificar o código fonte da aplicação. Um *Singleton* foi utilizado para fornecer esta funcionalidade pelo motivo de melhor representar o uso de variáveis globais, e por fornecer um ponto global de acesso para as mesmas. O atributo **config** representa uma instância da classe *Properties*[21], responsável por prover acesso aos parâmetros de configuração. Os principais métodos desta classe são:
  - getAppPath(): método que retorna o *path*[22] da aplicação dentro do servidor de aplicações ***tomcat***[23] (APACHE, 2003);
  - getXmlPath(): obtém o *path* referente ao diretório no sistema de arquivos onde estão armazenadas as descrições audiovisuais da VideoLib. Esse diretório deve estar contido na mesma hierarquia do *path* da aplicação;
  - getVideosPath(): obtém o *path* referente ao diretório no sistema de arquivos onde estão armazenadas os vídeos locais da biblioteca digital. Esse diretório deve estar contido, preferencialmente, na mesma hierarquia do *path* da aplicação. Não só vídeos locais podem fazer parte da biblioteca digital, como também qualquer vídeo que esteja localizado em uma URL válida na Web. Estes ficarão susceptíveis a inconsistência no que diz respeito à localização física, caso sejam retirados do seu local de origem.

3.3.3 Pacote *Servlets*

As classes constituintes deste pacote são responsáveis pela funcionalidade do componente *Controller* no contexto do padrão MVC. A seguir, detalhamos o papel de cada classe:

- **SubmitServlet:** classe responsável pela submissão e processamento das consultas. Assim que o usuário finaliza a entrada de dados nas caixas de texto da interface e dá início à operação de busca, este *servlet* fica encarregado de selecionar apenas os

[21] Classe que armazena pares de valores, assim como a Hashtable, com uma diferença: ambos os valores (chave e valor) são Strings.

[22] Rota por onde o sistema operacional deve procurar os arquivos que serão utilizados pelo sistema.

[23] Tomcat é um container de servlet usado na implementação da referência oficial para as tecnologias *Java Servlet* e *Java Server Pages*.

campos que possuem valores associados, efetuando a preparação dos parâmetros para submeter à lógica interna. Logo depois, executa-se a consulta de acordo com os campos requisitados, e redireciona-se a saída para um arquivo JSP (resultado.jsp) responsável pela exibição dos resultados.

Este procedimento permite a separação, de maneira independente, entre a lógica da aplicação e os componentes da interface com o usuário. O método *treatElements()* apresenta um papel secundário dentro da classe. Ele realiza um tratamento das palavras-chave para que possam ser destacadas do restante do texto, quando o resultado da consulta for apresentado ao usuário.

- **ListSegmentsServlet:** Uma consulta pode gerar resultados em que apenas o mais alto nível hierárquico do vídeo seja disponibilizado, caso em que somente elementos Dublin Core são utilizados na submissão da consulta. Para que se tenha acesso às cenas ou *shots* que constituem um determinado vídeo, retornado nesta condição, é necessário que o usuário requisite por esta funcionalidade. Este *servlet* estende o resultado de um item retornado para o nível desejado, tomando como referência o identificador do vídeo e o tipo de expansão solicitado pelo usuário. Ao término do processamento, o fluxo de execução é redirecionado para uma página JSP (detalhaSegmento.jsp) encarregada pela apresentação dos resultados.

- **PlayerVideoServlet:** este *servlet* é responsável pela exibição do trecho de um vídeo no *browser*. Para que isto seja possível, são obtidos o nome do arquivo, o tempo inicial e o tempo final do intervalo de vídeo. De posse desses dados, o trecho correspondente a este intervalo é extraído do vídeo original e enviado sob a forma de bytes para o *browser*, que se encarrega de apresentar o conteúdo.

3.3.4 Pacote *Model*

Este pacote contém as classes que compõem a lógica da aplicação. A seguir, definimos o papel de cada classe:

- **Cut:** classe responsável por “recortar” um trecho de vídeo tomado como referência e gerar o resultado para um arquivo de saída, conservando a integridade física do arquivo inicial. A definição do trecho de vídeo é feita ao fornecer os instantes de

tempo inicial e final correspondentes ao segmento de interesse. Utilizamos a tecnologia *Java Media Framework* (JMF, 2003) para implementar o seu funcionamento. Apenas para efeito de observação, realizamos testes com a criação de trechos de vídeo em vários formatos de saída (AVI, MOV e MPEG), onde obtivemos a conclusão de que o programa se comporta melhor – em termos de completar sua tarefa – quando o resultado final é salvo no formato de vídeo *quicktime*[24] (extensão do arquivo: MOV).

- **QueryProcessor:** principal classe do sistema, tem o objetivo de planejar e executar as instruções de consulta na VideoLib, de acordo com os elementos requisitados na interface com o usuário. A classe **ConfigApp** fornece o local de armazenamento das descrições audiovisuais, também chamado de repositório. De posse dessa informação, um objeto da classe **AccessRepository** é instanciado para que seja possível ativar a mesma consulta em cada documento textual existente no repositório de arquivos.

  Além de possuir algumas constantes de classe para representação das operações espaciais e modo de precisão da consulta, os seguintes atributos merecem destaque: **xmlPath,** que representa a localização do repositório de descrições audiovisuais que será utilizado no processo de busca; e **result,** que armazena uma coleção de objetos **VideoObj** retornados em uma consulta.

  Os principais métodos são:

  - getHits(): função que devolve o número de ocorrências de uma consulta;
  - getResponseTime(): retorna o tempo gasto, em segundos, com o processamento de uma consulta;
  - search(): método que planeja a instrução de consulta XQuery de forma dinâmica e se encarrega de executá-la, invocando internamente o método privado execute(). Caso seja requisitado o uso do suporte espaço-temporal, as ações necessárias também serão executadas por este método.
  - processChildNode() e processNode(): uma instrução XQuery permite definir o resultado de saída – também no formato XML – quando um determinado documento satisfaz aos critérios de uma consulta. Estes métodos navegam entre os elementos retornados e recupera as informações

[24] http://www.apple.com/quicktime/

necessárias para montar a lista de resultado. Cada descrição audiovisual que estiver de acordo com os critérios da consulta terá um correspondente objeto da classe **VideoObj** como componente da lista de resultados**.**

- isRelatedAs(): este método verifica o relacionamento espacial entre dois lugares geográficos, de acordo com o operador solicitado pelo usuário (*Inside*, *Overlaps* ou *Outside*). Para que isso seja possível, deve ser estabelecida uma conexão com o SGBD que fornece o suporte espacial, e enviada a cláusula SQL correspondente à operação;
- iterator(): retorna uma coleção de objetos **VideoObj** correspondente ao resultado da consulta;
- selectSegmentsByIdentifier() e selectShotsByIdentifier(): métodos que permitem expandir o resultado de uma consulta para o nível de cena ou *shot*, respectivamente.

- **VideoObj:** representa um item de vídeo selecionado pelo processo de busca. O resultado de uma consulta pode gerar zero ou mais objetos desta classe. São usados, principalmente, para montar a lista de resultado de uma pesquisa. Contém uma série de atributos que podem ser acessados via métodos *get* e *set*[25].

- **AccessRepository:** responsável pela seleção de todos os arquivos XML que estiverem presentes em um determinado diretório a ser passado como parâmetro. O atributo **repositoryPath** corresponde ao local de armazenamento das descrições audiovisuais contidas na VideoLib. O atributo **sFile –** um vetor de *Strings* **–** contém o nome de todos os arquivos XML encontrados no repositório. Esta classe possui considerável importância porque ela é quem fornece a lista de descrições audiovisuais para o processador de consulta executar suas instruções e selecionar aquelas que atendem a uma determinada requisição. Seus métodos são:
  - iterator(): Uma implementação do *Design Pattern Iterator* (GAMMA *et al*, 2000) que tem a função de definir métodos – utilizando uma interface – que acessem seqüencialmente todos os elementos de uma coleção, sem expor sua estrutura interna. Na VideoLib, este iterador fornece acesso seqüencial às descrições audiovisuais contidas no repositório de arquivos;

[25] São métodos que só permitem a leitura e modificação do valor de um atributo por meio de sua chamada. Também são conhecidos como métodos *accessor/mutator* (ECKEL, 1998).

- getNumberFiles(): obtém o número de arquivos XML manipulados pela classe em questão;
- checkRepository(): realiza uma verificação no diretório para certificar que dentro dele só existem arquivos XML.

3.4 Arquitetura da VideoLib

A arquitetura define as partes que compõem o sistema e o relacionamento entre eles. O principal propósito da arquitetura da VideoLib é prover uma interface simples e funcional, em múltiplas camadas, que proporcione um fácil acesso às informações audiovisuais. Uma característica importante é que essa arquitetura deve dispor, aos usuários, a possibilidade de acessar o sistema em diferentes locais no mundo.

Para disponibilizar a aplicação na Internet, a arquitetura é dividida em camadas. Um modelo habitualmente utilizado, que separa a lógica da aplicação em uma camada intermediária, das camadas de apresentação (interface com o usuário) e de armazenamento persistente dos dados, é conhecido como arquitetura em três camadas (LARMAN, 2000). A camada de apresentação é relativamente livre de processamento ligado à aplicação, ficando geralmente encarregada de repassar solicitações de tarefas para uma camada intermediária ou receber o resultado de um processamento. A camada intermediária (lógica da aplicação ou camada de aplicação) se comunica com a camada de armazenamento, internamente, de forma transparente. A camada de armazenamento (camada de banco de dados) constitui o mecanismo de armazenamento persistente adotado pela arquitetura.

O modelo de três-camadas apresenta algumas vantagens, tais como: utilização de um *browser* como cliente universal; evita instalação em computadores de clientes; e permite a separação de responsabilidades entre as camadas de Apresentação e Intermediária, possibilitando a reutilização da lógica em outros sistemas (EDWARDS, 1999).

De acordo com Larman (2000, p. 270), "quando a arquitetura possui uma decomposição mais fina, podemos dispensar a denominação '*arquitetura em três camadas*' e, em seu lugar, falar de '*arquiteturas em múltiplas camadas*'". Uma arquitetura é dita em *múltiplas camadas* (quatro ou mais) quando inclui a separação de responsabilidades, de forma implícita, na arquitetura clássica de três camadas, decompondo a camada intermediária (lógica da aplicação) em outras camadas, freqüentemente feita separando as responsabilidades

relacionadas ao domínio do problema dos aspectos de serviços (interação com o banco de dados, segurança, etc.). Este tipo de arquitetura proporciona escalabilidade, propriedade esta que facilita a adição de novas camadas e funcionalidades sem que haja grandes impactos na estrutura da aplicação que está sendo desenvolvida.

A VideoLib possui arquitetura em 4 camadas (multicamadas). Neste caso, a lógica da aplicação foi decomposta em duas camadas: a Camada de Domínio e a Camada de Serviços. A Camada de Domínio é composta pelos pacotes Model, Servlets e Util. A Camada de Serviços é constituída pelo pacote Database, que fornece acesso aos dados armazenados.

Utilizando a abordagem de pacotes[26], a figura 4 apresenta a arquitetura da VideoLib.

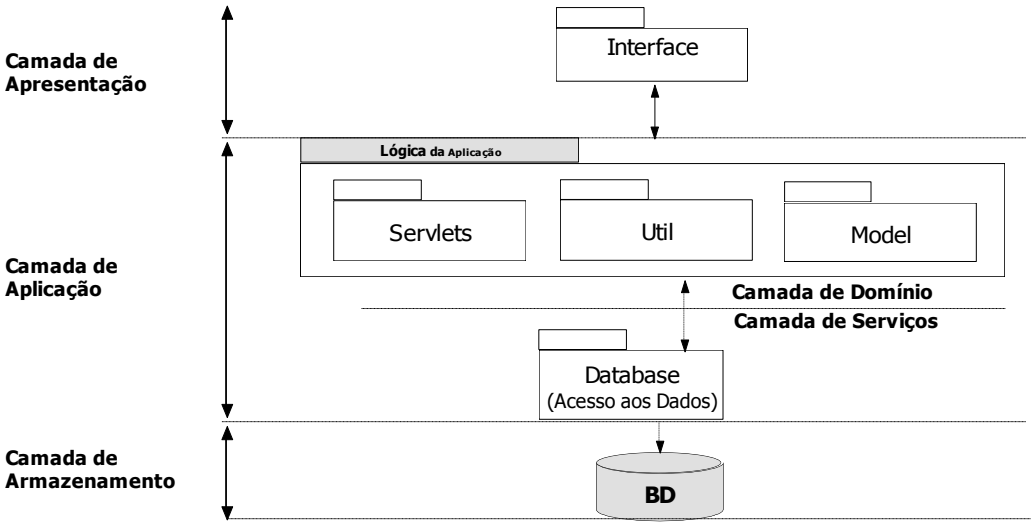

Figura 4. Arquitetura da VideoLib estruturada em pacotes

A Camada de Armazenamento corresponde ao banco de dados objeto-relacional *PostgreSQL* (POSTGRESQL, 2003), responsável pelo suporte da característica espacial da VideoLib, e pelo conjunto de descrições audiovisuais MPEG-7/Dublin Core, em formato XML, armazenados em sistema de arquivos na máquina servidora da aplicação.

A Camada de Serviços é constituída pelo pacote Database, responsável pela comunicação da aplicação com o SGBD e descrições audiovisuais localizadas no sistema de arquivos.

[26] Pacotes são mecanismos utilizados pela UML (*Unified Modeling Language*) que tem a finalidade de ilustrar o agrupamento de elementos semanticamente interligados.

A Camada de Domínio é formada por três pacotes: o pacote Servlets, constituído pelas classes que recebem as informações da interface, preparam os dados, ativam a lógica da aplicação e encaminham o resultado do processamento para um arquivo JSP (*Java Server Page*); o pacote Util, responsável pelo acesso aos parâmetros de configuração da aplicação; e por fim, o pacote Model, composto pelas classes que representam os conceitos do domínio e que modelam o processo de interação com o usuário.

Esta mesma arquitetura implementa os conceitos relacionados ao padrão de projeto MVC (*Model-View-Controller*) (GAMMA *et al*, 2000). Este padrão tem o objetivo de tentar diminuir ao máximo o acoplamento direto dos componentes da API com os objetos da GUI (*Graphical User Interface*), proporcionando a reutilização da lógica em novas aplicações e minimizando o impacto das mudanças de requisitos de interface sobre a camada de domínio. A utilização do padrão MVC permite definir a separação, de maneira independente, do *Model* (Modelo) – correspondente à lógica da aplicação – da *View* (Visualização), que corresponde à camada de interface com o usuário. O *Controller* (Controle) define a maneira como a interface do usuário reage às entradas da mesma, ficando responsável pelo controle do fluxo da aplicação. A figura 5 apresenta a arquitetura da VideoLib sob esse ponto de vista.

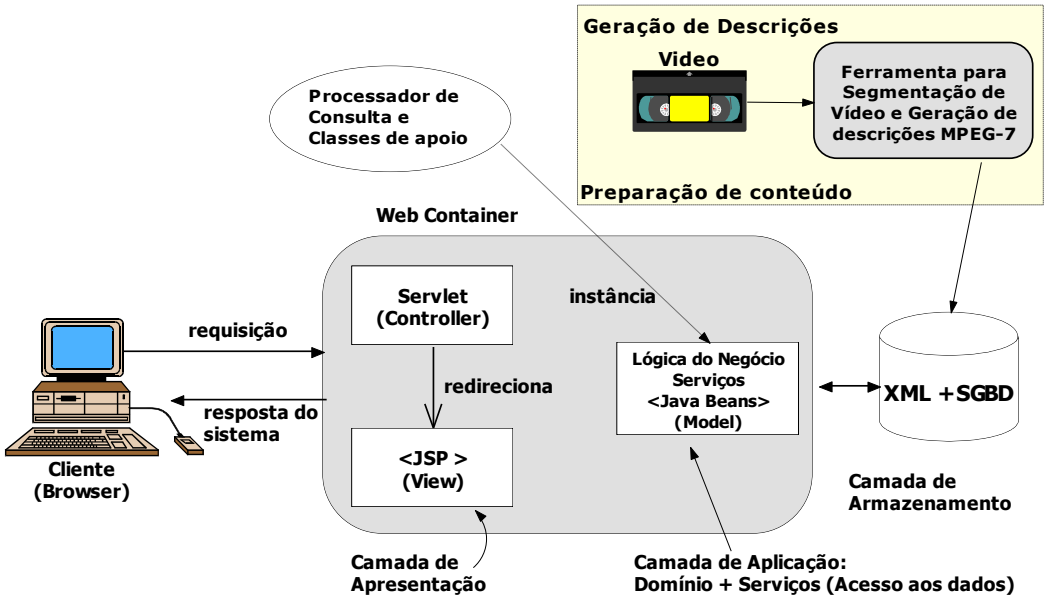

Figura 5. Arquitetura da VideoLib sob o ponto de vista MVC

Em aplicações Web, assim como na VideoLib, o navegador (*browser*) gera uma solicitação que é atendida pelo *Controller* (Servlets especializados contidos no pacote *Servlets*). Dependendo da requisição, o Servlet responsável se encarrega de obter as

informações necessárias, preparar parâmetros e executar as devidas ações, instanciando objetos da lógica da aplicação para completar a tarefa. Ao término do processamento, o *Controller* encaminha a exibição da interface de retorno para um arquivo JSP.

Conforme visto na figura 5, as quatro camadas da VideoLib estão dispostas da seguinte forma:

- Cliente: software para navegação na Web (*browser*);
- Apresentação: páginas para exibição de conteúdo;
- Aplicação: lógica do negócio e serviços;
- Armazenamento: mecanismo de armazenamento persistente (sistema de arquivo e SGBD).

3.5 Considerações Finais

Este capítulo apresentou os aspectos relacionados à fase de análise e projeto da VideoLib. Relacionamos os requisitos que devem ser atendidos pelo sistema; ressaltamos os atributos considerados essenciais para o desenvolvimento da aplicação; especificamos os casos de uso no formato de alto nível e na forma expandida; projetamos a solução lógica para atender aos requisitos; e finalmente, definimos a arquitetura do sistema, explicitando os benefícios obtidos com a utilização do padrão MVC. Utilizamos a notação UML para realizar a modelagem do sistema, usando conceitos orientados a objetos e aplicação de padrões de projeto (*Design Patterns*).

Capítulo 4

Implementação

No Capítulo 3, especificamos as atividades correspondentes à fase de análise e projeto do sistema. Os requisitos, casos de uso, diagrama de classes e arquitetura foram descritos a fim de fornecer uma visão geral sobre as principais características da VideoLib. Neste capítulo, delineamos os passos que envolvem a preparação das descrições audiovisuais e os detalhes de implementação.

A estrutura deste capítulo está organizada da seguinte forma: na seção 4.1, abordamos as particularidades empregadas no processo de implementação. A seção 4.2 relata a metodologia utilizada para fornecer o suporte espaço-temporal.

4.1 Implementação da VideoLib

Esta seção abrange os principais aspectos relacionados à implementação do sistema. Na subseção 4.1.1, discorremos sobre as tecnologias empregadas. A subseção 4.1.2 descreve a etapa de preparação de conteúdo. Finalmente, na subseção 4.1.3, especificamos o modelo de descrição audiovisual planejado para a VideoLib.

4.1.1 Tecnologias Utilizadas

A VideoLib foi desenvolvida como uma aplicação para ser acessada via Web, onde o acesso às informações é efetivado por meio de um *browser*. Esta funcionalidade torna desnecessária a instalação de qualquer software adicional na máquina do usuário, bastando apenas que ele possua um computador com acesso à Internet.

A seguir, apresentamos as principais tecnologias empregadas no desenvolvimento da VideoLib:

- **Java** (GOSLING *et al*, 2000): Java é uma linguagem de programação orientada a objeto, desenvolvida pela *Sun Microsystem*[27]. Atualmente, Java é considerada uma poderosa ferramenta para o desenvolvimento de projetos voltados para a Internet. Suas principais características são: independência de plataforma, pois seu código pode ser executado em qualquer sistema de computador sem que seja necessário modificar uma única linha, bastando apenas que haja um interpretador Java instalado; portabilidade (*byte-code* é portável entre sistemas operacionais); suporte à construção de sistemas distribuídos e acessibilidade para realização de conexões com banco de dados (DEITEL; DEITEL, 2001a). Para compilação e interpretação da linguagem Java, foi usado o JDK[28] (*Java Development Kit*) 1.4.0. A VideoLib utilizou Java para desenvolver a lógica da aplicação, o serviço de acesso aos dados e Servlets.
- **Servlets**: Servlets são programas Java compilados, executados em um servidor Web[29], que atuam como camada intermediária entre um pedido de requisição de um *browser* (ou outro cliente HTTP) e o banco de dados ou aplicações residentes no servidor Web (HALL, 2000), podendo produzir na maioria das vezes, páginas HTML dinâmicas como retorno. Na VideoLib, Servlets foram implementados para modelar o padrão MVC, fazendo o papel do *Controller.*
- **JSP**: JSP (*Java Server Pages*) é uma tecnologia para o desenvolvimento de aplicações Web que utiliza código Java embutido em HTML para produzir conteúdo dinâmico (HALL, 2000). Com JSP podemos realizar alterações e verificar as mudanças, sem ter que reiniciar a aplicação no servidor Web. Arquivos JSP são utilizados na VideoLib para exibir o resultado das consultas e gerar a hierarquia de localizações geográficas utilizadas pelo *Gazetteer*.
- **XML** (BRAY *et al*, 2000): XML é uma linguagem de marcação de dados padronizada pelo W3C[30] (*Wide World Web Consortium*), que fornece um formato para a descrição de dados estruturados e semi-estruturados, de forma que qualquer

[27] http://java.sun.com

[28] http://java.sun.com/j2se/1.4/

[29] Servidor Web é um programa que está sendo executado em uma máquina ligada à Rede, esperando por conexões do mundo exterior para apresentar certos documentos solicitados por um *browser*.

[30] http://www.w3.org

software por meio de um *parser*[31] possa entender e usar seu conteúdo (DEITEL; DEITEL, 2001b). A portabilidade, flexibilidade em representar os mais diversos tipos de informações e a separação do conteúdo da apresentação são as principais características do XML (ABITEBOUL *et al*, 2000). As descrições audiovisuais da VideoLib são instanciadas no formato XML, visando fornecer suporte à representação de metadados Dublin Core e MPEG-7.

- **JMF*:*** A API (*Application Programming Interface*) do JMF[32] (*Java Media Framework)* permite que áudio, vídeo ou outro tipo de mídia baseado em tempo sejam manipulados em aplicações e *applets* que utilizam Java (JMF, 2003). Atualmente na versão 2.1.1, JMF foi utilizado na VideoLib – especificamente na implementação da classe Cut – para receber como parâmetro de entrada a localização de um arquivo de vídeo, no formato MPEG, MOV ou AVI, e gerar um arquivo de saída contendo um determinado trecho de vídeo, de acordo com o intervalo temporal especificado pelo usuário.
- **PostgreSQL:** O PostgreSQL (POSTGRESQL, 2003) é um SGBD Objeto-Relacional de código aberto[33], que pode ser instalado em diversas plataformas de hardware e sistemas operacionais. Utilizamos o PostgreSQL para a implementação do *Gazetteer*, pois este SGBD disponibiliza suporte espacial e consiste em uma solução de âmbito público. As operações de geoprocessamento são executadas neste SGBD via JDBC[34] 2.0 (*Java DataBase Connectivity*), uma API que possibilita aos programas Java estabelecer uma conexão com um banco de dados e enviar comandos SQL para serem executados (MELTON; EISENBERG, 2000), desde que haja um *driver* específico disponível.
- **JXQI**: JXQI (*Java XQuery API*) é uma API Java para XQuery, proposta pela Oracle (JXQI, 2003; ORACLE, 2003), que pode ser utilizada para executar e produzir o resultado das consultas em documentos XML, semelhantemente ao que JDBC faz com instruções SQL (*Structured Query Language*). JXQI é utilizado neste trabalho para executar consultas nas descrições audiovisuais da VideoLib.

---

[31] Um *parser* pode ser visto como um componente de software que recebe como entrada um programa fonte, em formato específico, e o interpreta, de uma forma que possa ser deduzido por computadores.

[32] http://java.sun.com/products/java-media/jmf/

[33] Um código aberto, por definição, significa que o usuário pode obter o código fonte, usar o programa e realizar modificações livremente, sem restrições do proprietário do software.

[34] http://java.sun.com/products/jdbc

4.1.2 Preparação de Conteúdo

Pesquisas relacionadas a vídeo têm produzido padrões, técnicas, ferramentas para criação e apresentação de conteúdo, como também para fornecer semântica a esse conteúdo. A utilização de padrões de metadados baseados em conteúdo desponta como uma das grandes promessas para se desenvolver mecanismos de busca multimídia eficientes na Web. Um dos objetivos do MPEG-7 é fazer com que a Web também seja pesquisada por conteúdo multimídia, como hoje é pesquisada por texto. Atualmente, a disponibilização de informações audiovisuais no formato digital na Internet vem aumentando significativamente, e, por este motivo, torna-se necessário buscar soluções que permitam uma eficiente indexação, consulta e entrega de dados multimídia.

A VideoLib realiza uma segmentação ao nível de cenas e *shots*. Cada vídeo pode ser estruturado como uma árvore onde cada nodo corresponde a um intervalo de tempo, e cada nodo, por sua vez, é decomposto por sucessivos subintervalos temporais. Um exemplo dessa estrutura hierárquica pode ser visto na figura 6.

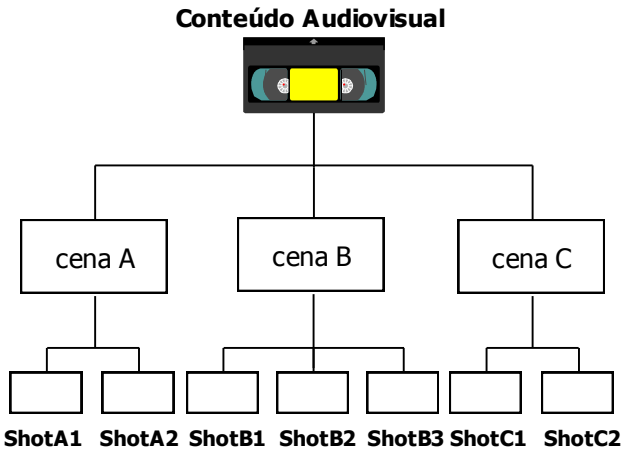

Figura 6. Estrutura hierárquica de um vídeo

Essa estrutura pode ser automaticamente (ou semiautomaticamente) gerada por algoritmos – presentes nas ferramentas de anotação – que realizam detecção de mudança de *shot*, um processo que lida com variações de características de baixo nível do conteúdo audiovisual, tais como som, cor, textura e movimento (SALEMBIER *et al*, 2000).

De modo geral, os usuários preferem utilizar conceitos de alto nível, tais como um título, nome do autor, resumo do conteúdo, etc., para acessar as informações de interesse. Entretanto, o resultado de uma análise automática de vídeo geralmente se detém à extração de características de baixo nível (RUI *et al*, 1998). Assim, ao iniciar o processo de descrição,

torna-se importante adicionar as informações supracitadas, de forma manual, para especificar a representação digital da mídia. Estas anotações facilitam a execução de busca e recuperação de conteúdo, já que objetos não-textuais, como vídeo ou imagens, não possuem uma semântica compreensível por um computador. Existe uma lacuna semântica (*semantic gap*) que deve ser preenchida por anotações que possam tanto ser entendidas por humanos quanto por máquinas, já que ambos possuem uma diferente forma de lidar e interpretar conteúdo (MELAND *et al*, 2003). Como exemplo de *semantic gap*, um humano interpreta um vídeo por meio de ações que estão acontecendo, pessoas envolvidas ou objetos que possivelmente venham a aparecer numa cena. Já um computador abstrai de um vídeo informações sobre número de *frames*, duração, mudanças de características de baixo nível, etc.

A descrição das informações bibliográficas em cada conteúdo audiovisual da VideoLib se dá por meio da representação semântica dos elementos Dublin Core (creator, title, subject, description, coverage, etc.). *Description Schemes* MPEG-7 são instanciados para representar informações sobre a duração do vídeo, tamanho em bytes, conteúdo textual de cada cena ou *shot*, aspectos estruturais, decomposição temporal e especificação espaço-temporal. Metadados são utilizados para representar informações sobre o formato da mídia, duração e localização do conteúdo digital. Cada cena é estruturada em termo de seus *shots* constituintes, decomposição temporal e anotações textuais livres. Os *shots* representam o último nível estrutural de um vídeo, possuindo decomposição temporal e anotações estruturadas sobre pessoas, objetos e ações presentes num subintervalo temporal.

A produção de conteúdo consiste em uma tarefa que requer um maior esforço humano e necessita de ferramentas que auxiliem na extração de informações. Anotações audiovisuais podem ser realizadas de forma manual (processo cansativo e que demanda bastante tempo), semi-automática (segmentação automática e adição de anotações manuais que necessitam de interação humana) e automática (ideal, mas que ainda apresenta resultados imprecisos por constituir em uma tarefa de grande complexidade).

Descrições MPEG-7 são consideravelmente complexas, exigindo desta forma um profundo conhecimento do padrão durante o processo de geração de conteúdo. Para facilitar a criação de tais descrições, torna-se extremamente útil a utilização de uma ferramenta que proporcione a segmentação e anotação de vídeo com metadados que obedeçam a DDL (*Description Definition Language*) do MPEG-7. O *MPEG-7 Alliance Website* relaciona algumas ferramentas que realizam esse tipo de trabalho (Apêndice A).

A VideoLib utiliza o software *Video Description Tool* (VIDETO, 2003), uma ferramenta de extração semi-automática, para gerar suas anotações em conformidade com o

padrão MPEG-7. Inicialmente, a preparação de um vídeo envolve as tarefas de detecção de *shots*, extração de *keyframes* e fornecimento de semântica. Com o VIDETO, um vídeo é carregado e segmentado em diversas partes, denominadas *video shots*, que podem ser discriminadas com algumas informações presentes (lugar, pessoa(s) envolvida(s), objetos, ação, etc.) e associadas a um *keyframe* em particular. Algumas informações de baixo nível sobre a mídia são extraídas automaticamente (como localização, tamanho do arquivo, tipo do arquivo, duração, etc.). Após a análise dos *shots*, as anotações poderão ser salvas no formato MPEG-7. Como podemos notar, informações manuais são necessárias porque dependem da interpretação humana para poder descrever o que está presente em cada cena/*shot* de um vídeo.

É comum haver junção de *shots* na etapa de análise e segmentação para que seja possível descrever uma seqüência de *frames* relevante, pois nem sempre a detecção de *shots* realizada pelas ferramentas expressa um conteúdo satisfatório.

A VideoLib possui seu próprio modelo de descrição (Apêndice B) que difere, em parte, do resultado gerado pelo VIDETO. Neste caso, desenvolvemos um programa de apoio implementado em Java, que tem a responsabilidade de gerar uma descrição final que esteja de acordo com o modelo de descrição adotado por este trabalho. Para isso, o anotador fornece as informações semânticas referentes aos elementos Dublin Core, e em seguida importa os metadados relacionados à decomposição de vídeo e características de baixo nível, produzidos pelo VIDETO, que são relevantes para a VideoLib. Ao final, teremos a descrição audiovisual desejada pela nossa aplicação.

O uso do VIDETO ajuda no trabalho de produção de conteúdo, principalmente na tarefa de detecção automática de *shots* e anotações estruturadas, no entanto, apresenta algumas limitações que coincidem com outras ferramentas afins: o conjunto de metadados MPEG-7 utilizado em suas anotações é muito limitado, e o mesmo não realiza uma segmentação ao nível de cenas e *shots* (apenas em *shots*), tarefa esta que acaba sendo realizada de forma manual e que demanda bastante tempo.

Para vídeos de longa duração, todas as ferramentas analisadas (Apêndice A) não funcionaram de forma satisfatória, deixando o sistema ocupado e muitas vezes sem responder, impossibilitando a continuidade do processo. Vale salientar que esta não é uma limitação da VideoLib, e sim das ferramentas que realizam a segmentação automática de vídeo.

O padrão MPEG-7 independe do conteúdo codificado, podendo ser utilizado para descrever objetos em qualquer formato de vídeo, tais como AVI, WMV, MPEG, MOV, etc. Suas anotações podem ser geradas sem levar em consideração questões de armazenamento,

visualização, transmissão, codificação ou formato da mídia. Entretanto, as ferramentas de anotações MPEG-7 disponíveis, só aceitam, na grande maioria das vezes, arquivos de vídeo codificados no formato MPEG, o que impossibilita a geração de descrições audiovisuais quando outros formatos digitais são manuseados.

A decisão de se trabalhar com o software VIDETO, mesmo com suas limitações, se explica pelos seguintes motivos: possui um conjunto de metadados MPEG-7 que mais pôde ser aproveitado no modelo de descrição da VideoLib; e, principalmente, por ser uma das poucas ferramentas disponibilizadas para *download* sem custo adicional, porém, limitado a um período de avaliação de três meses.

4.1.3 Modelo de descrição

O padrão MPEG-7 pode ser visto como um conjunto de ferramentas de estruturas de descrição genérica (por meio de seus descritores e esquemas de descrição), com semântica associada para ser instanciada por qualquer aplicação multimídia que faça uso de seus metadados. A padronização de esquemas MPEG-7 é útil quando se deseja maximizar a interoperabilidade, porém, é muito improvável que uma única aplicação necessite do completo, e às vezes complexo, conjunto de ferramentas que MPEG-7 propõe. Neste caso, uma forma prática de utilizar o padrão é selecionar as estruturas de descrição desejadas e utilizar àquelas que atendem aos requisitos da aplicação.

O modelo de descrição da VideoLib foi idealizado de modo que possibilitasse a representação semântica e definição de aspectos estruturais, espaciais e temporais de um arquivo multimídia, utilizando, para tais fins, metadados provenientes dos padrões Dublin Core e MPEG-7. O padrão Dublin Core possui um conjunto de elementos especializados (Apêndice C) que têm o objetivo de facilitar a descoberta de recursos eletrônicos e proporcionar uma maior interoperabilidade entre as aplicações.

MPEG-7 permite a seleção de alguns de seus esquemas de descrição para serem usados em conjunto com outros esquemas específicos. O MPEG-7 MDS (*Multimedia Description Scheme*) são estruturas de metadados para a descrição e anotação de conteúdo audiovisual. Possui um conjunto de ferramentas de descrição que lidam com a organização, gerenciamento e representação do conteúdo audiovisual (SALEMBIER; SMITH, 2001).

A interpretação da semântica em um segmento de vídeo é baseada no conteúdo dos descritores MPEG-7 que especificam elementos do contexto, representados pelas seguintes

dimensões: *Who* (pessoas, animais e coisas abstratas), *Where* (lugar), *When* (tempo), *WhatObject* (objetos presentes) e *WhatAction* (Ação). Um contexto pode ser definido como qualquer informação que seja relevante para a interação entre o usuário e a aplicação. Um contexto pode ser categorizado em termos de localização, identidade, atividade e tempo (GOULART; MOREIRA, 2002).

TextAnnotation é um importante componente de muitos DSs (*Description Schemes*) do MPEG-7, possuindo diferentes construções básicas para anotações textuais. A mais flexível construção de anotação textual é o tipo de dado para texto livre. *FreeTextAnnotation* permite a formação de qualquer *string* de texto, que opcionalmente inclui uma informação sobre a linguagem em que o texto está sendo escrito. Entretanto, MPEG-7 fornece também uma ferramenta para anotação textual mais estruturada, por incluir campos específicos correspondentes às questões "Who? What object? What action? Where? e When?". Deste modo, anotações textuais mais complexas podem ser definidas por descrever explicitamente a dependência sintática entre elementos gramaticais pela formação de sentenças (ex: a relação entre um verbo e um sujeito, etc.) (SALEMBIER; SMITH, 2001). Com a interligação do conteúdo dos elementos apropriados, poderemos formular consultas para atender às construções gramaticais de interesse, por exemplo: "Quais os vídeos que contém cenas do piloto Ayrton Senna levantando uma taça?"**,** onde as palavras-chave sublinhadas estariam ligadas, respectivamente, às questões *Who, What Action* e *What Object.*

Embora seja possível realizar um mapeamento entre elementos Dublin Core e seus correspondentes descritores no esquema MPEG-7, a prática deste relacionamento requer uma maior complexidade por motivo de não existir uma correlação 1:1 entre os elementos dos dois padrões, implicando em uma maior dificuldade em termos de recuperação da informação (HUNTER, 2002). Por serem freqüentemente acessados, os elementos Dublin Core deveriam ser facilmente identificados e recuperados dentro do modelo de descrição da VideoLib. Entretanto, utilizando a atual estrutura hierárquica do MPEG-7, elementos Dublin Core como "language" e "format" se encontram no quarto nível do esquema de descrição MPEG-7:

DC:format = MPEG7:MediaInformation.MediaProfile.MediaFormat.FileFormat

DC:language=MPEG7:CreationmetaInformation.Classification.Language.LanguageCode

Muitos dos descritores MPEG-7 que são equivalentes aos elementos Dublin Core são embutidos em um baixo nível dentro dos esquemas de descrição (DSs) do MPEG-7. Objetivando facilitar a aplicação de uma linguagem de consulta XML, para a realização de busca e recuperação de conteúdo audiovisual, a VideoLib resolveu adotar a estrutura híbrida proposta por Hunter (HUNTER, 2002), que representa metadados Dublin Core e MPEG-7 sob

um único esquema de representação em XML. Com esta medida, a semântica dos elementos Dublin Core formariam a base para as descrições audiovisuais, com anotações mais genéricas sobre o vídeo, enquanto que as ferramentas MPEG-7 definiriam os aspectos estruturais, temporais e espaciais do vídeo, características de baixo nível e conteúdo semântico das cenas ou *shots* constituintes.

A interoperabilidade acerca dos elementos Dublin Core seria mantida pelas descrições audiovisuais da VideoLib, podendo ser consultadas por aplicações externas que trabalham com metadados Dublin Core. No entanto, as mesmas descrições não sustentariam a mesma interoperabilidade em ambientes que lidassem exclusivamente com descrições puramente MPEG-7, situação esta de fator previsível porque o padrão permite selecionar um particular subconjunto de relevantes DSs e utilizar este subconjunto na formação estrutural de outros esquemas.

A VideoLib utiliza XQuery (W3C, 2003) como linguagem de consulta aos dados contidos em suas descrições audiovisuais (Apêndice D). Segundo Liu (LIU; HSU, 2002), a aplicação de uma linguagem de consulta XML em documentos MPEG-7 apresenta alguns problemas, visto que MPEG-7 pode apresentar diferentes aspectos estruturais na formação de sua XML. Outro problema consiste na freqüente estrutura irregular de suas descrições, que possibilita a chamada recursiva de um mesmo elemento ao invés de seguir uma hierarquia.

Visando minimizar esses problemas, a VideoLib adotou um padrão de descrição que permitisse a anotação e recuperação audiovisual em dois níveis: cenas e *shots*. Suas descrições devem obedecer o esquema planejado (Apêndice B) para que seja possível realizar as consultas exigidas pela aplicação.

Ademais, a VideoLib incorpora suporte espacial e temporal na realização de suas consultas audiovisuais. No que tange ao espaço, é permitido que o usuário informe um local geográfico onde um determinado vídeo foi filmado, em conjunto com o tipo de operador espacial suportado pela aplicação. Com relação ao tempo, o usuário pode utilizar as funções *Before, After e During* para estabelecer a forma de consulta temporal a ser aplicada em conjunto com datas.

A figura 7 mostra o modelo de descrição da VideoLib.

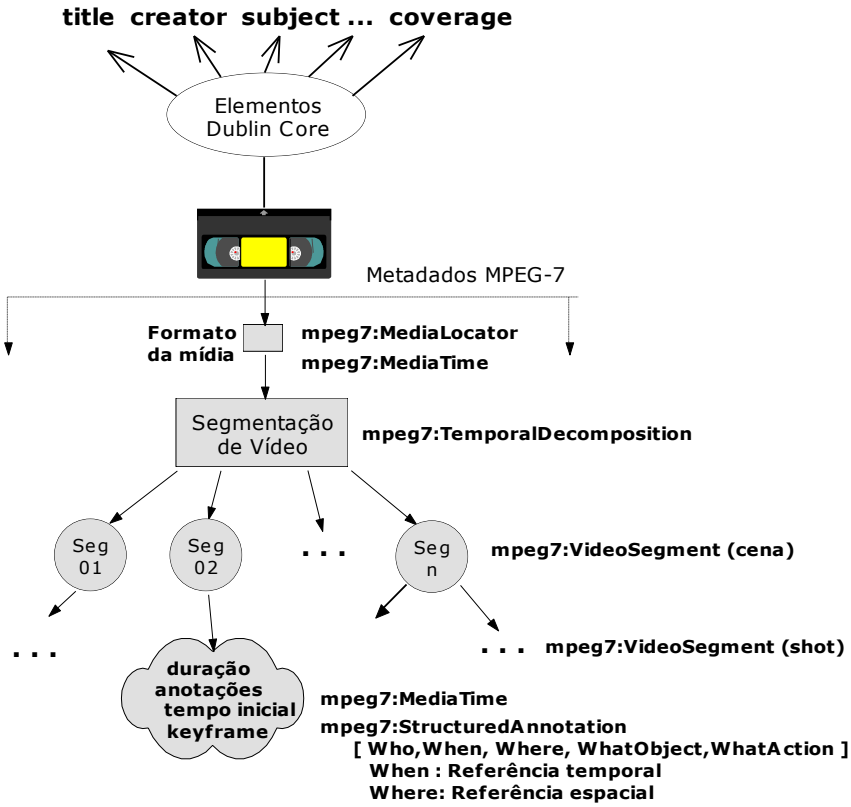

Figura 7. Modelo de descrição da VideoLib

4.1.3.1 Exemplo de uma descrição audiovisual

A figura 8 mostra um exemplo de descrição audiovisual da VideoLib, constituído resumidamente por 1 cena e dois *shots* para que não fique muito extenso. As anotações bibliográficas foram feitas simulando dados reais do conteúdo multimídia, ou seja, como não tivemos acesso às verdadeiras informações de produção, inserimos anotações que dessem continuidade ao processo de descrição, obedecendo a semântica de cada metadado. Como a API JXQI é *case-sensitive*[35] para consultas XQuery, resolvemos padronizar as anotações da VideoLib com caracteres minúsculos.

```
<?xml version="1.0" encoding="UTF-8"?>
<videolib xmlns:dc="http://purl.org/dc/elements/1.1"
          xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:mpeg7="urn:mpeg:schema:2001"
          xs:schemaLocation="http://www.dsc.ufpb.br/~alexscr
```

[35] Sensível à diferença entre caracteres minúsculos e maiúsculos.

```
videolib.xsd
          http://purl.org/dc/elements/1.1 dc.xsd
          urn:mpeg:schema:2001 Mpeg7-2001.xsd">


   <title>nike - selecao brasileira no aeroporto</title>
   <creator>sms producoes audiovisuais</creator>
   <subject>futebol, nike, bola, drible, aviao, aeroporto</subject>
   <description>a selecao brasileira de futebol aguarda o momento de
embarque no aeroporto de londres. Nao suportando o atraso, ronaldo abre
sua bolsa e retira uma bola de futebol, onde comeca a ensaiar algumas
embaixadas. Logo, todos os jogadores participam da brincadeira e
transformam o saguao do aeroporto em um centro de treinamento, dando
inicio a uma serie de muitos dribles e jogadas de
habilidade.</description>
   <publisher>max studio</publisher>
   <contributor>canal 1</contributor>
   <date>1998-06-10</date>
   <type>image.moving</type>
   <format>video-MPEG</format>
   <identifier>videolib006</identifier>
   <source>ufcg</source>
   <language>pt</language>
   <relation>http://www.dsc.ufpb.br/~alexscr</relation>
   <coverage>brazil</coverage>
   <right>copyright ufcg</right>

   <MediaLocator>
      <MediaUri>
http://localhost:8080/videolib/videos/nike01.mpg</MediaUri>
   <MediaLocator>

   <MediaTime>
      <MediaTimePoint>T00:00:00</MediaTimePoint>
      <MediaDuration> PT01M29S</MediaDuration>
   </MediaTime>

   <MediaFormat>
      <Content href="urn:mpeg:cs:ContentCS:2001:2">
         <Name>audiovisual</Name>
      </Content>
      <FileSize>6001000</FileSize>
   </MediaFormat>

   <TemporalDecomposition>
      <!-- Nivel de cenas -->
      <VideoSegment id="01">
         <TextAnnotation>
            <FreeTextAnnotation>denilson corre com a bola e inicia a uma
sequencia de dribles no aeroporto. varias pessoas sao dribladas ate que
ele passa a bola para romario, que por sua vez, mostra toda sua
habilidade pelas dependencias do aeroporto. a cena termina com romario
correndo em direcao ao patio do aeroporto.</FreeTextAnnotation>
         </TextAnnotation>

         <Relation target="
http://localhost:8080/videolib/videos/vd006cen01.jpg" />

         <MediaTime>
           <MediaTimePoint>T00:00:23</MediaTimePoint>
           <MediaDuration>PT00M23S</MediaDuration>
         </MediaTime>
```

```
         <!-- shots constituintes -->
         <TemporalDecomposition>
            <VideoSegment id="shot01">
               <TextAnnotation>
                  <FreeTextAnnotation>denilson recebe a bola e dribla
varias pessoas que circulam pelo aeroporto. o shot termina com denilson
passando a bola para romario</FreeTextAnnotation>
               </TextAnnotation>

               <StructuredAnnotation>
                  <Who>
                     <Name>denilson</Name>
                     <Name>player</Name>
                  </Who>
                  <WhatObject>
                     <Name>bola</Name>
                  </WhatObject>
                  <WhatAction>
                     <Name>driblando</Name>
                  </WhatAction>
                  <Where>
                     <Name>england</Name>
                  </Where>
                  <When>
                     <Name>1998-06-10</Name>
                  </When>
               </StructuredAnnotation>

               <Relation target="
http://localhost:8080/videolib/videos/vd006sht01.jpg" />

               <MediaTime>
                  <MediaTimePoint>T00:00:23</MediaTimePoint>
                  <MediaDuration>PT00M13S</MediaDuration>
               </MediaTime>
            </VideoSegment>

            <VideoSegment id="shot02">
               <TextAnnotation>
                  <FreeTextAnnotation>romario recebe a bola de denilson,
dribla alguns policiais, chuta a bola em direcao ao detector de metais e
recebe do outro lado. em seguida, romario caminha com a bola ate o patio
de avioes.</FreeTextAnnotation>
               </TextAnnotation>

               <StructuredAnnotation>
                  <Who>
                     <Name>romario</Name>
                     <Name>player</Name>
                  </Who>
                  <WhatObject>
                     <Name>bola</Name>
                     <Name>aviao</Name>
                  </WhatObject>
                  <WhatAction>
                     <Name>chutando</Name>
                     <Name>driblando</Name>
                  </WhatAction>
                  <Where>
                     <Name>england</Name>
```

```
                    </Where>
                    <When>
                       <Name>1998-06-10</Name>
                    </When>
                 </StructuredAnnotation>

                 <Relation target="
http://localhost:8080/videolib/videos/vd006sht02.jpg "/>
                 <MediaTime>
                    <MediaTimePoint>T00:00:36</MediaTimePoint>
                    <MediaDuration>PT00M03S</MediaDuration>
                 </MediaTime>
              </VideoSegment>
           </TemporalDecomposition>
        </VideoSegment>
     </TemporalDecomposition>
</videolib>
```

Figura 8. Instância de uma descrição audiovisual na VideoLib

Os elementos Dublin Core são representados pela seqüência de metadados compreendida entre as marcações <title> e <right>, cuja semântica pode ser discriminada no Apêndice C. As anotações remanescentes compreendem o uso de descritores e esquemas de descrição do padrão MPEG-7 utilizados pela VideoLib.

O descritor <MediaLocator> é necessário para especificar a localização do vídeo, provendo referência aos dados da mídia por meio de uma URI (*Universal Resource Identifier*). Os DSs para descrição de tempo são baseados no padrão ISO 8601 (W3C, 1997), uma especificação para representações numéricas envolvendo data e tempo que tem o objetivo de evitar confusões de comunicação causadas pelas diversas notações. O DS <MediaTime> segue a estratégia de descrever um instante de tempo em um intervalo temporal. A instanciação de <MediaTime> antes da segmentação provê informação sobre a duração total do vídeo. Dentro dos segmentos, este DS descreve seu instante inicial e sua duração.

As informações sobre o conteúdo codificado (ex: tamanho em bytes, formato, taxa de bits, tipo do conteúdo, etc.) são representadas pelo descritor <MediaFormat>, que possui uma série de elementos de ocorrência opcional.

<TemporalDecomposition> fornece um conjunto de DSs necessários para realizar a segmentação de vídeo e sua decomposição temporal. A marcação <VideoSegment>, instanciada por meio de um *VideoSegment* DS, permite ser chamada recursivamente. Esta característica possibilita que um vídeo seja estruturado em cenas e *shots*. O primeiro nível hierárquico de um <VideoSegment> na descrição audiovisual corresponde à definição de uma cena. Conseqüentemente, cada ocorrência de um <VideoSegment> dentro de uma cena define

a quantidade de *shots* que a mesma possui. Cada <VideoSegment> possui um atributo "id" que identifica unicamente um segmento de vídeo na descrição audiovisual.

A associação de uma cena ou *shot* com seu *keyframe* é feita por meio do elemento <Relation> do *VideoSegment* DS. O atributo *target* especifica a URL em que a imagem está localizada. *Relations,* em MPEG-7, especificam como um segmento de origem relaciona-se com um segmento alvo.

O *TextAnnotation* DS e seus elementos são usados para prover uma descrição do conteúdo de cada cena ou *shot*. <FreeTextAnnotation>, de conteúdo textual livre, descreve resumidamente o que se passa em cada cena ou *shot.* *TextAnnotation* também permite a representação de anotações estruturadas (<StructuredAnnotation>) em termos de pessoas/animais/coisas abstratas ("Who"), objetos ("WhatObject"), ação ("WhatAction"), lugar ("Where") e tempo ("When"). Para representar a ação que acontece em um segmento de vídeo, convencionamos o uso do verbo no gerúndio.

As anotações audiovisuais da VideoLib estão localizadas no repositório de dados da aplicação (diretório no sistema de arquivos). Com relação ao conteúdo codificado, este pode tanto ser armazenado no contexto da aplicação, como em qualquer URL válida na Web.

4.2 Características Temporal e Espacial da VideoLib

Uma das principais contribuições deste trabalho consiste na sua capacidade de realizar recuperação audiovisual envolvendo suporte espaço-temporal. Salientamos que as características espaço-temporais em vídeos, têm sido exploradas sob a forma de relacionamentos espaciais e temporais entre objetos da cena. Por exemplo, um relacionamento temporal e espacial entre objetos, poderiam ser representados, respectivamente, por cenas onde "um beijo acontece antes de um crime" e aquelas em que "um cachorro estivesse à esquerda de um automóvel".

A VideoLib adotou uma diferente visão para representar estes conceitos. Nela, uma informação temporal está relacionada à data de criação/disponibilização da mídia. Quando citamos a característica espacial, estamos nos referindo à área geográfica onde o vídeo foi filmado, e quais operações de geoprocessamento podem ser aplicadas aos objetos espaciais que representam as superfícies.

Nesta seção, apresentamos detalhes sobre a metodologia utilizada para implementar estas características.

4.2.1 Suporte Espacial

Os usuários que pretendem localizar informações relevantes sobre um determinado assunto, necessitam, na maioria das vezes, de uma referência para uma localização específica usualmente descrita por um nome geográfico. Por exemplo, "Localizar todos os vídeos que foram filmados na América do Sul". Neste caso, os usuários têm interesse em encontrar todas as produções audiovisuais relacionadas à área geográfica da América do Sul, qualquer que seja o país que estiver nela contido. Essa forma de referência espacial, feita através do nome do lugar geográfico, é conhecida como referência indireta e é suportada pelo uso de *Gazetteers* (PAZINATTO, 2003). Um *Gazetteer* pode ser definido como um dicionário de termos geográficos utilizados para determinar um "lugar" geográfico, tal como uma cidade ou país.

O suporte espacial da VideoLib é implementado utilizando-se o método de referência indireta por meio de *Gazetteers*. Este método permite que os usuários referenciem o espaço de diferentes formas para obter a localização espacial. Entretanto, esta forma de referência pode apresentar problemas, pois existem diversos locais geográficos com o mesmo nome (por exemplo: Califórnia no Nordeste do Brasil e Califórnia, nos Estados Unidos) e diferentes nomes para um mesmo lugar geográfico (por exemplo: Pequin e Beijing, na China).

A localização geográfica é indicada pelas coordenadas de latitude e longitude relativas ao lugar geográfico. Para permitir a recuperação espacial, cada local geográfico deve ter seus dados espaciais cadastrados em um SGBD que suporte a manipulação desse tipo de informação.

Um dado espacial é composto pelo *footprint* de um lugar geográfico. Um *footprint* corresponde à área de cobertura de um dado espacial, baseado em geometrias tais como pontos, linhas e polígonos, que representam entidades espaciais, como lagos, rios, cidades e países. Um dado espacial contém atributos como:

- O nome do lugar: nome que determina um lugar geográfico, podendo não ser único para um mesmo local geográfico, como explicitado nas linhas anteriores;
- A localização: determinada pelas coordenadas de latitude e longitude do local geográfico, podendo ser representada por um ponto, retângulo ou polígono;

- O tipo: determina a característica de um lugar geográfico. Por exemplo, o tipo "país" representa lugares geográficos como Brasil, Espanha e Inglaterra.

Os relacionamentos entre os objetos espaciais da VideoLib – os quais correspondem às localizações geográficas – são realizados através de operadores espaciais. Nesta dissertação, utilizamos alguns operadores dos relacionamentos topológicos.

Os relacionamentos topológicos determinam se dois objetos interceptam-se em um plano e qual tipo de interseção existe entre estes objetos. Os principais tipos de relacionamentos obtidos da união dos métodos dos relacionamentos topológicos são: disjunto, encontra (ou toca), igual, dentro_de/contém, cobre/coberto_por e sobrepõe. A figura 9 apresenta os relacionamentos topológicos segundo o método *4-intersection* (EGENHOFER *et al*, 1991). Este método baseia-se em conjuntos para representar as topologias existentes.

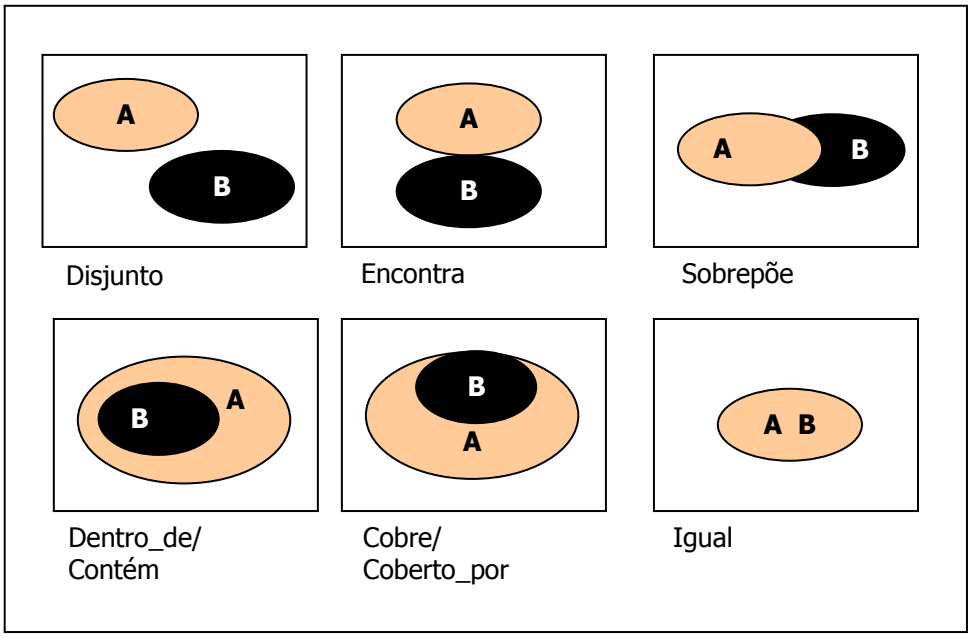

Figura 9. Relacionamento entre conjuntos utilizando operadores espaciais topológicos.
Fonte: Egenhofer *et al* (1991).

O método *4-intersection* indica se existe ou não relacionamento topológico entre dois objetos quaisquer, baseado na comparação das interseções de limite e interior de um objeto A com as partes do objeto B, pela análise dos casos a seguir:

- Disjunto: não existe nenhuma forma de interseção entre os conjuntos A e B;
- Encontra: os limites de A e B se interceptam;
- Igual: existe igualdade entre os conjuntos A e B, ou seja, ambas as interseções de limite e interior não são vazias;

- Dentro_de: existe interseção entre o interior do conjunto A com o interior do conjunto B, no entanto, suas fronteiras não se interceptam. “Dentro_de” possui um relacionamento inverso, chamado “Contém”, com o mesmo significado, mas de sentido contrário;
- Sobrepõe: existe interseção entre os conjuntos A e B, onde os limites do objeto A têm partes em comum com o interior do objeto oposto;
- Cobre: existe compartilhamento entre os limites de B e o interior do conjunto A, com interseção, em parte, dos limites de B com os limites de A. Assim como o “Dentro_de”, “Cobre” tem um relacionamento inverso, chamado “Coberto_por”, com especificação correspondente, mas de sentido contrário.

A VideoLib implementa sua recuperação espacial por meio das operações *Inside* (dentro_de/Contém), *Overlaps* (sobrepõe) e *Outside* (disjunto) entre dois objetos espaciais. Estes operadores foram escolhidos no intuito de simplificar a interação com o usuário.

Cada localização geográfica possui suas coordenadas de latitude e longitude cadastradas no SGBD. A escolha da localização geográfica é feita por meio de um *Gazetteer* (figura 10), que disponibiliza uma interface contendo a estrutura hierárquica de lugares suportada pela VideoLib. A relação das localizações geográficas é mantida em um arquivo XML. Neste caso, o usuário só poderá selecionar localizações devidamente tratadas pela aplicação.

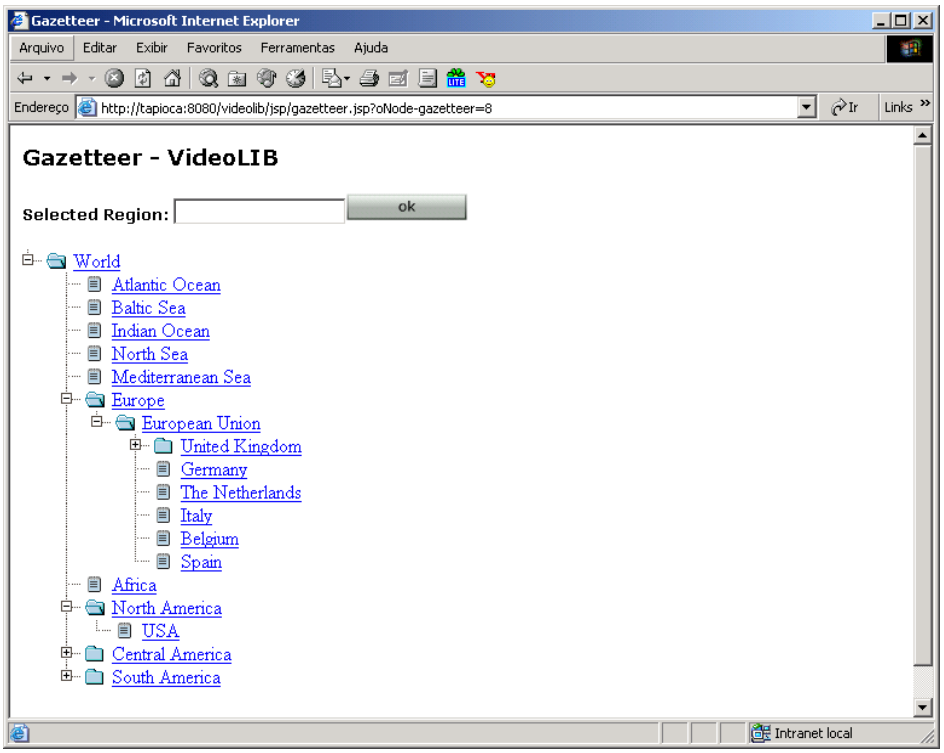

Figura 10. Gazetteer VideoLib

A seleção do lugar “Europa” e da operação espacial “Outside” vai fazer com que sejam recuperados todos os vídeos que tenham sido filmados fora do continente Europeu, ou seja, de acordo com a própria definição do operador *Outside* (Disjunto), serão selecionados todos os vídeos cujo local de produção não intercepta os pontos definidos para a região européia.

O funcionamento completo da recuperação espacial é descrito da seguinte maneira:

1. Acessando a interface Web, o usuário aciona o *Gazetteer* e seleciona o local geográfico desejado, fazendo com que a dimensão “Where” seja utilizada como termo da consulta;
2. Define-se o operador espacial correspondente ao relacionamento topológico de interesse (*Inside, Outside* ou *Overlaps*);
3. Na submissão da consulta, a lógica da aplicação formula a expressão XQuery de acordo com as opções requisitadas na interface. O conteúdo da dimensão “Where” não é utilizado, neste ponto, como cláusula restritiva. Isto se justifica porque a linguagem de consulta XQuery não tem como saber se duas localizações geográficas estabelecem algum tipo de relacionamento espacial;
4. Após executar a expressão de consulta, a relação de vídeos é disponibilizada, internamente, sob a forma de um iterador. Para cada iteração, é enviada uma instrução SQL ao SGBD espacial, cujo objetivo consiste em verificar se as duas localizações geográficas (valor do metadado *Where* da descrição audiovisual e conteúdo da dimensão *Where* na interface gráfica) atendem ao relacionamento topológico. Havendo a confirmação do relacionamento, o vídeo em questão será disponibilizado como item de resultado.

4.2.2 Suporte Temporal

Toda produção audiovisual tem associada ao seu conteúdo uma informação temporal que representa o período em que foi produzido. O tempo pode ser representado por meio de duas formas: um instante, que determina um único momento; ou por um período, que pode ser representado por um intervalo de tempo entre duas datas diferentes (ALLEN, 1983). O uso de dados temporais recebe destaque quando temos sistemas que necessitam recuperar informações baseadas no tempo em que os fatos ocorreram no mundo real.

A dimensão temporal possibilita aos usuários a realização de consultas sobre eventos ou fatos que aconteceram ao longo do tempo. A VideoLib representa suas informações temporais por meio da data de produção de cada conteúdo audiovisual. Assim, torna-se possível recuperar as mais diversas mídias tomando como base um período de referência.

Através dos relacionamentos temporais podemos efetuar análises sobre determinadas informações. Na VideoLib, estes relacionamentos são definidos por meio de três operadores: "*During*", "*After*" e "*Before*". A definição de cada operador é apresentada abaixo:

- Before: um determinado intervalo A está antes de B e eles não se sobrepõem;
- During: o intervalo A começa e termina durante a execução do intervalo B (A contido em B);
- After: um determinado intervalo A se encontra depois de B e eles não se sobrepõem;

Estes operadores temporais são utilizados em diversas aplicações, sendo bastante relevantes para aquelas que lidam com a utilização de conteúdo audiovisual.

Do mesmo modo que os operadores espaciais, as operações temporais também auxiliam no refinamento das consultas. No entanto, é preciso que o usuário especifique um período de tempo para que os operadores possam ser utilizados.

A definição de um período de tempo pode ser feita de duas formas: a primeira, por meio da definição de duas datas, correspondente às datas inicial e final do intervalo; e um instante de tempo, definido por uma data específica.

Após a definição de um período de tempo, o usuário pode selecionar a operação temporal que será aplicada na consulta, representada pelas operações "*During", "After" e "Before"* (figura 11).

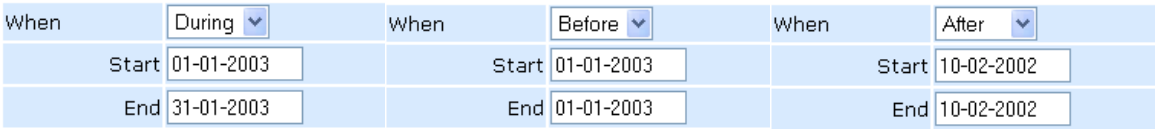

Figura 11. Definição de períodos temporais.

Como podemos observar, todos os operadores temporais necessitam da definição de duas datas, que indicam o início e fim do período.

4.3 Considerações Finais

Neste capítulo, apresentamos os detalhes inerentes à implementação da VideoLib. Relacionamos as principais tecnologias utilizadas; delineamos todo o processo de preparação de conteúdo, destacando as etapas que devem ser seguidas no processo de indexação, bem como o uso de ferramentas de extração semi-automática de informação; descrevemos o modelo de descrição planejado, formado por metadados MPEG-7 e Dublin Core; e finalmente, expomos as informações necessárias para compreensão do suporte temporal e espacial desenvolvido neste trabalho.

O processo de geração de conteúdo, mesmo com a utilização de ferramentas de apoio, tornou-se bastante cansativo. Foram detectados problemas para se trabalhar com vídeos de longa duração, e principalmente, de formatos diferentes, já que MPEG-7 independe do conteúdo codificado.

Obviamente, considerando que os dados utilizados no processo de descrição audiovisual são fictícios e de diversos domínios (comerciais de TV, partidas de futebol, clipes musicais, etc.), não cabe aqui uma discussão sobre a qualidade das anotações. Porém, o mais importante é perceber a capacidade da VideoLib de poder realizar a busca por informações em suas descrições audiovisuais, independente do domínio, e disponibilizar o resultado da consulta. Este resultado foi possível devido à união de metadados Dublin Core e MPEG-7 em um mesmo esquema de descrição.

Capítulo 5

Exemplo de uso da VideoLib

No capítulo 4, especificamos os detalhes inerentes à implementação da VideoLib. Por ser um sistema construído para utilização na Web, o usuário pode acessar a VideoLib e realizar suas consultas através de um *browser*. A interface com o usuário foi idealizada de modo que permitisse realizar busca por conteúdo multimídia utilizando a combinação de vários tipos de informações textuais, obedecendo a semântica de cada elemento de metadado.

O *Gazetteer* implementado pela VideoLib permite a recuperação de informações geográficas por meio do fornecimento de uma localização geográfica como referência. Com a definição de um local geográfico, os usuários podem realizar consultas com o suporte de operações espaciais sobre este lugar. O suporte temporal facilita a recuperação audiovisual através de operações que são aplicadas em conjunto com a data de produção do vídeo desejado. O resultado de uma consulta permite que o vídeo de interesse seja visualizado por completo, ou apenas a seqüência de *frames* que atende aos interesses do usuário.

Neste capítulo, particularizamos os aspectos de utilização da VideoLib. Reproduzimos a interface desenvolvida, apresentando de uma maneira geral, as principais características de seu funcionamento. As janelas de submissão das consultas, com a apresentação dos resultados e todas as etapas que envolvem o processo de uma consulta são descritas. Finalizando, apresentamos as conclusões obtidas com a realização de um teste de performance.

5.1 Interface com o usuário

A interface com o usuário da VideoLib é dividida em duas partes:

1. Janela 1: localizada na parte esquerda da tela, abriga todas as opções que podem ser utilizadas no processo de formação de uma consulta pelo usuário,
2. Janela 2: esta segunda janela, localizada na parte direita da tela, tem o objetivo de exibir o resultado do processamento da consulta e permitir a visualização do vídeo.

A interface inicial da VideoLib, composta por duas janelas, é apresentada na figura 12.

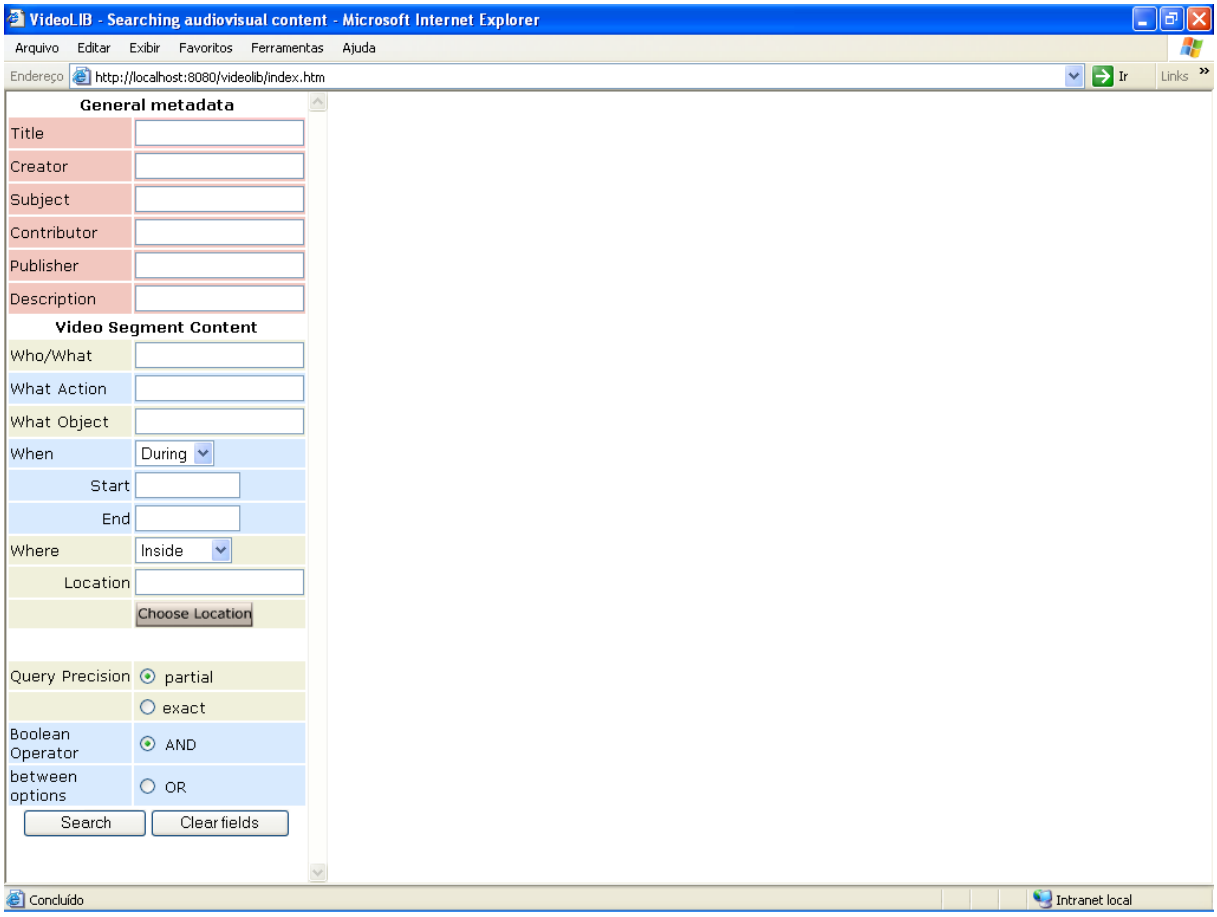

Figura 12. Interface de consulta da VideoLib

No painel principal, relacionado à janela de preparação da consulta, temos a opção de pesquisar pela semântica dos elementos Dublin Core e/ou MPEG-7, este último por meio das dimensões "Who/What", "What Object", "What Action", "When" e "Where".

A opção *"Query Precision"* define a precisão que deve ser levada em conta no momento de execução da consulta. Se a consulta for exata, a busca por uma determinada informação deve obedecer, obrigatoriamente, o conteúdo que foi tomado como referência. Para exemplificar o funcionamento desta opção, tomemos como base as seguintes situações:

1. Digamos que exista uma descrição audiovisual cujo conteúdo do metadado *creator* seja "Washington Olivetto". Se na interface com o usuário a opção *Creator* contiver a palavra chave "Olivetto", e a opção *query precision* estiver com o valor "*exact*" selecionado, a descrição audiovisual supracitada não aparecerá como item de resultado. Para que isto fosse possível, seria necessário que o usuário indicasse na interface, a mesma informação que reflete o conteúdo do metadado *creator* na descrição audiovisual, ou seja, teria que ser fornecido na opção *Creator* da

interface o valor “Washington Olivetto”. Como podemos notar, o modo de precisão *“exact”* é bastante restritivo;

2. Para o mesmo exemplo de descrição audiovisual, consideremos então a seleção da opção *“partial”*. Esta opção funciona similarmente à operação *LIKE “%**keyword**%”* do SQL, onde ***keyword*** representa uma *substring*[36] a ser pesquisada. Este tipo de operação verifica a ocorrência da *substring* em qualquer posição de um texto tomado como referência. Portanto, com o modo de precisão *“partial”,* a consulta anterior ocasionaria resultados. O uso de *“partial”* possui valor default na interface por ser um modo de precisão que melhor reflete o interesse do usuário.

Operadores booleanos são utilizados como conectivos entre as opções selecionadas na interface. Quando uma caixa de texto recebe valor na interface, a VideoLib percebe posteriormente (no momento de submissão da consulta) que tipo de metadado está sendo solicitado como alvo da pesquisa. Quando mais de uma opção de consulta é selecionada na interface, estes são ligados uniformemente entre si por meio de um operador booleano (AND ou OR). Para escolher o operador booleano de interesse, o usuário deve selecionar o valor AND ou OR localizado na opção “*Boolean operator between options”*.

Na prática, o uso do operador funciona da seguinte forma:

1. Se for submetida uma consulta envolvendo as opções *Title=”Brasil”, Creator=”Leonardo” e Publisher=”3D propaganda”,* e o operador escolhido for o AND, implicitamente a VideoLib prepara a expressão de consulta de modo que seja verdadeira a ocorrência de descrições audiovisuais que atendam obrigatoriamente às três solicitações, de acordo com o modo de precisão escolhido (*partial* ou *exact*).
2. Como pudemos notar, o operador booleano AND exige que seja verdadeira a ocorrência das três condições para poder selecionar uma descrição audiovisual. Caso o usuário desejasse obter a relação de vídeos cuja documentação registrasse no *Title* a ocorrência da palavra “Brasil” ou que o criador (*Creator)* fosse “Leonardo”, o operador OR deveria ser escolhido. Com esse tipo de operador, é necessário apenas que uma das opções seja verdadeira para que uma descrição audiovisual seja escolhida. Neste caso, um documento que contenha a ocorrência

[36] *substring* é um termo usado na informática que representa uma cadeia de caracteres que faz parte de uma sequência de caracteres maior.

da palavra “Brasil” no *Title,* mas que não tenha “Leonardo” como *Creator*, será selecionado como item de resultado.

O modelo de descrição da VideoLib permite a ocorrência de todos os elementos Dublin Core em suas descrições audiovisuais, seguindo a orientação do padrão de que todos os elementos são de uso opcional e de ocorrência ilimitada (Apêndice C). Embora o padrão Dublin Core possua um modelo de metadados composto por 15 elementos básicos, apenas seis destes elementos (*title, creator, subject, description, publisher, contributor)* foram considerados mais relevantes para fazer parte da janela de seleção de metadados (janela 1). De toda forma, os metadados que não estão presentes na interface possuem ocorrência nas descrições audiovisuais, e se for o caso, basta apenas que sejam feitas pequenas modificações na lógica da aplicação e interface para que outros elementos possam ser disponibilizados.

Esta limitação também foi influenciada pelo espaço físico de apresentação dos metadados, procurando evitar a rolagem de página da janela de seleção de opções, obedecendo às diretrizes que tratam de questões relacionadas à interface com o usuário (SCHNEIDERMAN, 1993). A ausência de alguns elementos na interface se justifica da seguinte maneira:

- *type:* a VideoLib trabalha apenas com uma categoria de recurso: Vídeos *(Image.Moving).* Por este motivo, não é relevante prover este tipo de informação como opção de consulta porque a VideoLib só possui descrições de vídeo;
- *date:* Consultas envolvendo datas de criação, disponibilização ou acontecimento de um evento podem ser expressas tanto pelo elemento *date* quanto pelo elemento *When* do MPEG-7. Por questão de planejamento, a VideoLib resolveu disponibilizar em sua interface o elemento *When,* embora implicitamente o conteúdo do campo *date* contido nas descrições audiovisuais seja manipulado quando a característica temporal da VideoLib é requisitada. Essa divisão se deu pelo seguinte motivo: caso exista um vídeo do tipo documentário, cujo conteúdo esteja relacionado com a história e imagens de todas as copas do mundo, seria possível recuperar um trecho de vídeo para um determinado período utilizando o elemento *When.* Dessa forma, o metadado *date* estaria relacionado apenas com a data de publicação da mídia, por exemplo.
- *coverage:* a representação textual de uma localização geográfica pode ser expressa pelos metadados *coverage* e *Where,* sendo este último elemento o escolhido para fazer parte da interface por questão de planejamento. Tanto *coverage* quanto *Where* são utilizados como referência quando a característica espacial da VideoLib

é solicitada. A presença destes dois elementos em uma descrição audiovisual se justifica da seguinte forma: tomando como exemplo o fictício documentário citado anteriormente, trechos de vídeo poderiam estar relacionados a diferentes localizações geográficas correspondentes aos países sede de cada copa. Deste modo, o metadado *coverage* estaria relacionado apenas ao local de produção do vídeo, enquanto o elemento *Where* de cada *shot* forneceria a informação sobre o país anfitrião do evento. Assim, o metadado *Where* provê maiores possibilidades de recuperação.

5.2 Processo de submissão de consulta

Conforme apresentado na seção 5.1, assim que o usuário acessa a interface da VideoLib, são apresentados todos os metadados disponíveis para a combinação e formação de consultas. Nesta seção, apresentaremos quatro consultas realizadas na VideoLib com seus respectivos resultados.

5.2.1 Exemplo 1

Consulta: "Localizar todos os vídeos cujo título tenha a ocorrência do nome **comercial**".

Especificamente, um usuário deseja localizar vídeos que estejam relacionados com propagandas comerciais. A palavra-chave "comercial" poderia ser fornecida como valor para as opções "*Title"* ou *"Subject",* segundo a semântica de cada elemento.

Esta consulta lida com uma informação de caráter bibliográfico relacionado ao acervo da VideoLib. Neste caso, deve-se escolher o metadado alvo para consulta, correspondente ao título (*title)*, e fornecer sua respectiva palavra chave (figura 13). Consultas exatas ocasionam em um número menor de resultados porque o valor a ser pesquisado deve coincidir exatamente com o conteúdo do seu metadado correspondente na descrição audiovisual. Caso o usuário não esteja preocupado com o fator exatidão, ele pode realizar uma consulta parcial, considerando apenas a ocorrência da palavra chave em qualquer posição no conteúdo do metadado. Entretanto, o resultado da consulta poderia fugir do verdadeiro interesse do usuário, mas não deixa de ser o modo de precisão mais adequado para ser aplicado. Deste

modo, descrições de vídeo cujo título fossem "**Comercial**ização de Veículos" ou "Escritório de Representação **Comercial**", seriam recuperados pelo mecanismo de busca da VideoLib.

É importante lembrar que elementos Dublin Core e MPEG-7, este último representado pelas entradas Who, What Action, What Object e When, podem ser usados em conjunto para formular uma consulta. Cada caixa de texto da interface possui uma dica visando auxiliar o usuário na escolha correta do metadado que será usado como referência na pesquisa.

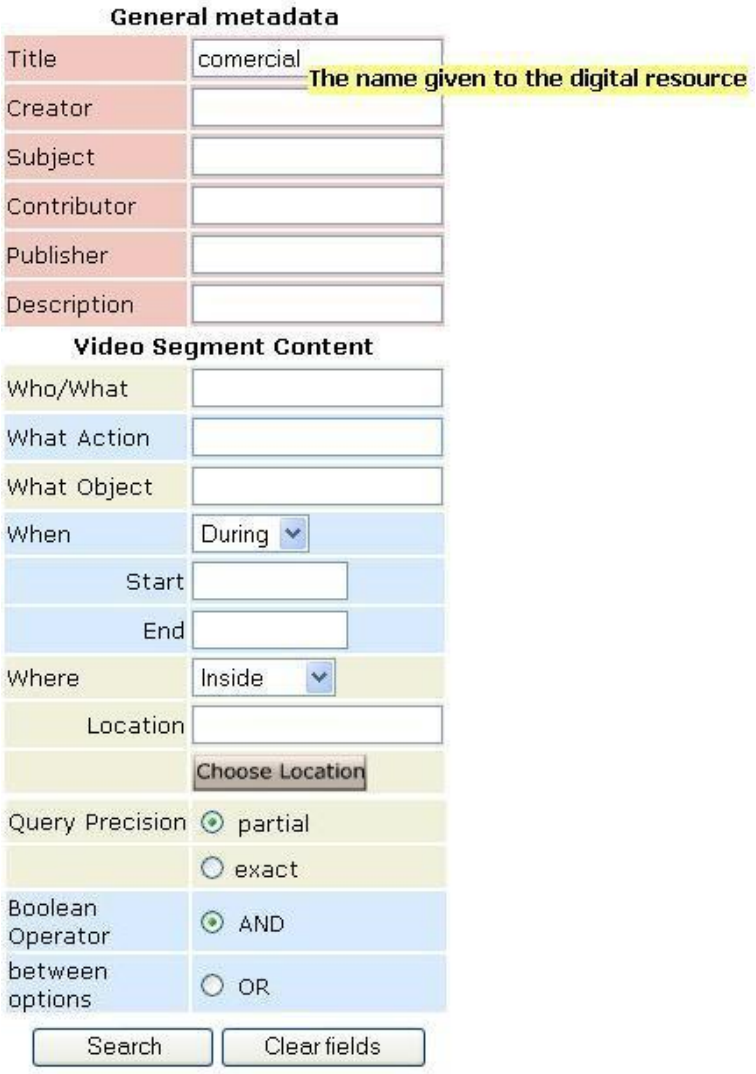

Figura 13. Formulação da consulta 1

Toda consulta leva em consideração o tipo de operador selecionado (AND ou OR) e o modo de precisão da consulta (*exact* ou *partial*). Após o preenchimento das caixas de texto de interesse, o usuário deve pressionar o botão *Search* para dar início ao processamento da consulta, tarefa esta realizada por um *Servlet* especializado. Internamente, a lógica da aplicação da VideoLib planeja a seguinte consulta XQuery (figura 14):

```
    FOR $library IN document('     ')/videolib
       WHERE contains($library/title/text(),"comercial")
       RETURN
           <videolib>
             { $library/title }
             { $library/creator }
             { $library/MediaLocator/MediaUri }
             { $library/format }
             { $library/date }
             { $library/description }
             { $library/identifier }
             { $library/coverage }
</videolib>
```

Figura 14. Expressão XQuery para a consulta 1.

A consulta é elaborada de maneira que sejam retornados apenas os elementos de cada descrição audiovisual considerados importantes para montar a lista de resultados. Podemos notar que a expressão não está ligada diretamente a um tipo de descrição audiovisual em particular (instrução em negrito), isto porque a VideoLib aplica a mesma consulta para cada arquivo XML contido no repositório de dados, fazendo a substituição do nome do arquivo a cada iteração.

A figura 15 mostra uma parte do resultado obtido, após a realização da consulta em questão, com destaque visual para a palavra-chave utilizada como parâmetro de consulta. Os vídeos selecionados disponibilizam um *link* na parte superior de cada resultado (URL) para que seu conteúdo seja visualizado por completo. Os botões *"List Segments"* e *"List Shots"* expandem o resultado de cada ocorrência encontrada, selecionando respectivamente as cenas e *shots* constituintes no vídeo.

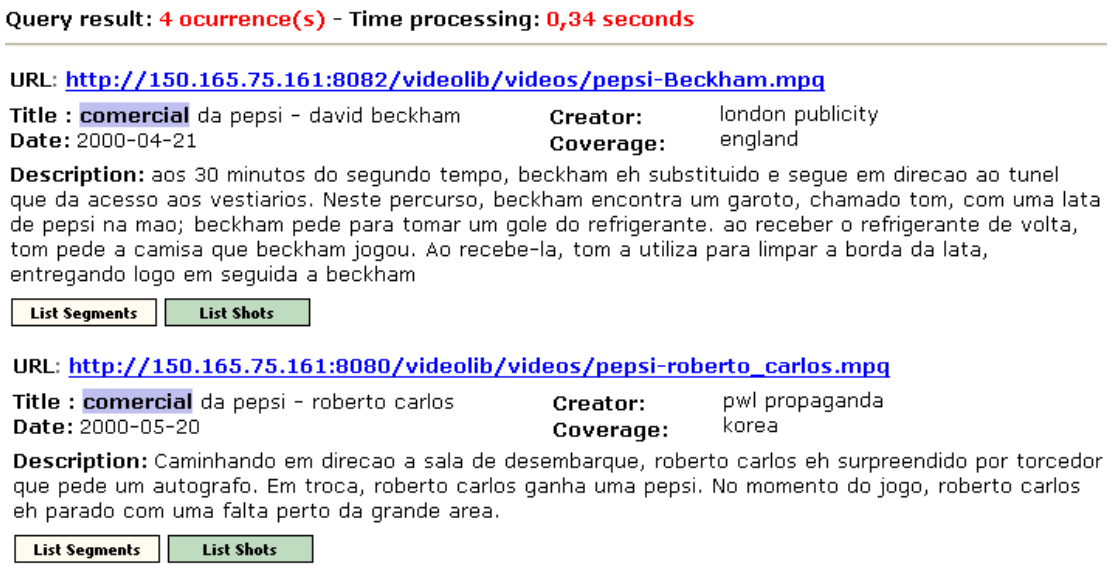

Figura 15 Trecho do resultado da consulta 1

Tomando como exemplo o primeiro resultado da lista, ao pressionar o botão *"List Segments"*, teremos o seguinte resultado (figura 16):

Figura 16. Consulta expandida para visualização das cenas

Como os segmentos listados pertencem unicamente ao vídeo tomado como referência, todas as ocorrências apresentarão as mesmas informações relativas ao *Title, Creator, Date, Coverage* e URL de localização do vídeo. A diferença entre cada segmento é definida por meio do seu resumo textual, visível na interface, e intervalo temporal pelo qual está implicitamente associado.

5.2.2 Exemplo 2

Consulta: "Localizar todos os segmentos de vídeo em que o jogador **Beckham** esteja **tomando refrigerante**".

Neste tipo de consulta, devemos abstrair algumas informações relevantes do enunciado e utilizá-las como entrada para a execução da pesquisa. Como exemplo, temos a presença de uma pessoa (**Beckham**) envolvida na cena, praticando a ação de **tomar** um **refrigerante**, este último enquadrado na categoria "objeto". Para formular esta consulta, o usuário deve selecionar as dimensões de interesse (*Who*, *What Action* e *What Object*) e entrar com dados em suas respectivas caixas de texto (figura 17). Por default, consultas que envolvem mais de uma dimensão de contexto possuem seus elementos interligados pelo operador booleano AND. Se for do interesse do usuário, ele pode modificar o operador de consulta para OR e

realizar uma consulta do tipo: “Selecione os segmentos de vídeo em que o jogador **Beckham** esteja presente OU que haja a ação de **tomar** (no gerúndio) alguma bebida OU que seja mostrado um **refrigerante**”.

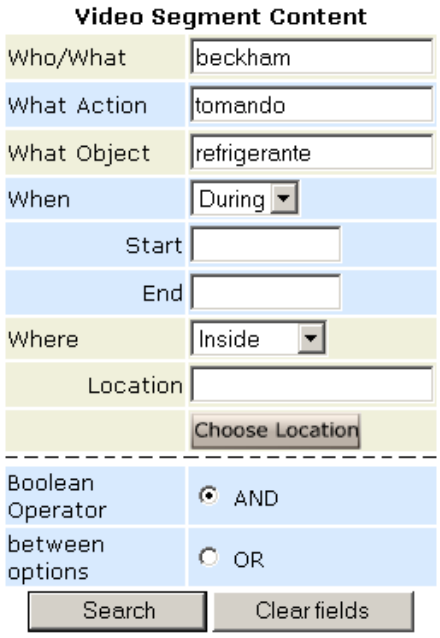

Figura 17. Formulação da consulta 2

O diferencial desta consulta em relação a anterior é que, além de verificar o relacionamento do conteúdo entre as dimensões contextuais selecionadas, temos a opção de visualizar o trecho de vídeo, representado pelo botão “*View Content*” (figura 18), correspondente à consulta submetida pelo usuário. Isto é possível para toda consulta que requisitar informações sobre as dimensões “Who, What Action, What Object e When”.

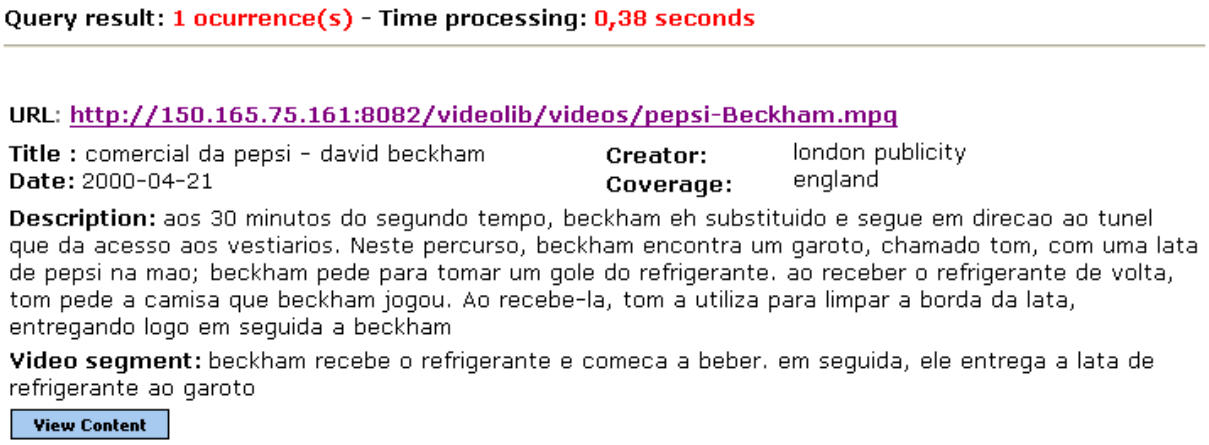

Figura 18. Resultado da consulta 2

Pressionando o botão “View Segment”, a VideoLib localiza a seqüência de *frames* correspondente ao segmento de vídeo e envia o conteúdo para que possa ser exibido pelo *browser* (figura 19). Essa seqüência de *frames* é obtida por meio de um processamento interno que utiliza a tecnologia JMF. Um *Servlet* fica encarregado de receber como entrada o intervalo temporal requisitado e a URL do vídeo, acionar a lógica da aplicação repassando os parâmetros necessários e enviar o trecho de vídeo correspondente para que possa ser visualizado pelo *browser* do usuário.

Figura 19. Visualização do segmento de vídeo

Para produzir o resultado desta consulta, a expressão XQuery da figura 20 foi gerada e executada. Todas as expressões XQuery da VideoLib são produzidas dinamicamente, de acordo com os metadados que foram escolhidos na interface com o usuário.

```
<resultado>
{
LET $library := document('    ')/videolib
FOR $cena IN $library/TemporalDecomposition/VideoSegment,
    $shot IN $cena/TemporalDecomposition/VideoSegment,
    $video IN $shot/StructuredAnnotation
WHERE contains($video/WhatAction/Name/text(),"tomando")
   AND contains($video/Who/Name/text(),"beckham")
   AND contains($video/WhatObject/Name/text(),"refrigerante")
RETURN
    <videolib>
      { $library/title }
      { $library/creator }
      { $library/MediaLocator/MediaUri }
      { $library/format }
      { $library/date }
      { $library/description }
      { $library/identifier }
      { $library/coverage }
```

```
            {
               FOR $temp IN $shot
               WHERE contains($video/WhatAction/Name/text(),"tomando")
          AND contains($video/Who/Name/text(),"beckham")
          AND contains($video/WhatObject/Name/text(),"refrigerante")
               RETURN
               <VideoSegment id="{ $temp/@id }">
                  { $temp/TextAnnotation/FreeTextAnnotation }
                  { $temp/MediaTime }
               </VideoSegment>
            }
        </videolib>
      }
      </resultado>
```

Figura 20. Expressão XQuery para a consulta 2.

5.2.3 Exemplo 3

Consulta: "Localizar todos os segmentos de vídeo em que **Ronaldo** esteja jogando **Futebol** no período de **25/06/2002** a **30/06/2002**".

Este exemplo ilustra o funcionamento da característica temporal da VideoLib. A dimensão "When" permite especificar um intervalo de tempo como restrição da consulta. Deste modo, o usuário deverá selecionar a opção ***"During"*** (período compreendido entre duas datas), fornecendo logo em seguida os valores 25-06-2002 e 30-06-2002 como entrada de dados para os campos *Start* e *End*. A formulação da consulta é concluída assim que os valores "futebol" e "Ronaldo" forem associados, respectivamente, às opções "Subject" e "Who/What" (figura 21). Além de determinar um período compreendido entre duas datas, também é possível refinar a consulta para que, por exemplo, só sejam mostrados os segmentos de vídeos produzidos a partir do ano de 2003, selecionando na interface o operador de tempo *"After"* e fornecendo como período inicial e final, o valor 01-01-2003. Da mesma forma, pode ser feita a seleção dos segmentos de vídeo produzidos antes do ano 2003, escolhendo na interface o operador de tempo *"Before"* e fornecendo como data de referência o valor 01-01-2003 (período inicial e final).

Figura 21. Formulação da consulta 3

Como podemos verificar na figura 22, mesmo que haja mais trechos de vídeo relacionados ao jogador Ronaldo, apenas aqueles que foram produzidos entre 25/06/2002 e 30/06/2002 farão parte do resultado da pesquisa.

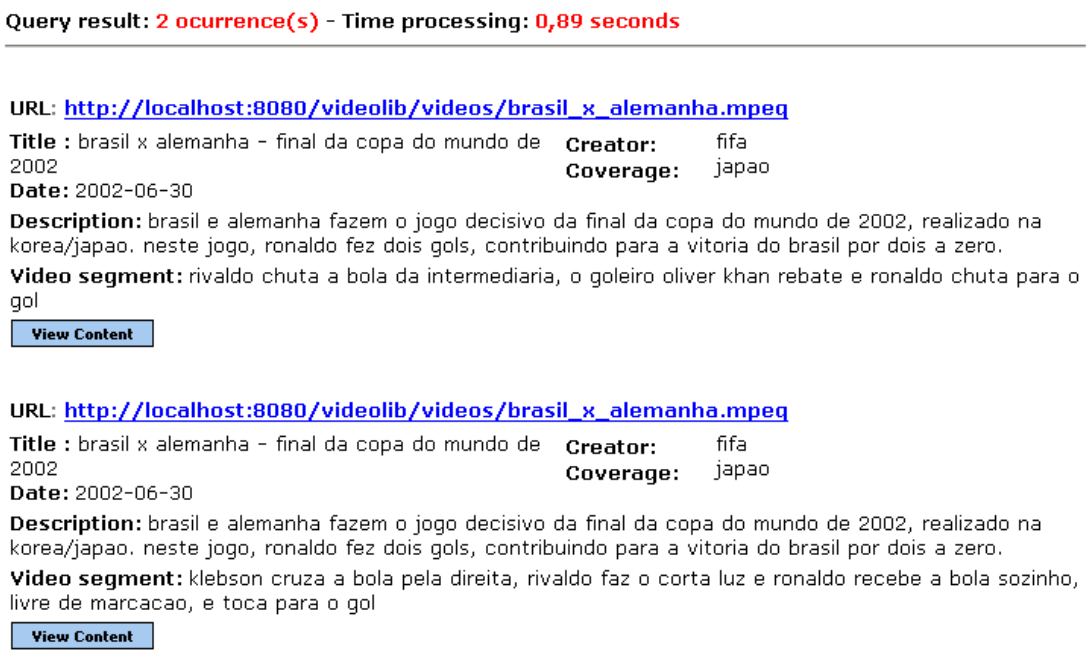

Figura 22. Resultado da consulta 3

Os valores temporais são fornecidos na interface da VideoLib no formato Dia-Mês-Ano (DD-MM-AAA). Como as anotações temporais nas descrições audiovisuais seguem o padrão ISO 8601 (AAAA-MM-DD), torna-se necessário realizar uma conversão no formato da data para que a expressão XQuery possa ser executada de forma correta (figura 23).

```
<resultado>
{
LET $library := document(' ')/videolib
FOR $cena IN $library/TemporalDecomposition/VideoSegment,
    $shot IN $cena/TemporalDecomposition/VideoSegment,
    $video IN $shot/StructuredAnnotation
WHERE contains($library/subject/text(),"futebol")
    AND contains($video/Who/Name/text(),"ronaldo")
    AND $video/When/Name >= '2002-06-25'
    AND $video/When/Name <= '2002-06-30'
RETURN
    <videolib>
       { $library/title }
       { $library/creator }
       { $library/MediaLocator/MediaUri }
       { $library/format }
       { $library/date }
       { $library/description }
       { $library/identifier }
       { $library/coverage }
       {
       FOR $temp IN $shot
       WHERE contains($library/subject/text(),"futebol")
             AND contains($video/Who/Name/text(),"ronaldo")
             AND $video/When/Name >= '2002-06-25'
             AND $video/When/Name <= '2002-06-30'
       RETURN
          <VideoSegment id="{ $temp/@id }">
             { $temp/TextAnnotation/FreeTextAnnotation }
             { $temp/MediaTime }
          </VideoSegment>
        }
    </videolib>
}
</resultado>
```

Figura 23. Expressão XQuery para a consulta 3.

Analisando a expressão da figura 23, poderemos observar que a condição de teste da consulta (cláusula WHERE) é executada em dois momentos. Isto se justifica pelos seguintes motivos:

1. A primeira expressão FLWR (Apêndice D) tem o objetivo de verificar apenas se existe algum *shot* no documento com as características desejadas pela consulta, validando a descrição audiovisual que está sendo analisada;
2. Caso a condição seja verdadeira, a segunda expressão FLWR, localizada no construtor de elementos, fica responsável por preparar o resultado de saída para cada *shot* retornado. Se uma descrição audiovisual selecionada na primeira expressão possuir muitos *shots*, apenas aqueles que atendem aos requisitos da pesquisa serão recuperados.

5.2.4 Exemplo 4

Consulta: "Localizar todos os vídeos que foram filmados no **Reino Unido**".

Este exemplo demonstra a utilização da característica espacial da VideoLib.

Os usuários que pretendem localizar informações relevantes sobre um determinado assunto necessitam, na maioria das vezes, de uma referência para uma localização específica usualmente descrita por um nome geográfico. No exemplo em questão, os vídeos que foram produzidos no Reino Unido englobam quaisquer produções que tenham sido filmadas, por exemplo, na Inglaterra, Escócia ou País de Gales. Essa forma de referência espacial, feita através do nome do lugar geográfico, é conhecida como referência indireta e é suportada pelo uso de *Gazetteers* (PAZINATTO, 2003). Esta tarefa é realizada em conjunto com um banco de dados que forneça suporte a operações de geoprocessamento.

Visando selecionar a localização geográfica de interesse, o usuário deverá clicar no botão "*Choose Location"* (figura 24). Como resultado, é montada uma estrutura hierárquica correspondente às localizações geográficas tratadas pela VideoLib sob a forma de uma árvore, cujas informações estão armazenadas em um arquivo XML. Ao selecionar a região de interesse e pressionar o botão "OK", o resultado da operação é transportado para a caixa de texto correspondente à dimensão "Where" da interface com o usuário.

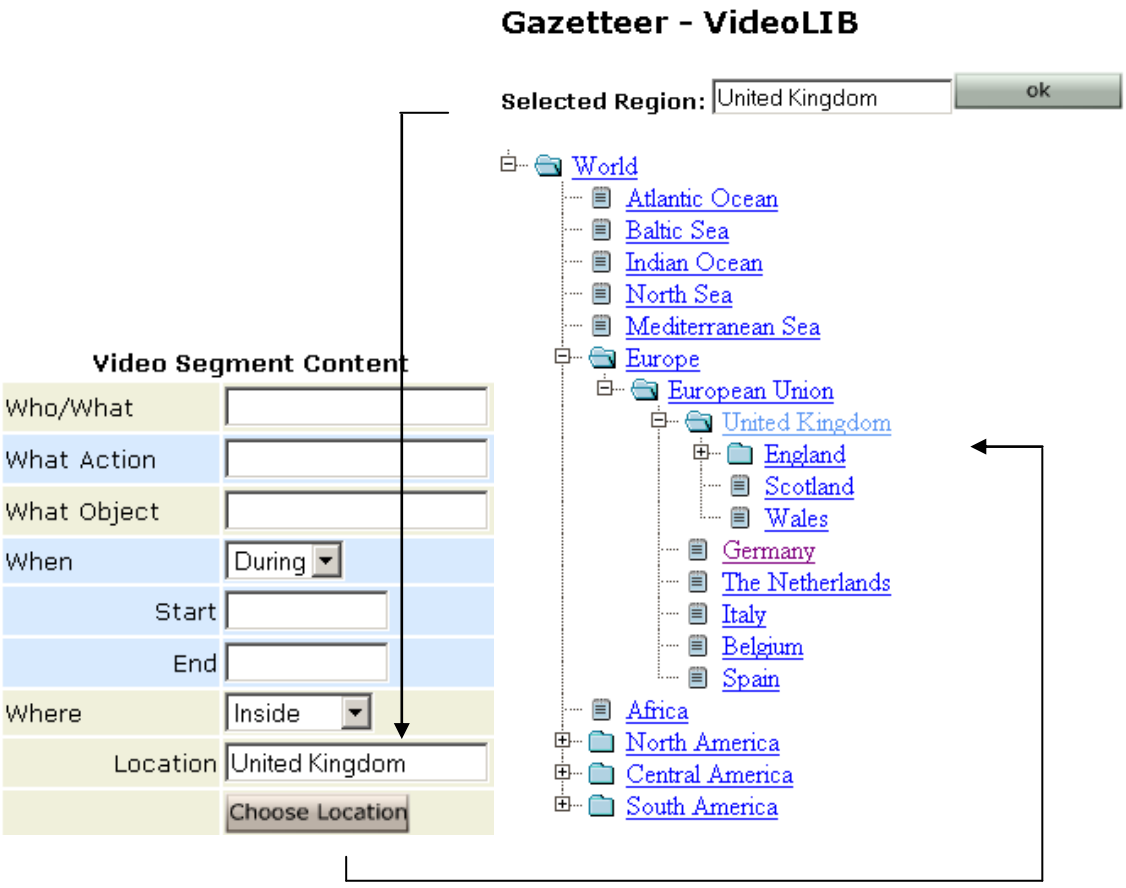

Figura 24. Formulação da consulta 4

Finalmente, para "consultar os vídeos que foram filmados no Reino Unido", deve-se aplicar o tipo de operação espacial correspondente à opção desejada. Neste caso, o operador *Inside* deve ser selecionado antes da submissão da consulta. Além do *Inside*, a VideoLib dá suporte ao uso das operações *Outside* e *Overlaps*. Vale salientar que outros elementos de metadado da interface com o usuário podem ser combinados juntamente com a característica espacial da VideoLib para formular uma consulta.

O resultado da execução da consulta (figura 25) exibe uma lista cujo conteúdo consiste nos mais diversos vídeos que tiveram sua filmagem realizada no Reino Unido.

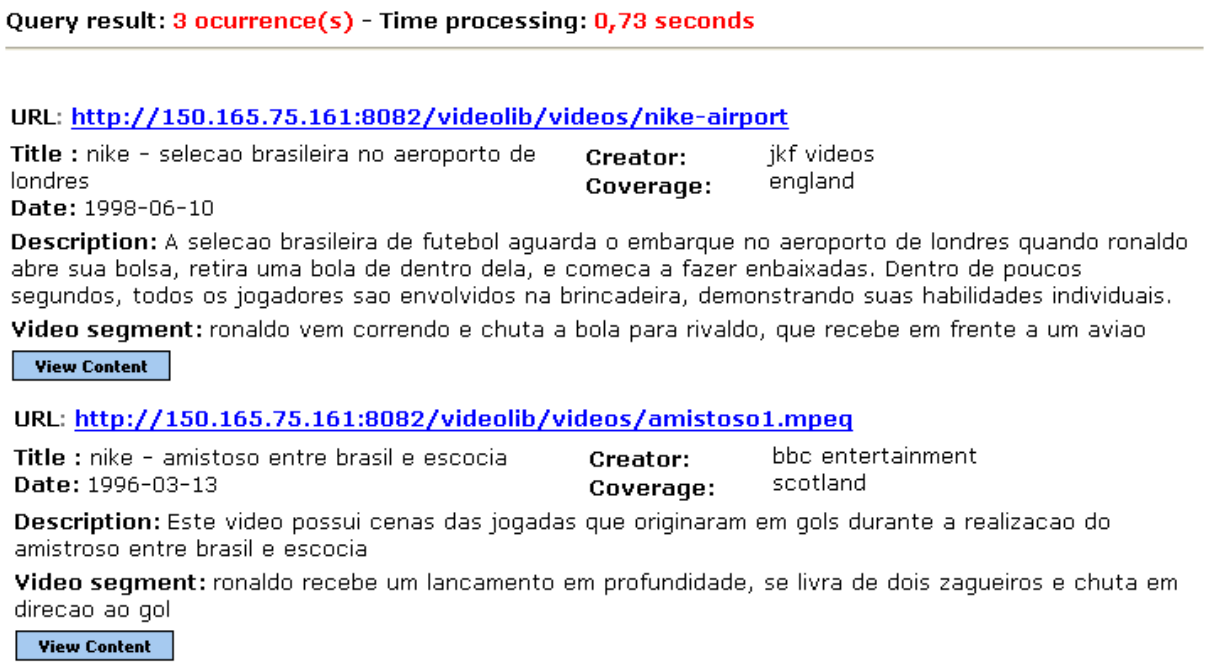

Figura 25. Resultado da consulta 4

A recuperação espacial da VideoLib funciona da seguinte forma:

1. A princípio, a expressão XQuery não tem como saber se uma localização geográfica está dentro dos limites de outra e que tipo de interseção existe entre eles. Esta tarefa fica por conta de um SGBD com suporte a operações espaciais. Portanto, o conteúdo da dimensão “Where” não entra na formação da expressão XQuery (figura 26) , mas é utilizado em outro momento;

```
<resultado>
{
LET $library := document('    ')/videolib
FOR $cena IN $library/TemporalDecomposition/VideoSegment,
    $shot IN $cena/TemporalDecomposition/VideoSegment,
    $video IN $shot/StructuredAnnotation
RETURN
    <videolib>
      { $library/title }
      { $library/creator }
      { $library/MediaLocator/MediaUri }
      { $library/format }
      { $library/date }
      { $library/description }
      { $library/identifier }
      { $library/coverage }
      {
         FOR $temp IN $shot
                  RETURN
         <VideoSegment id="{ $temp/@id }">
            { $temp/TextAnnotation/FreeTextAnnotation }
```

```
                { $temp/MediaTime }
              </VideoSegment>
          }
      </videolib>
    }
    </resultado>
```

Figura 26. Expressão XQuery para a consulta 4.

2. Desta forma, a expressão de consulta XQuery primeiramente seleciona todas as produções audiovisuais que satisfazem a consulta,. Em seguida, a lógica da aplicação conecta-se ao SGBD e executa a operação espacial correspondente, verificando a ocorrência do relacionamento espacial entre as duas localizações geográficas, representadas pelo conteúdo da dimensão "Where" (interface com o usuário) e da *tag* <Where> na descrição audiovisual. Para cada iteração resultante, apenas as descrições de vídeo que satisfazem o relacionamento espacial serão retornados.

A maioria das consultas mostradas tem o objetivo de ilustrar separadamente as funcionalidades da VideoLib. Nada impede, por exemplo, que diversos elementos sejam unidos para resolver consultas do tipo: "recupere todas as **propagandas** da **pepsi** em que o jogador de futebol **Beckham** esteja **tomando refrigerante**, com a filmagem do vídeo realizada no **Reino Unido** no ano de **1998".**

5.3 Teste de Performance

A VideoLib trabalha com descrições audiovisuais armazenadas no sistema de arquivos da máquina servidora. Como estamos utilizando XQuery para elaborar e ativar as consultas, a utilização de um padrão de pesquisa deve ser aplicado a todas as instâncias de documentos XML contidos em seu domínio. Neste caso, temos um repositório de arquivos, situado no diretório raiz do servidor de aplicação, que armazena todas as descrições de vídeo. Este diretório tem o objetivo similar ao de um banco de dados no que diz respeito apenas à questão de armazenamento, sem levar em consideração características como controle de transação,

criação de índices, etc. O mecanismo de busca foi implementado em Java utilizando JXQI, uma API Java XQuery proposta pela Oracle (ORACLE, 2003).

Fizemos uma avaliação de performance da VideoLib, para testar o desempenho das consultas XQuery para determinadas quantidades de instâncias descritivas, ou seja, pela quantidade de descrições audiovisuais existentes no repositório de dados. Fizemos testes realizando uma busca aproximada (*partial)* para as seguintes consultas:

1. "Localizar todos os segmentos de vídeo que tenha(m) **computador**(es)".

2. "Localizar todos os segmentos de vídeo em que **Romário** esteja presente".

3. "Localizar todos os segmentos de vídeo em que **Toquinho** esteja **tocando violão**".

Inicialmente, utilizamos 10 descrições de vídeo diferentes. Para cada consulta executada, uma determinada quantidade de segmentos de vídeo atendia às necessidades do usuário. Vale salientar que na consulta de um vídeo, zero (se os segmentos não atenderem a consulta) ou mais segmentos podem retornar como resposta. As descrições foram clonadas na mesma proporção para simular o custo de processamento com 100, 500 e 1000 anotações de vídeo. Portanto, o resultado do número de ocorrências é diretamente proporcional ao número de vídeos, como pode ser observado na tabela 1.

Tabela 1. Demonstrativo do número de ocorrências de segmentos de vídeos para as consultas pré-determinadas de acordo com a quantidade de descrições audiovisuais.

Tarefa	100 Vídeos	500 Vídeos	1.000 Vídeos
Consulta 1	70	350	700
Consulta 2	40	200	400
Consulta 3	10	50	100

Para os testes realizados, utilizamos um computador com 1 GB de memória RAM e processador Pentium IV 1.8 Ghz para realizar as simulações em um ambiente que utiliza a tecnologia JSP/Servlet. Podemos ver no gráfico 1 que as consultas começam a perder em performance a partir de 500 vídeos analisados. Como a mesma consulta é realizada isoladamente para cada descrição de vídeo contida no repositório, a tendência era de que houvesse uma queda de rendimento quando o número de anotações audiovisuais aumentasse significativamente.

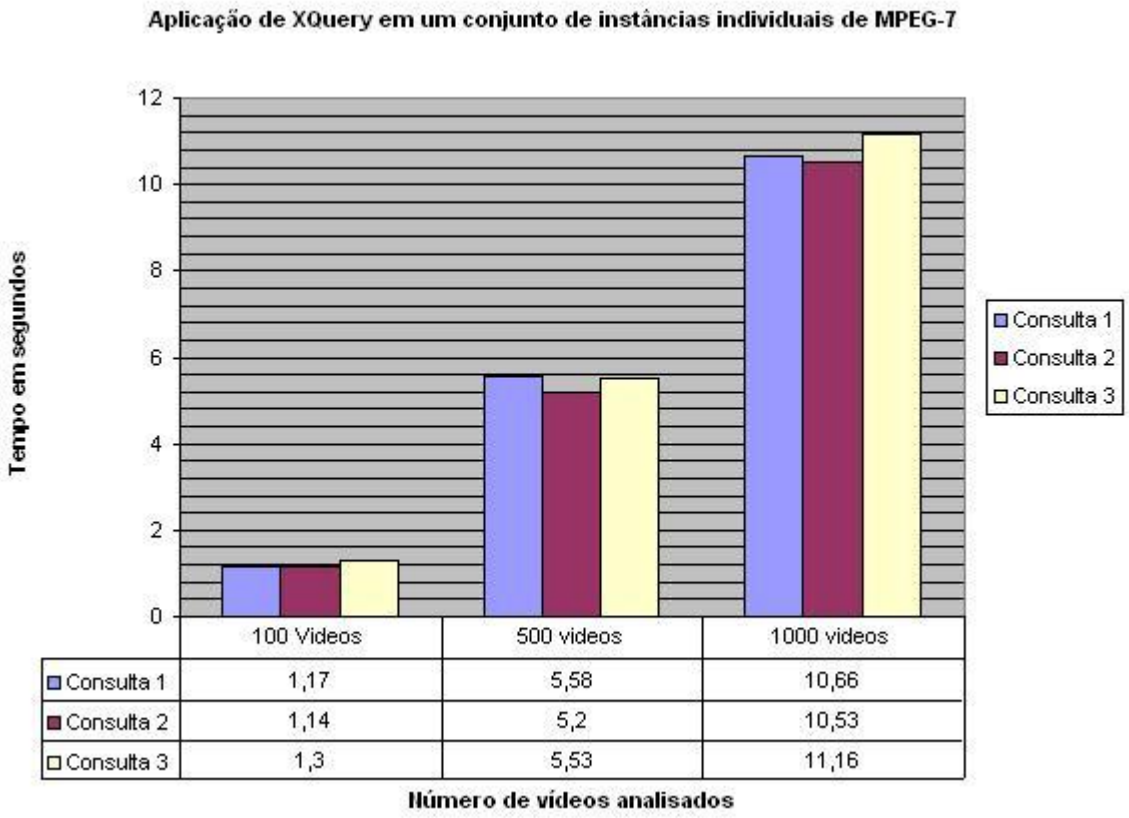

Gráfico 1. Tempo de processamento de cada consulta ao conjunto de descrições de vídeo contidos no repositório de dados.

Para melhorar os tempos de resposta, estudos já estão sendo feitos para verificar a possibilidade de migrar a base de dados para um Sistema de Gerência de Banco de Dados com suporte a XML e que, principalmente, sustente o mesmo padrão de consulta planejado pelo nosso trabalho, mantendo assim uma homogeneidade na elaboração das consultas.

Capítulo 6

Conclusão

O desenvolvimento de bibliotecas digitais de vídeo consiste, na atualidade, em um problema desafiador. Nesse contexto, a manipulação de dados multimídia torna aparente dois problemas: o cansativo processo manual de anotação e a subjetividade da percepção humana devido às diferentes interpretações que podem ser extraídas de um conteúdo audiovisual. Esta subjetividade de interpretação, ligada à possível imprecisão do processo de anotação, pode ocasionar inconsistência entre a procura por dados multimídia e o resultado da busca.

A produção de conteúdo, mesmo com o auxílio de softwares de apoio, ainda consiste em uma tarefa cansativa. As ferramentas de segmentação automática disponíveis que trabalham com MPEG-7 apresentam deficiências e estão sendo alvo de aprimoramento. Como exemplo, na maioria dos casos, só é possível segmentar um vídeo em nível de *shots*; diferentes formatos de vídeo não são suportados; e há problemas de processamento para arquivos com grande quantidade de conteúdo.

O conjunto de metadados MPEG-7 foi desenvolvido com o objetivo de padronizar a descrição de dados multimídia e beneficiar a criação de aplicações em diversos domínios. A geração de descrições audiovisuais é produzida em conjunto com alguma ferramenta que realize a extração semi-automática de informações do conteúdo audiovisual. Com seu conjunto de ferramentas descritivas, MPEG-7 fornece uma eficiente indexação e representação de conteúdo por possibilitar a busca por material multimídia em uma busca por descrição textual. O uso do padrão Dublin Core – em lugar dos metadados equivalentes no padrão MPEG-7 – facilitou a representação das informações bibliográficas e, como resultado direto, tornou mais fácil a recuperação de informações e a aplicação de uma linguagem de consulta XML.

Nesta dissertação, desenvolvemos uma biblioteca digital que permite o armazenamento e recuperação de vídeo, utilizando um esquema de descrição que engloba metadados Dublin Core e MPEG-7 para a representação do conteúdo digital. Por ser um padrão recente, poucos trabalhos lidam com a utilização do MPEG-7 na recuperação de conteúdo audiovisual. Visando explorar os recursos oferecidos pelo padrão, utilizamos as ferramentas descritivas do MPEG-7 para estruturar hierarquicamente os vídeos em cenas e *shots*, proporcionar a decomposição temporal de cada cena/*shot* e possibilitar a anotação de

informações estruturadas. Através destes mecanismos, um vídeo pode ser indexado para que seu conteúdo não seja acessado de forma linear, permitindo a visualização do segmento de vídeo de interesse do usuário.

A VideoLib procura unir as funcionalidades de outros trabalhos e suprir as carências apresentadas pelos mesmos. Deste modo, estabelecemos uma nova abordagem de acesso a vídeos, ao introduzirmos suporte às dimensões espacial e temporal. Estas dimensões elevam qualitativamente o processo integral de busca, disponibilizando operadores espaciais e temporais para melhor atender as necessidades do usuário. O suporte espacial é implementado utilizando-se o método de referência indireta, através do uso de um *Gazetteer*.

6.1 Principais Contribuições

A implementação de algumas características diferencia a VideoLib de outros trabalhos desenvolvidos. A seguir, descrevemos as suas principais contribuições:

- Suporte à dimensão espacial: com o *Gazetteer*, implementamos a forma de referência espacial indireta através do nome de um local geográfico. Este *Gazetteer* disponibiliza uma interface, contendo a estrutura hierárquica de lugares suportada pela VideoLib, que deverá ser utilizada para selecionar a localização geográfica de interesse. A recuperação espacial permite a seleção de conteúdo audiovisual de forma mais inteligente, verificando o relacionamento entre as localizações geográficas de acordo com um local de referência e operador espacial selecionado (*overlaps*, *outside* e *inside*);
- Suporte à temporalidade dos dados: com a informação sobre a temporalidade dos dados é possível recuperar material multimídia produzido em um período de tempo determinado. Introduzimos a utilização dos operadores *"before", "after"* e *"during"* para a realização das operações temporais;
- Diferentes formas de submissão de consulta: a consulta pode ser formulada através da combinação dos elementos contidos na interface do usuário, onde cada elemento possui uma semântica diferente, relacionados aos metadados do padrão Dublin Core e MPEG-7. Todo elemento selecionado é interligado entre si, para efeito de busca, por meio de um conectivo (AND ou OR) a ser definido pelo usuário.

6.2 Trabalhos Futuros

A partir dos resultados obtidos com o desenvolvimento da VideoLib, identificamos algumas questões importantes para o aperfeiçoamento e extensão deste trabalho. A seguir, apresentamos algumas tarefas que podem ser desenvolvidas como continuidade de nossa pesquisa:

- Desenvolvimento de uma ferramenta de extração de vídeo que forneça detecção automática de *shot* e produza anotações utilizando metadados Dublin Core e MPEG-7: no decorrer desta dissertação, relatamos as dificuldades encontradas durante o processo de geração de conteúdo. Por exigir distintas etapas para produzir uma única anotação (detecção de *shots* por uma ferramenta de extração semi-automática, estruturação manual ao nível de cenas e *shots* e acréscimo de informações Dublin Core), esta tarefa se torna bastante cansativa. Para minimizar este problema, torna-se necessário desenvolver uma ferramenta de anotação, para integração com a VideoLib, que realize a detecção automática de *shots* e produza anotações utilizando metadados Dublin Core e MPEG-7, no modelo de descrição idealizado por este trabalho.
- Migração das anotações audiovisuais para um SGBD que suporte o armazenamento e recuperação de arquivos XML: as descrições audiovisuais da VideoLib se encontram armazenadas em um diretório acessível pelo servidor de aplicação *tomcat*. Uma instrução XQuery é formulada e aplicada em cada descrição audiovisual contida neste diretório. Pudemos verificar, pelo teste de performance, que a aplicação começa a perder em desempenho quando manipula um número de descrições audiovisuais superior a 500 anotações de vídeo.

  Para melhorar o tempo de resposta para grandes quantidades de descrições de vídeo, o uso de um SGBD com suporte a XML seria bastante relevante por disponibilizar índices de acesso aos dados, controle de concorrência, dentre outras vantagens. Deste modo, um estudo deve ser feito para indicar qual SGBD pode ser utilizado para armazenar, e consultar as descrições audiovisuais de forma eficiente.

  Durante a fase de implementação, tivemos a oportunidade de realizar esta migração para o banco de dados Oracle 9i, que fornece esse tipo de suporte através

do tipo de dado *XMLType*[37]. Este tipo de dado possui funções que operam sobre o conteúdo XML, com maior destaque para a função *extract()*[38]. Entretanto, por não possuir estruturas de repetição e outros recursos que podemos encontrar na linguagem XQuery, a função *extract()* não consegue devolver um resultado similar ao que conseguimos com instruções XQuery. Como as descrições audiovisuais que utilizam metadados MPEG-7 são geralmente complexas, não foi possível obter os mesmos resultados utilizando a função *extract().*

Deste modo, outros bancos de dados com suporte a XML nativo devem ser analisados, para verificar se atendem às questões de armazenamento e recuperação de descrições audiovisuais, influenciando diretamente na melhoraria das medidas de desempenho.

- Implementação de um servidor de vídeo: o foco da VideoLib não está direcionado para a apresentação de vídeo. Desenvolvemos nesse trabalho uma solução que permite a visualização do trecho de um vídeo (resultante do processo de indexação), mas que necessita de aprimoramento. Porém, o desenvolvimento de um servidor de vídeo que implementasse as funções de codificação, armazenamento e entrega do conteúdo através de *streaming* traria uma significante qualidade na tarefa de apresentação do conteúdo multimídia.
- Proporcionar a elaboração de consultas mais complexas: a representação de elementos Dublin Core em uma descrição é opcional e de ocorrência ilimitada. Assim, o sistema deve dar suporte para que o usuário forneça, opcionalmente, vários valores em cada elemento de metadado (*Title*, *Creator*, *Subject*, etc.), separando cada valor por meio de um operador booleano (AND/OR). Da mesma forma, essa extensão se aplica aos metadados Who/What, What Action e What Object.
- Estender as características espaciais e temporais, acrescentando a noção de relacionamento entre objetos presentes em uma cena, tempo relativo (início do filme) e absoluto (da cena em si).

---

[37] http://download-west.oracle.com/docs/cd/B10501_01/appdev.920/a96620/xdb04cre.htm

[38] Baseada em XPath, *extract()* é uma função utilizada em cláusulas SQL que extrai nodos específicos de uma instância XMLType.

REFERÊNCIAS

(ABITEBOUL *et al*, 2000) ABITEBOUL, S.; BUNEMAN, P.; SUCIU, D.: **Gerenciando dados na Web**. Porto Alegre: Campus, 2000.

(AL-KHATIB *et al*, 1999) AL-KHATIB, W.; DAY, Y., D.; FELLOW, A., G.; FELLOW, P.,B.,B.: **Semantic Modeling and Knowledge Representation in Multimedia Databases**. IEEE Transactions on Knowledge and Data Engineering, vol.11, n. 1, p. 64-80, Jan./Feb. 1999.

(ALLEN, 1983) ALLEN, J.: **Maintaining knowledge about temporal intervals**. Comunications of ACM, vol. 26, Issue 11, p. 832-843, 1983.

(APACHE, 2003) The Apache Jakarta Project. Disponível em: <http://jakarta.apache.org>. Acesso em: 20 jan. 2003.

(ARMS, 2001) ARMS, W.: **Digital Libraries**. [s.l.]: MIT Press, 2001.

(BOOCH, 1999) BOOCH, G.: **The Unified Modeling Language User Guide**. Massachusetts: Addison-Wesley, 1999.

(BRAY *et al*, 2000) BRAY, T.; PAOLI, J.; SPERBERG-MCQUEEN, M., C.; MALER, E.: **Extensible Markup Language (XML) 1.0 (Second Edition)**. W3C Recommendation, October, 2000. Disponível em: <http://www.w3.org/TR/2000/REC-xml-20001006>. Acesso em: 15 dez. 2003.

(CHANG ; SIKORA, 2001) CHANG, F.; SIKORA, T.: **Overview of the MPEG-7 Standard**. IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, n. 6, June, 2001.

(CHRISTEL *et al*, 1996) CHRISTEL, M.; STEVENS, S.; KANADE, T.; MAULDIN, M.; REDDY, R.; WACTLAR, H.: **Techniques for the Creation and Exploration of Digital Video Libraries**. In: Multimedia Tools and Applications. Kluwer Academic Publishers, 1996.

(CHRISTEL, 1995) CHRISTEL, M.: **Addressing the Contents of Video in a Digital Library**. In: ACM Workshop on Effective Abstractions in Multimedia. San Francisco, California, 1995.

(CHRISTEL; WACTLAR, 2002) CHRISTEL, M.; WACTLAR, H.: **Digital Video Archives: Managing Through Metadata**. In: Building a National Strategy for Digital Preservation, Issues in Digital Media Archiving, Library of Congress, April, 2002.

(CURTY, 2002) CURTY, M., G.; CRUZ, A.,M.; MENDES, M.,T.,R.: **Apresentação de Trabalhos Acadêmicos, Dissertações e Teses.** NBR14724. Maringá: Dental Press, 2002.

(DC, 2003) Dublin Core Metadata Element Set, Version 1.1: Reference Description. Disponível em: <http://dublincore.org>. Acesso em: 05 abr. 2003.

(DEITEL; DEITEL, 2001a) DEITEL, H., M.; DEITEL, P., J.: **Java Como Programar.** Porto Alegre: Bookman, 2001.

(DEITEL; DEITEL, 2001b) DEITEL, H., M.; DEITEL, P., J.: **XML Como Programar.** Porto Alegre: Bookman, 2001.

(ECKEL, 1998) ECKEL, B.: **Thinking in Java**. New Jersey: Prentice Hall, 1998.

(EDWARDS, 1999) EDWARDS, J.: **3-Tier Server/Client at Work**. [s.l.]: John Wiley & Sons, 1999.

(EGENHOFER *et al*, 1991) EGENHOFER, M.; HERRING, J.; SMITH, T.; PARK, K.: **A Framework for the Definition of Topological Relationships and Algebraic Approach to Spatial Reasoning within this Framework**. In: National Center for Geographic Information and Analysis, NCGIA, 1991.

(FATEMI; KHALED, 2001) FATEMI, N.; KHALED, O.: **Indexing and Retrieval of TV News Programs Based on MPEG-7**. ICCE International Conference, June, 2001.

(GAMMA *et al*, 2000) GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSIDES, J.: **Padrões de Projeto: soluções reutilizáveis de software orientado a objetos**. Porto Alegre: Bookman, 2000.

(GOSLING *et al*, 2000) GOSLING, J.; JOY, B.; STEELE, G.; BRACHA, G.: **Java™ Language Specification**. [s.l.]: Addison-Wesley Publisher, Second Edition, 2000.

(GOULART; MOREIRA, 2002) GOULART, R.; MOREIRA, R.: **Representação de Objetos de Mídia para Aplicações Conscientes de Contexto em TV Interativa.** SBMIDIA, Fortaleza, 2002.

(HALL, 2000) HALL, M.: **Core Servlets and Java Server Pages**. [s.l.]: Sun Microsystems Press, First Edition, May, 2000.

(HUNTER, 1999) HUNTER, J.: MPEG-7 Behind the Scenes. In: **D-Lib Magazine**, v. 5, n. 9, September, 1999.

(HUNTER, 2001) HUNTER, J.: **An Overview of the MPEG-7 Description Definition Language (DDL)**. IEEE Transactions on Circuits and Systems for Video Technology, v. 11, n. 6, June, 2001.

(HUNTER, 2002) HUNTER, J.: **An Application Profile which combines Dublin Core and MPEG-7 Metadata Terms for Simple Video Description**. In: ViDE Video Access Group, Feb 12, 2002.

(IANNELLA; HUNTER, 1998) IANNELLA, R.; HUNTER, J.: **The Application of Standards to Video Indexing**. In: Second European Conference on Research and Advanced Technology for Digital Libraries. Greece, September, 1998.

(ISO/IEC, 2002) Martínez, J.: **Overview of the MPEG-7 Standard, MPEG Document**: ISO/IEC/JTC1/SC29/WG11 Klangenfurt, July, 2002. Disponível em: <http://mpeg.telecomitalialab.com/standards/mpeg-7/mpeg-7.htm.>. Acesso em: 23 ago. 2003.

(JACOBSON *et al,* 1992) JACOBSON, I.; CHRISTERSON, M.; JONSSON, P.; OVERGAARD, G.: **Object-Oriented Software Engineering: A Use-Case Approach**. Addison-Wesley, USA, 1992.

(JMF, 2003) Java Media Framework 2.1.1 Solutions (JMF): **Cut Sections from an Input**, April 2003. Disponível em: <http://java.sun.com/products/java-media/jmf/2.1.1/solutions/>. Acesso em: 05 nov. 2003.

(JXQI, 2003) JXQI – The Java XQuery API. Disponível em: <http://otn.oracle.com/sample_code/tech/xml/xmldb/jxqi.html>. Acesso em: 17 mar. 2003.

(KIM; SONG, 2001) KIM, D.; SONG, Y.: **Interoperable Summary Description Model Using Dublin Core**. Proc. Conf. on Dublin Core and Metadata Applications, 2001.

(LARMAN, 2000) LARMAN, C.: **Utilizando UML e Padrões: Uma introdução à análise e ao projeto orientado a objeto**. Porto Alegre: Bookman, 2000.

(LIU; HSU, 2002) LIU, P.; HSU, L., S.: **Queries in MPEG-7 and MPEG-21 XML Documents**. XMLEurope, Barcelona, Spain, May, 2002.

(MARTÍNEZ *et al,* 2002) MARTÍNEZ, J.; KOENEN, R.; PEREIRA, F.: **MPEG-7: The Generic Multimedia Content Description Standard, Part 1**. IEEE Multimedia, v. 9, Issue: 2 , Apr-Jun, 2002.

(MELAND *et al,* 2003) MELAND, P., H.; AUSTVIK, J.; HEGGLAND, J.: **Using Ontologies and Semantic Networks with Temporal Media**. In: Semantic Web Workshop at the 26th Annual International ACM SIGIR Conference, Toronto, Canada, 2003.

(MELTON; EISENBERG, 2000) MELTON, J.; EISENBERG, A.: **Understanding SQL and Java Together**. [s.l.]: Academic Press, 2000.

(NACK; LINDSAY, 1999) NACK, F.; LINDSAY, A.T.: **Everything you Wanted to Know about MPEG-7**. IEEE Multimedia, v. 6 , Issue: 3 , p. 65 – 77, July-Sept. 1999.

(ORACLE, 2003) Oracle Tecnology Network 2003. Disponível em: <http://otn.oracle.com>. Acesso em: 03 jun. 2003.

(PAZINATTO, 2003) PAZINATTO, E.: **GeoLocalizador: Um Sistema de Referência Espaço-Temporal Indireta na Web**. 2003. Dissertação (Mestrado) – Centro de Ciências e Tecnologia, Universidade Federal de Campina Grande, Campina Grande, 2003.

(PEREIRA, 1997) PEREIRA, F.: **MPEG-7: A Standard for Content-Based Audiovisual Description**. In: VISUAL'97, San Diego (CA) - USA, December, 1997.

(PFEIFFER; SRINIVASAN, 2000) PFEIFFER, S; SRINIVASAN, U.: **TV Anytime as an Application Scenario for MPEG-7**. In: Proceedings of the 2000 ACM Workshop on Multimedia, Los Angeles, California, November, 2000.

(POSTGRESQL, 2003) PostgreSQL Global Development Group: "*PostgreSQL Documentation".* Disponível em: <http://www.postgresql.org/docs/>. Acesso em: 21 dez. 2003.

(RICOH, 2003) RICOH. Ricoh Movie Tool. Disponível em <http://www.ricoh.co.jp/src/multimedia/MovieTool/>. Acesso em: 07 jun. 2003.

(RUI *et al*, 1998) RUI, Y.; HUANG, T., S.; CHANG, S.: **Digital Image/Video Library and MPEG-7: Standardization and Research Issues**. In: IEEE ICASSP'98 , p.3785-3788, May 12-15, Seattle, 1998.

(SALEMBIER *et al*, 2000) SALEMBIER, P.; LLACH, J.; GARRIDO, L.: **Visual Segment Tree Creation for MPEG-7 Description Schemes**. In: International Conference on Multimedia and Expo, ICME'2000, vol. 2, p. 907-910, New York City, USA, July, 2000.

(SALEMBIER, 2002) SALEMBIER, P.: **Overview of the MPEG-7 Standard and of Future Challenges for Visual Information Analysis**. EURASIP Journal on Applied Signal Processing, 2002.

(SALEMBIER; SMITH, 2001) SALEMBIER, P; SMITH, J.: **MPEG-7 Multimedia Description Scheme**. In: IEEE Transactions on Circuits and Systems for Video Tecnology, vol 11, n. 6, June, 2001.

(SCHNEIDERMAN, 1993) SCHNEIDERMAN, B.: **Designing the User Interface - Strategies for Effective Human-Computer Interaction**. [s.l.]: Addison-Wesley Publishing Company, Second Edition, 1993.

(SETTEN; OLTMANS, 2001) SETTEN, M.; OLTMANS, E.: **Demonstration of a Distributed MPEG-7 Video Search and Retrieval Application in the Educational Domain.** In: Proceedings of the Ninth ACM International Conference on Multimedia, October, 2001.

(SMEATON, 2000) SMEATON, A.: **Indexing, Browsing and Searching of Digital Video and Digital Audio Information.** In: European Summer School in Information Retrieval, Tutorial Notes, Varenna, Lago di Como, Italy, 2000.

(SWAIN, 1999) SWAIN, M., J.: **Searching for Multimedia on the World Wide Web**. In: IEEE International Conference on Multimedia Computing and Systems, vol. 1, July, 1999.

(TV-ANYTIME, 2003) The TV Anytime Forum. Disponível em: <http://www.tv-anytime.org>. Acesso em: 07 jan. 2004.

(VIDEOANNEX, 2003) IBM MPEG-7 Annotation Tool. Disponível em: <http://alphaworks.ibm.com/tech/videoannex>. Acesso em: 14 out. 2003.

(VIDETO, 2003) **VIDETO – Video Description Tool**. Disponível em: <http://www.rostock.zgdv.de/ZGDV/Abteilungen/zr1/Produkte/videto/index_html_en>. Acesso em 09 nov. 2003.

(W3C, 1997) World Wide Web Consortium (W3C): **Date and Time Formats**, W3C Note (September 1997). Disponível em: <http://www.w3.org/TR/NOTE-datetime>. Acesso em 17 ago. 2003.

(W3C, 1999a) World Wide Web Consortium (W3C): **XSL Transformations (XSLT) Version 1.0**, on-line (November 1999). Disponível em: <http://www.w3.org/TR/xslt>. Acesso em: 28 dez. 2003.

(W3C, 1999b) World Wide Web Consortium (W3C): **XML Path Language – XPath, Version 1.0** (November 1999). Disponível em: <http://www.w3.org/TR/1999/REC-xpath-19991116>. Acesso em: 28 ago. 2003.

(W3C, 2001) World Wide Web Consortium (W3C) : **XML Schema part 0: Primer**, on-line (May 2001): Disponível em: <http://www.w3.org/TR/xmlschema-0/>. Acesso em: 05 set. 2003.

(W3C, 2003) World Wide Web Consortium (W3C): **XQuery 1.0: An XML Query Language**, on-line (May 2003). Disponível em: <http://www.w3.org/TR/2003/WD-xquery-20030502/>. Acesso em: 02 jun. 2003.

(XQL, 1999) **XQL – XML Query Language**. (August 1999). Disponível em: <http://www.ibiblio.org/xql/xql-proposal.html>. Acesso em: 10 jan. 2004.

(YEO; YEUNG, 1997) YEO, B.; YEUNG, M., M.: **Retrieving and Visualizing Video**. In: Communications of the ACM, December, 1997.

Apêndice A

Estudo de Caso das Ferramentas de Anotação MPEG-7

A produção de conteúdo audiovisual necessita da utilização de ferramentas que auxiliem na extração de informações. Estas ferramentas são desejadas para facilitar o processo de segmentação de vídeo, adicionar informações manuais e gerar o resultado textual das anotações. No *MPEG-7 Alliance Website*[39] são indicados alguns softwares que realizam essas tarefas e, principalmente, armazenam o resultado das anotações como descrições MPEG-7. Fizemos uma breve análise de três ferramentas disponíveis.

A.1 – IBM VideoAnnEx Annotation Tool

O ***IBM VideoAnnEx Annotation Tool*** (VIDEOANNEX, 2003) é uma ferramenta que possibilita realizar anotações com metadados MPEG-7 em arquivos no formato MPEG. Cada *shot* em uma seqüência de vídeo pode ser anotado com descrições de uma cena estática (cena de fundo estática), descrição de objetos chaves e eventos que estão ocorrendo. As anotações são associadas com cada *shot* de vídeo e armazenadas como descrições MPEG-7 em um arquivo de saída XML. A interface do VideoAnnEx é ilustrada na figura 27.

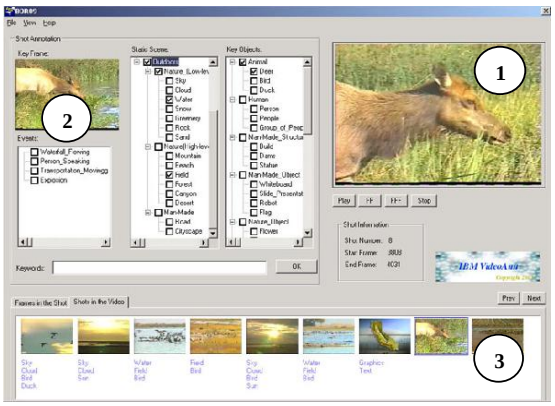

Figura 27. Interface do IBM VideoAnnEx Annotation Tool

A ferramenta é dividida em três regiões: (1) Video Playback; (2) Anotação de *shot* com uma imagem de *keyframe*; e (3) Painel de visualização**.** Possui ainda uma quarta janela

[39] http://www.mpeg-industry.com/

destinada para especificação de anotações em regiões, funcionalidade esta que não se encontra disponível na versão oferecida para avaliação.

A.2 – Ricoh Movie Tool

MovieTool (RICOH, 2003) é uma ferramenta para descrição de vídeo baseado nas regras da DDL MPEG-7. Segundo o fabricante, a maior vantagem em se utilizar *MovieTool* é que o usuário pode rapidamente e facilmente verificar a correspondência entre as descrições MPEG-7 e a estrutura de vídeo de cada cena. Anotações podem ser adicionadas a cada cena, tornando-as parte do arquivo MPEG-7.

As principais características do *MovieTool* são:

- Cria uma descrição MPEG-7 ao carregar um vídeo (MPEG-1);
- Fornece orientações visuais para ajudar o usuário na criação da estrutura do vídeo;
- A estrutura do vídeo é automaticamente refletida na descrição MPEG-7;
- Exibe o relacionamento entre a estrutura e a descrição MPEG-7;
- Apresenta *tags* candidatas para ajudar o usuário na escolha das *tags* MPEG-7 apropriadas;
- Verifica a validação das descrições MPEG-7 para ver se está de acordo com o esquema MPEG-7;
- Permite descrever todos os metadados definidos no padrão MPEG-7;
- Pode refletir quaisquer futuras mudanças e extensões no esquema MPEG-7.

Não foi possível realizar uma análise desta ferramenta porque o fabricante do software não disponibiliza uma versão para avaliação, sendo necessário adquirir uma licença de uso. Dentre as ferramentas relacionadas, teoricamente esta foi a que mais despertou nosso interesse em realizar a geração de conteúdo audiovisual no formato MPEG-7.

A.3 – VIDETO

O *Vídeo Description Tool* (VIDETO, 2003) é uma ferramenta para anotação de vídeo de fácil uso, que permite uma fácil geração de descrições de vídeo para aplicações específicas.

VIDETO foi projetado para facilitar a criação de descrições MPEG-7, embora suporte diferentes formatos de saída. As principais características do VIDETO são:

- Adaptável a diferentes cenários de aplicações e formatos de saída, utilizando para isso *templates*[40] personalizados;
- Pacote de descrições livremente personalizável para ser usado no processo de anotação;
- Detecção automática de cenas e extração de *keyframe*, com opção de correção e verificação manual;
- Interface amigável para a anotação e revisão do vídeo ou seus segmentos (figura 28);

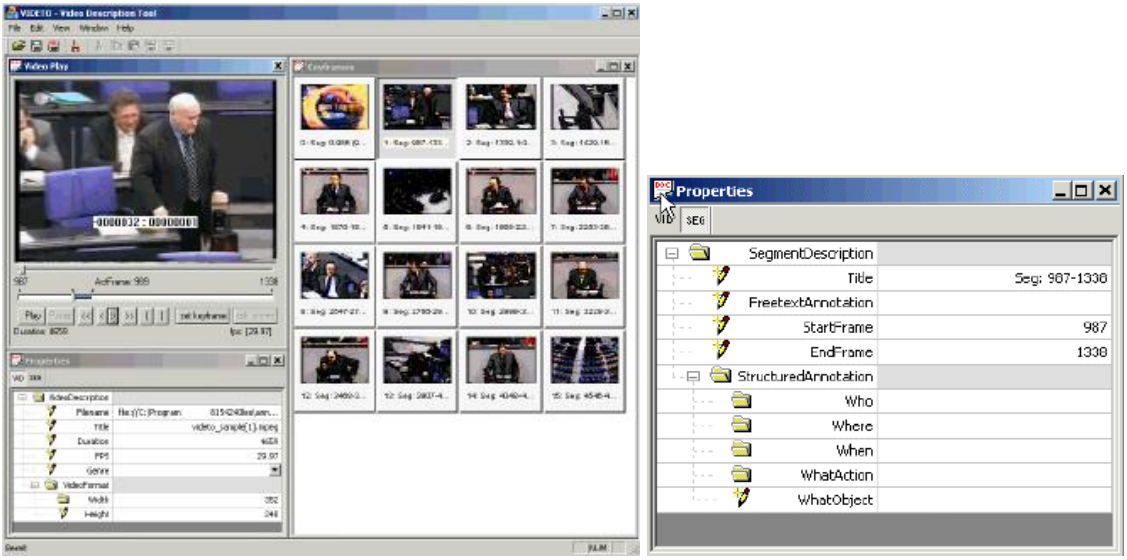

Figura 28. Interface do Videto (esquerda) e janela para descrição de segmento audiovisual

Considerações Finais

As vantagens do *VideoAnnEx Annotation Tool* são: capacidade de trabalhar com arquivos no formato MPEG-1 e MPEG-2 (embora a sustentação ao MPEG-2 apresente alguns problemas), e suporte para anotação de regiões[41], características estas não encontradas nas outras ferramentas analisadas. Como desvantagem, a ferramenta não realiza uma segmentação de vídeo multi-nível (segmentação de vídeo em cenas e *shots*).

[40] Modelos usados como base para novos documentos.

[41] Funcionalidade que permite a associação de uma região retangular com uma anotação de texto rotulada.

O *MovieTool* possui um protótipo mais completo, mas o acesso às suas funcionalidades não pode ser realizado na prática e, conseqüentemente, não pudemos expressar nossas conclusões.

O VIDETO esconde a complexidade do MPEG-7 de forma hábil, fundamentando as propriedades de descrição em um simples *template,* que pode ser mapeado para MPEG-7 usando XSLT[42] (*Extensible Stylesheet Language Transformations*) (W3C, 1999a). Sua interface é bastante simples e de fácil entendimento. Como desvantagem, VIDETO não disponibiliza segmentação hierárquica.

[42] Linguagem projetada para transformar a estrutura de um documento XML em outro documento com formato de saída diferente (ex: HTML, PDF), separando o conteúdo da apresentação.

Apêndice B

Esquema de Descrição da VideoLib (XML Schema)

A validação das descrições audiovisuais da VideoLib é realizada por meio de três arquivos implementados através da linguagem XMLSchema. Embora MPEG-7 também utilize XMLSchema, sua DDL não vai ser mostrada nesse apêndice devido à sua extensão e pelo motivo de que apenas alguns de seus DSs terem sido usados no modelo em questão. Para se ter acesso aos metadados MPEG-7, a VideoLib realiza uma importação do esquema MPEG-7 para poder ter acesso aos seus elementos de metadados.

- **dc.xsd**

Arquivo que modela os elementos semânticos do padrão Dublin Core, que deverão ser importados pelo arquivo **videolib.xsd.**

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://purl.org/dc/elements/1.1"
        xmlns:dc="http://purl.org/dc/elements/1.1"
        xmlns="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="unqualified"
        attributeFormDefault="unqualified">

      <simpleType name="title">
            <restriction base="string" />
      </simpleType>

      <simpleType name="creator">
            <restriction base="string" />
      </simpleType>

      <simpleType name="subject">
            <restriction base="string" />
      </simpleType>

      <simpleType name="description">
            <restriction base="string" />
      </simpleType>

      <simpleType name="publisher">
            <restriction base="string" />
      </simpleType>

      <simpleType name="contributor">
            <restriction base="string" />
      </simpleType>
```

```
      <simpleType name="date">
            <restriction base="string" />
      </simpleType>

      <simpleType name="type">
            <restriction base="string" />
      </simpleType>

      <simpleType name="format">
            <restriction base="string" />
      </simpleType>

      <simpleType name="identifier">
            <restriction base="string" />
      </simpleType>

      <simpleType name="source">
            <restriction base="string" />

      </simpleType>

      <simpleType name="language">
            <restriction base="string" />
      </simpleType>

      <simpleType name="relation">
            <restriction base="string" />
      </simpleType>

      <simpleType name="coverage">
            <restriction base="string" />
      </simpleType>

      <simpleType name="rights">
            <restriction base="string" />
      </simpleType>


      <element name="title" type="dc:title" />
      <element name="creator" type="dc:creator" />
      <element name="subject" type="dc:subject" />
      <element name="description" type="dc:description" />
      <element name="publisher" type="dc:publisher" />
      <element name="contributor" type="dc:contributor" />
      <element name="date" type="dc:date" />
      <element name="type" type="dc:type" />
      <element name="format" type="dc:format" />
      <element name="identifier" type="dc:identifier" />
      <element name="source" type="dc:source" />
      <element name="language" type="dc:language" />
      <element name="relation" type="dc:relation" />
      <element name="coverage" type="dc:coverage" />
      <element name="rights" type="dc:rights" />

</schema>
```

- **videolib.xsd**

Este arquivo define o modelo de descrição da VideoLib. Internamente, videolib.xsd importa os metadados Dublin Core (dc.xsd) e MPEG-7 (Mpeg7-2001.xsd), usando a abordagem de *namespaces*[43].

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:dc="http://purl.org/dc/elements/1.1"
      xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
      elementFormDefault="qualified"
      attributeFormDefault="unqualified">

      <xs:import namespace="http://purl.org/dc/elements/1.1"
schemaLocation="dc.xsd"/>
      <xs:import namespace="urn:mpeg:mpeg7:schema:2001"
schemaLocation="Mpeg7-2001.xsd"/>

      <xs:element name="videolib">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element ref="dc:title" minOccurs="0" />
                        <xs:element ref="dc:creator" minOccurs="0" />
                        <xs:element ref="dc:subject" minOccurs="0"/>
                        <xs:element ref="dc:description" minOccurs="0" />
                        <xs:element ref="dc:publisher" minOccurs="0"/>
                        <xs:element ref="dc:contributor" minOccurs="0"/>
                        <xs:element ref="dc:date" minOccurs="0"/>
                        <xs:element ref="dc:type" minOccurs="0"/>
                        <xs:element ref="dc:format" minOccurs="0"/>
                        <xs:element ref="dc:identifier" minOccurs="0"/>
                        <xs:element ref="dc:source" minOccurs="0"/>
                        <xs:element ref="dc:language" minOccurs="0"/>
                        <xs:element ref="dc:relation" minOccurs="0"/>
                        <xs:element ref="dc:coverage" minOccurs="0"/>
                        <xs:element ref="dc:right" minOccurs="0"/>

                        <!-- Metadados MPEG-7 -->
                        <xs:element name="MediaLocator"
type="mpeg7:MediaLocatorType" minOccurs="1" maxOccurs="1" />
                        <xs:element name="MediaTime"
type="mpeg7:MediaTimeType" minOccurs="0" maxOccurs="1" />
                        <xs:element name="MediaFormat"
type="mpeg7:MediaFormatType" minOccurs="0" maxOccurs="1" />
                        <xs:element name="TemporalDecomposition"
type="mpeg7:VideoSegmentTemporalDecompositionType" minOccurs="1"
maxOccurs="1" />
                  </xs:sequence>
            </xs:complexType>
      </xs:element>
</xs:schema>
```

[43] prefixos que têm o objetivo de evitar a colisão entre diferentes elementos de mesmo nome em um documento XML.

Apêndice C

Dublin Core

O Dublin Core Metadata Initiative é uma organização que direciona seus esforços na adoção e difusão de um padrão de metadados que proporcione interoperabilidade, e no desenvolvimento de um vocabulário especializado de metadados para a descrição de recursos. A criação de um consenso internacional em torno da centralização deste conjunto de elementos é a principal característica da evolução do Dublin Core (DC, 2003).

As características do Dublin Core (DC) que o distinguem como um candidato proeminente para a descrição de recursos eletrônicos, se encaixa nas seguintes categorias:

- **Simplicidade de criação e manutenção**: DC foi planejado para ser utilizado tanto por não-catalogadores, quanto por especialistas na descrição de recursos. Muitos dos elementos possuem uma semântica de fácil entendimento;
- **Interoperabilidade semântica**: promovendo um conjunto de descritores de entendimento geral e que ajudam a unificar outros padrões de conteúdo de dados, a possibilidade de interoperabilidade semântica é aumentada;
- **Consenso Internacional**: DC foi beneficiado com a promoção e participação ativa de diversos países, tais como Inglaterra, Estados Unidos, Suécia, Alemanha e França. O reconhecimento internacional contribui para a aceitação do padrão;
- **Extensibilidade**: o modelo permite que diferentes comunidades de metadados utilizem DC como o centro de suas informações descritivas, possibilitando a adição de novos elementos para cobrir um domínio específico. DC inclui flexibilidade e extensibilidade suficientes para codificar a estrutura adicional e elaborar a semântica apropriada para domínios de descrição de recursos mais formais.

O padrão DC fornece um modelo de metadados composto por 15 elementos básicos que foram projetados com a intenção de facilitar a descoberta de recursos eletrônicos pelas aplicações. Atualmente na versão 1.1, a definição dos elementos DC auxilia no aperfeiçoamento da consistência com outras comunidades de metadados. Cada definição DC refere-se ao recurso (qualquer coisa que tenha uma identidade) que está sendo descrito. Todos

os elementos são opcionais e com ocorrência ilimitada. Cada elemento pode ser definido semanticamente da seguinte maneira:

- **title**: um nome pelo qual o recurso é formalmente conhecido;
- **creator**: pessoa, organização ou serviço responsável pela criação do conteúdo do recurso;
- **subject**: o tópico do conteúdo do recurso. Pode ser expresso como palavras chaves, frases ou códigos de classificação que descrevem o tópico do recurso;
- **description**: um relato sobre o conteúdo do recurso. A descrição pode incluir (mas não é limitado) um abstract, índice ou referência para uma representação gráfica de um conteúdo;
- **publisher**: a entidade responsável pela disponibilização do recurso. Exemplos de publisher podem englobar uma pessoa, uma organização ou um serviço;
- **contributor**: a entidade (pessoa, organização ou serviço) que originou contribuições para a produção do conteúdo do recurso.
- **date**: data associada com a criação ou disponibilidade do recurso. Uma boa prática recomendada para a representação de datas segue o formato YYYY-MM-DD, seguindo o perfil ISO 8601 (W3C 1997);
- **type**: natureza ou gênero do conteúdo do recurso, tais como texto, imagem, som, vídeo, etc. Inclui termos que descrevem categorias gerais, funções, gênero ou níveis de agregação para o conteúdo;
- **format**: representação física ou digital do recurso. Este elemento pode englobar o tipo da mídia ou dimensões do recurso. Pode ser usado para identificar o software, hardware ou outro equipamento necessário para manipular o recurso;
- **identifier**: uma referência não ambígua para o recurso dentro de um determinado contexto. Recomenda-se identificar o recurso por meio de uma string ou número, conforme uma identificação formal de sistema, como por exemplo, uma URL, ISBN;
- **source**: informação sobre um segundo recurso pelo qual o presente recurso é derivado, seja na totalidade ou em parte, com representação sob a forma de uma string ou número;
- **language**: a linguagem do conteúdo intelectual do recurso. Recomenda-se a utilização de duas letras para representar o código da linguagem, seguido

opcionalmente por mais duas letras para representação do código do país. Exemplos: 'en' para inglês, 'fr' para francês ou 'en-uk' para inglês do Reino Unido.

- **relation**: referência com outros recursos. Permite estabelecer um link com um recurso relacionado.
- **coverage**: a extensão ou escopo do conteúdo do recurso. A cobertura poderá incluir localizações espaciais (nome de um lugar ou coordenadas geográficas), relação temporal (um rótulo de período, data, faixa de data) ou jurisdição (como é chamada uma entidade administrativa);
- **right**: informações sobre direitos assegurados ao recurso, caso o elemento venha a ser utilizado. Estes direitos freqüentemente representam Direitos de Propriedade Intelectual, copyright e vários outros direitos de propriedade.

Apêndice D

XQuery – Uma linguagem de consulta a dados XML

Um dos grandes pontos fortes do XML é a sua flexibilidade em representar diferentes tipos de informações de diversos domínios. Para explorar esta flexibilidade, uma linguagem de consultas em XML deve conceder características para recuperar e interpretar informações destas fontes diversificadas. Diante das novas tendências, a W3C[44] (*World Wide Web Consortium*) projetou **XQuery** (W3C, 2003) para ser uma linguagem que pudesse realizar consultas precisas e de fácil entendimento.

A modelagem de dados utilizada por XQuery é baseada em XPath (W3C, 1999b), linguagem esta que fornece uma notação para a seleção de elementos em um documento XML, estruturalmente similar a uma árvore constituída por nodos.

Por ser uma linguagem funcional[45], XQuery fornece vários tipos de expressões que podem ser organizadas e combinadas utilizando palavras-chaves, símbolos e operandos. Todas as palavras reservadas definidas em XQuery utilizam caracteres em letra minúsculas.

Sete tipos de expressões são definidas em XQuery, podendo ser agrupadas em conjunto, de forma seqüencial e aninhada. Para exemplificar cada uma destas expressões, tomaremos como base o arquivo XML ilustrado na figura 29.

```
<veiculos>
    <automovel tipo="popular" cilindradas="1000">
          <modelo>Palio</modelo>
          <acessorios>trava, aerofolio</acessorios>
          <fabricante>FIAT</fabricante>
          <ano>2001</ano>
          <cor>AZUL</cor>
          <preco>9000.80</preço>
    </automovel>
    <automovel tipo="luxo" cilindradas="2000">
          <modelo>Marea</modelo>
          <acessorios>ar-condicionado, vidro, alarme</acessorios>
          <fabricante>FIAT</fabricante>
          <ano>2002</ano>
```

[44] http://www.w3.org

[45] Linguagem onde cada consulta é uma expressão.

```
            <cor>PRATA</cor>
            <preco>34500.00</preco>
      </automovel>
      <automovel tipo="luxo" cilindradas="1500">
            <modelo>Uno XR</modelo>
            <acessorios>painel digital, cd, alarme</acessorios>
            <fabricante>FIAT</fabricante>
            <ano>1997</ano>
            <cor>CINZA</cor>
            <preco>6500.00</preco>
      </automovel>
  </veiculos>
```

Figura 29. Conteúdo do arquivo "veiculos.xml"

a) Path expressions

Uma *path expression* é utilizada para localizar nodos dentro de uma árvore XML, baseado na sintaxe do XPath.

Exemplo 1: Encontre todos os modelos de automóveis no documento veiculos.xml.

```
document("veiculos.xml")/veiculos/automoveis/modelo
```

O elemento "modelo" se encontra no terceiro nível do documento XML, portanto, torna-se necessário programar o acesso iniciando pelo nodo raiz do documento, correspondente ao elemento "veiculos", até chegar ao elemento de interesse.

Exemplo 2: Encontre todos os automóveis no documento veiculos.xml fabricados em 2001 do tipo "popular":

```
document("veiculos.xml")/veiculos/automoveis[ano=2001
                              AND @tipo="popular"]
```

A palavra reservada *document* é uma função de entrada que recebe como parâmetro uma *string* correspondente a URI de um documento XML, convertendo tal documento para uma representação de modelo de dados que retorne seu nodo correspondente.

b) Construtor de elementos

Tipo de expressão usado quando uma consulta necessita criar novos elementos XML. Em um construtor de elementos, o nome usado na *tag* finalizadora deve combinar com o nome da *tag* de início correspondente.
Exemplo: Gerar um elemento automóvel com atributo "tipo", cujo elemento filho é o modelo do veículo.

```
<automovel>
    { $b/@tipo }
    { $b/modelo }
</automovel>
```

$b é uma variável nomeada, definida em outra parte da consulta, vinculada aos valores de um nodo. As chaves delimitadoras demarcam "expressões fechadas", distinguindo-as de um texto literal. Quando a expressão completa é executada, o elemento construtor descrito acima produzirá um resultado semelhante a:

```
<automovel tipo="popular">
    <modelo>Uno XR</modelo>
</automovel>
```

c) Expressões FLWR

XQuery fornece uma característica chamada *FLWR expression* (pronunciado como "*flower*") que suporta iteração e ligação de variáveis para resultados intermediários. Esta expressão é freqüentemente útil para computar junções entre dois ou mais documentos. A expressão FLWR é análoga à construção SELECT-FROM-WHERE em SQL, formando a estrutura da expressão XQuery. Consiste das palavras-chave *for, let, where* e *return* (*For*, *Let, Where*, *Return*), podendo possuir múltiplas cláusulas *for* e *let.*

As cláusulas de uma expressão FLWR são definidas a seguir:

- ***for***: liga uma ou mais variáveis nomeadas a uma seqüência de valores retornada por outra expressão (geralmente uma expressão *path*), com a propriedade de poder interagir com esses valores, unindo a variável a cada item por vez;

- ***let:*** também pode conter múltiplas variáveis, cada uma com uma expressão associada, mas sem interação;
- ***where:*** atua como um filtro para o conjunto de nodos geradas pelas cláusulas *for/let;*
- ***return:*** produz a saída de uma expressão FLWR, geralmente contendo um ou mais elementos construtores.

Exemplo: Listar todos os automóveis fabricados pela FIAT depois do ano 2000, incluindo seu modelo e preço de venda.

```
<veiculos>
 {
   for $b in document ("veiculos.xml")/veiculos/automovel
   where $b/fabricante = "FIAT" and $b/ano > 2000
   return
     <automovel ano ="{ $b/ano }">
          { $b/modelo }
          { $b/preco  }
     </automovel>
 }
</veiculos>
```

d) Expressões que envolvem operadores e funções

XQuery provê muitos operadores e funções que também estão presentes em outras linguagens de consulta, incluindo operadores aritméticos, comparativos e lógicos. As funções embutidas incluem *avg*, *sum*, *count*, *max* e *min*, mas também funções relacionadas com nodos e documento XML, como *document*, *empty* e *distinct*. O seguinte exemplo lista todos os automóveis cujo preço é maior que a média de preços existente no documento.

```
for $b in document("veiculos.xml")//automovel
let $a = avg document("veiculos.xml")/veiculos/automovel/preco
where $b/price > $a
return $b
```

e) Expressões condicionais

XQuery suporta o uso de expressões condicionais baseada nas palavras-chaves *if, then* e *else.*

Exemplo: Para todos os automóveis contidos no arquivo "veiculos.xml", se o tipo for "popular", exibir a cor. Caso contrário, exibir o preço.

```
for $b in document("veiculos.xml")//automovel
return <automovel> if $b/@tipo = "popular" then $b/cor
                                           else $b/preco
</automovel>
```

f) Expressões quantificadas

Expressões quantificadas suportam quantificação existencial (*some*) e universal (*any*). O valor de uma expressão quantificada é sempre *true* ou *false.* Através da expressão *some* é possível identificar se pelo menos um nodo de um conjunto de nodos satisfaz o predicado. A expressão *every* é usada para testar se todos os nodos de um conjunto satisfazem um predicado. As duas expressões (*some* e *every*) são seguidas pela palavra chave *satisfies*.

Exemplo:

```
for $b in document("veiculos.xml")//automovel
where some $a in $b/acessorios satisfies contains($a,"alarme")
AND   contains($a,"ar-condicionado")
return $b/modelo
```

g) Expressões que testam ou modificam tipos

XQuery suporta tipos de dados padrões (baseado no sistema de tipos do XML Schema), como também tipos de dados definidos pelo usuário. As expressões *instanceof* e *typeswitch/case* são utilizadas para testar se uma instância é de um certo tipo de dado.

Exemplo:

```
5 instance of xs:integer
  (retorna true porque o determinado valor é uma instância do
  tipo xs:integer)
typeswitch ($automovel/origem)
  case $a as element of type SaoPauloOrigem return $a/bairro
  case $a as element of type CearaOrigem return $a/cidade
default return "desconhecido"
```

Apêndice E

Padrão MPEG-7

O conteúdo de um arquivo digital de vídeo reserva um grau de interpretação que deve ser passado ou acessado por um dispositivo eletrônico ou programa de computador (NACK; LINDSAY, 1999). MPEG-7 tem o objetivo de especificar um padrão para descrever e suportar estes requisitos operacionais.

MPEG-7 é um padrão de metadados desenvolvido pela MPEG (*Moving Picture Expert Group*) para descrição de conteúdo audiovisual. Diferentemente dos precedentes padrões MPEG (MPEG-1, MPEG-2 e MPEG-4), que lidam diretamente com a representação codificada do material multimídia, MPEG-7 focaliza suas atenções na representação de informações sobre o conteúdo em diferentes níveis, tais como:

- Estrutural: por exemplo, um vídeo consiste de uma seqüência de segmentos e cada segmento é composto por vários *shots*;
- Temporal: por exemplo, um segmento de vídeo é demarcado pelo seu instante inicial e final em relação à duração do seu conteúdo;
- Visual: por exemplo, um objeto tem a forma de uma estrela;
- Semântico: por exemplo: Romário conversa com Rivaldo na Itália;

MPEG-7 pode ser visto como uma forma de se anexar metadados ao conteúdo audiovisual, especificando um conjunto de ferramentas de descrição que podem ser usadas para descrever vários tipos de informações multimídia. Estas ferramentas podem ser associadas ao próprio conteúdo, para permitir uma eficiente busca por material multimídia de interesse do usuário (HUNTER, 1999). Suas anotações podem ser realizadas em diferentes tipos de mídia incluindo: fotos, áudio, vídeo, modelos 3D, gráficos e composição de informações sobre como tais elementos audiovisuais podem ser combinados em um cenário multimídia. A aplicação de metadados MPEG-7 pode ser empregada para a descrição de *streams* MPEG-4, vídeo tape, CD contendo músicas, gravura impressa no papel ou em aplicações interativas multimídia instaladas na Web.

As descrições MPEG-7 podem ser instanciadas de duas maneiras: a primeira, em formato textual através da linguagem XML, adequada para edição, busca, filtragem e *browsing;* a segunda, por uma representação binária apropriada para armazenamento, transmissão e *streaming* (SALEMBIER, 2002).

O padrão MPEG-7 não está limitado a uma classe de aplicação em particular ou conteúdo multimídia, pois não depende do modo como o conteúdo é codificado ou armazenado. Qualquer programa que faça uso de suas descrições não se encontra especificado dentro de seu escopo; cabe às empresas comerciais desenvolverem suas próprias soluções, estimulando a competição entre elas, visando obter os melhore resultados. MPEG-7 é independente dos outros padrões MPEG (como o MPEG-4), podendo ser utilizado para descrever objetos em diversos formatos, como AVI, MOV, WAV, etc. (NACK; LINDSAY, 1999).

O que diferencia MPEG-7 de outros padrões de metadados relacionados é a sua capacidade de representar vários níveis de abstração, desde características de baixo nível, até informações semânticas de alto nível (MARTÍNEZ *et al*, 2002). Por exemplo, para conteúdo visual, um baixo nível de abstração poderia ser descrito pela sua forma, cor, textura e trajetória. Um alto nível de abstração concederia informação semântica, tal como uma cena em que houvesse um gato em cima de uma bola, com um som de trem passando ao fundo.

Os principais propósitos do MPEG-7 são: a) alcançar a máxima interoperabilidade, b) padronizar a descrição de vários tipos de informação multimídia e, c) facilitar o surgimento de aplicações inovadoras, provendo um rico conjunto de ferramentas padronizadas que possibilitem o entendimento de descrições audiovisuais entre humanos e máquinas, permitindo uma recuperação de arquivos digitais de maneira rápida e eficiente (PEREIRA, 1997). MPEG-7 fará com que a Web seja pesquisada por conteúdo multimídia, como hoje em dia é pesquisada por texto.

Muitos domínios de aplicações serão beneficiados diretamente com o padrão MPEG-7. Exemplos de aplicações incluem:

- Bibliotecas Digitais (catálogo de imagens, acervo de vídeo);
- Edição multimídia (serviços de notícias eletronicamente personalizados);
- Seleção de mídia de transmissão (canal de rádio, canal de TV);
- Educação (repositório de cursos multimídia, busca multimídia por material de suporte);
- Jornalismo (busca por um discurso de um determinado político utilizando seu nome, sua voz ou imagem de sua face).

Principais Componentes do Padrão MPEG-7

Para atingir os seus objetivos, O padrão MPEG-7 especifica os seguintes elementos normativos: *Descriptors* (Ds), *Description Scheme* (DSs), *Description Definition Language* (DDL) e Ferramentas de Sistema (SALEMBIER, 2002).

Descriptors (Ds)

Descriptors são elementos de metadados que definem a sintaxe e a semântica para a representação de características. São idealizados para descrever propriedades de baixo nível audiovisual. A cor, textura e forma de uma imagem são exemplos destas características.

Muitas características de baixo nível podem ser extraídas do conteúdo audiovisual de forma automática pelas aplicações. Algoritmos para extração de características são incluídos na parte não-normativa de alguns descritores, com o propósito de estimular a competição comercial e levar vantagem do aperfeiçoamento da tecnologia.

Description Schemes (DSs)

Um *Description Scheme* especifica a estrutura sintática e semântica do relacionamento entre seus componentes, que podem ser Ds ou DSs. *Description Schemes* foram idealizados para descrever características audiovisuais de alto nível, por exemplo, regiões, segmentos e eventos. São mais complexos porque produzem descrições que integram múltiplos Ds e DSs.

Descrições MPEG-7 – um conjunto de Description Schemes instanciados – precisam ser ligadas ao próprio conteúdo para permitir uma eficiente busca por material audiovisual de interesse do usuário.

Description Definition Language (DDL)

A DDL do MPEG-7 é uma linguagem de esquema que permite definir as regras sintáticas de novos Ds e DSs, como também estender ou modificar Ds e DSs existentes (ISO/IEC, 2002; HUNTER, 2001). Através da DDL são especificados os descritores que podem ser usados em uma descrição e estabelecido o relacionamento entre eles ou entre outros DSs, permitindo a representação de hierarquias complexas.

MPEG-7 DDL é derivada por extensão da linguagem XML Schema (W3C, 2001) e utiliza XML (eXtensible Markup Language) (BRAY *et al*, 2000) para definição sintática de sua descrição textual. A DDL forma o núcleo do padrão MPEG-7.

O resultado da DDL satisfaz os seguintes requisitos chaves para o MPEG-7: definição de tipos, declaração de Ds e DSs, declaração de atributos, modelagem de conteúdo, referência de tipos, herança e inclusão de novos DSs (ISO/IEC, 2002).

MPEG-7 foi idealizada de modo que fosse extensível por meio de sua DDL. Assim os desenvolvedores de aplicações poderiam estender os DSs padrões para atender as suas necessidades.

Relacionamento entre os elementos MPEG-7

MPEG-7 fornece um conjunto de ferramentas para descrição audiovisual, composta por elementos de metadados, estruturas e relacionamentos, definidos sob a forma de *Descriptors* e *Description Schemes*. A figura 30 apresenta o relacionamento entre os principais elementos MPEG-7.

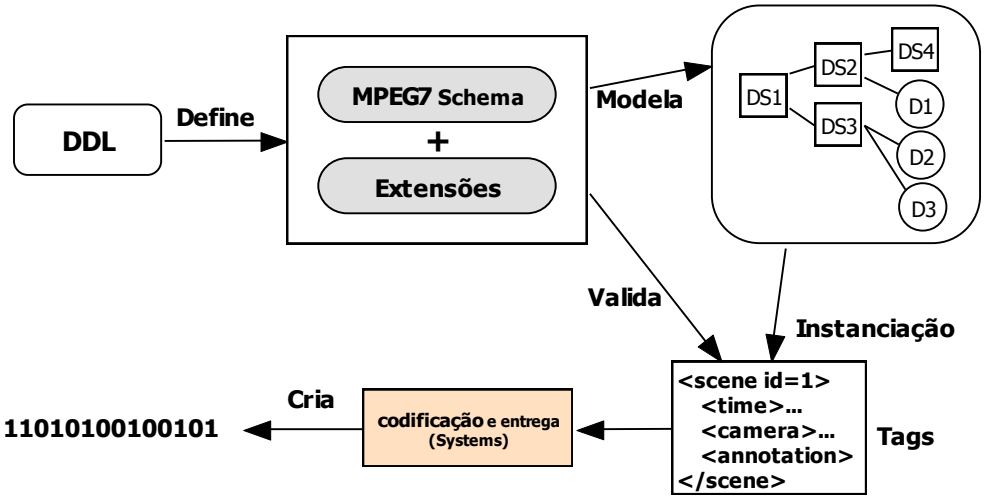

Figura 30. Principais elementos MPEG-7
Fonte: ISO/IEC JTC1/SC29/WG11

A definição da semântica de novos esquemas de descrição é feita utilizando a DDL. Quando um conjunto de Ds e DSs é instanciado para compor um fragmento de conteúdo audiovisual, a descrição resultante origina um documento XML, de acordo com as regras definidas pela DDL. O uso de XML é para facilitar a interoperabilidade entre aplicações

interessadas no conteúdo das descrições ou com futuros padrões (CHANG; SIKORA, 2001). O formato binário das descrições é obtido por meio do BiM (*Binary Format for MPEG-7*), um *framework* genérico que facilita o transporte e processamento de descrições MPEG-7, permitindo *streaming* e compressão de documentos XML.

Multimedia Description Schemes (MDS)

O MPEG-7 MDS consiste em estruturas de metadados para descrever e anotar conteúdo audiovisual (SALEMBIER; SMITH, 2001). Os *Descriptions Schemes* (DSs) fornecem uma forma padronizada de descrever em XML os importantes conceitos relacionados com a descrição e gerenciamento do conteúdo audiovisual a fim de facilitar a busca, indexação, filtragem e acesso. A Figura 31 ilustra a organização do MDS.

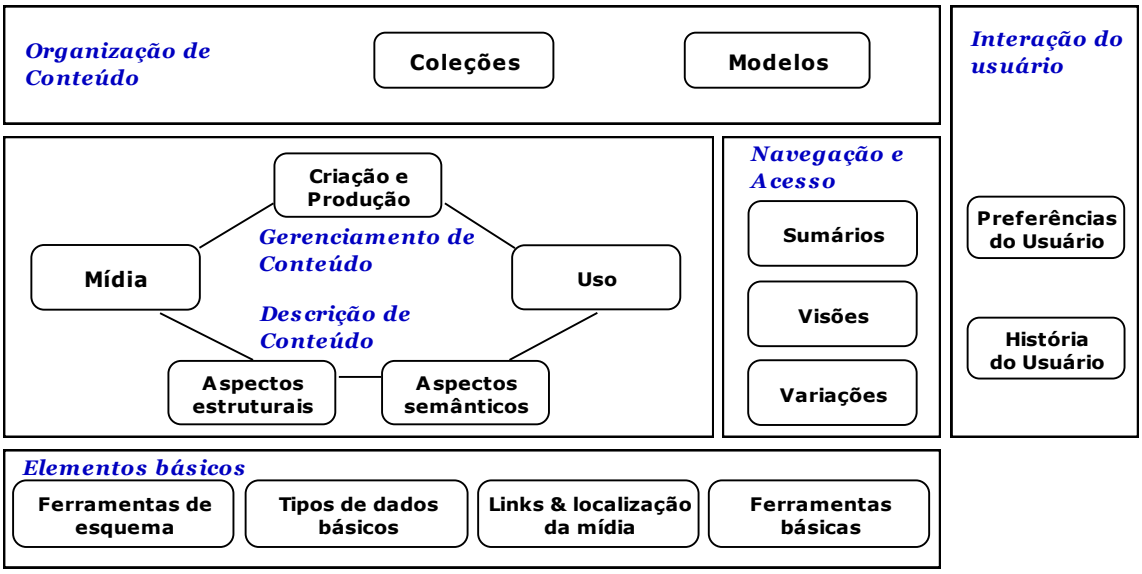

Figura 31. Visão geral do MPEG-7 MDS.
Fonte: ISO/IEC JTC1/SC29/WG11

Os significados de cada área funcional são os seguintes:

- Elementos básicos: possui um conjunto de ferramentas de esquemas que auxiliam na formação, empacotamento e anotação de descrições MPEG-7. Os metadados que se encaixam nesse grupo permitem descrever anotações temporais (Time DS) e textuais (TextAnnotation DS);

- Gerenciamento de conteúdo: Fornece DSs para o gerenciamento de conteúdo, cujas ferramentas descrevem informações sobre criação e produção, codificação da mídia, armazenamento, formato do arquivo e uso do conteúdo;
- Descrição de conteúdo: contém um conjunto de DSs para a descrição da estrutura e semântica do conteúdo audiovisual. As ferramentas estruturais descrevem a estrutura de um vídeo em termos de segmentos (segmentação em cenas e *shots), frames* e regiões em movimento. As ferramentas semânticas descrevem os objetos, eventos e noções do mundo real que podem ser abstraídos do conteúdo multimídia;
- Navegação e Acesso: suas ferramentas fornecem recursos que facilitam a navegação e acesso ao conteúdo audiovisual, disponibilizando metadados que permitem a descrição de sumários (o conteúdo pode ser navegado de forma hierárquica ou seqüencial) e variações do conteúdo codificado, incluindo mudanças de resolução ou modalidades (vídeo, áudio, imagem, texto, etc.). Variações do conteúdo audiovisual podem substituir o original, se necessário, para adaptar diferentes apresentações multimídia;
- Interação do usuário: os DSs que compõem esta área funcional têm a função de descrever as preferências do usuário com relação ao consumo do conteúdo audiovisual, ou seja, anotações podem ser unidas com descrições de preferência visando personalizar o conteúdo audiovisual para um eficiente acesso e apresentação;
- Organização do conteúdo: possui ferramentas destinadas à descrição de coleções do material audiovisual. As coleções podem ser utilizadas, por exemplo, para descrever um álbum de canções ou um grupo de objetos.

Os DSs/Ds MPEG-7 são categorizados em duas partes:

- Descrições (semi) automáticas: uso de ferramentas de segmentação visual. Estas ferramentas instanciam automaticamente os DSs de seu interesse, tais como *SegmentDS, MediaTimeDS,* etc.;
- Descrições manuais: entrada de texto manualmente. Por exemplo, utilização do *TextAnnotationDS, UsageInformationDS, CreationInformationDS,* etc.

Devido à complexidade e quantidade de DSs que compõem cada área funcional, nos detivemos aqui a uma breve descrição do escopo de cada área. Para um maior detalhamento, a referência oficial do padrão (ISO/IEC, 2002) deve ser consultada.