niversidade Estadual de Campinas
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE TELEMÁTICA

FEEC
Faculdade de Engenharia Elétrica e de Computação
UNICAMP

Uma Arquitetura para a Recuperação sobre Bancos de Dados de Imagens

Autor: Sahudy Montenegro González

Orientador: Prof. Dr. Akebo Yamakami

Banca: Prof. Dr. Rosane Minghim (ICMC/USP)

Prof. Dr. Ivan Luiz Marques Ricarte (FEEC/UNICAMP)

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos necessários para a obtenção do título de Mestre em Engenharia Elétrica.

DT/FEEC/UNICAMP
Campinas
Junho, 1999

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

M764a

Montenegro González, Sahudy
Uma arquitetura para recuperação sobre bancos de dados de imagens. / Sahudy Montenegro González.-- Campinas, SP: [s.n.], 1999.

Orientador: Akebo Yamakami
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Banco de dados relacionais. 2. Imagens – Interpretação. 3. Sistemas multimídia. I. Yamakami, Akebo. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Resumo

Muitos dos problemas na recuperação de imagens são originados pelas necessidades de domínio específico e por isto as soluções encontradas são limitadas somente a um conjunto de domínios de aplicação. A falta de modelos genéricos para dados e para recuperação de imagens traz, freqüentemente, grandes dificuldades aos pesquisadores. Como uma tentativa para solucionar esta situação, este trabalho define uma arquitetura independente de domínio das aplicações para a recuperação sobre bancos de dados de imagens: GRAID (*Generic Retrieval Architecture for Image Databases*). GRAID pretende ser genérica o suficiente para ser adaptável a qualquer domínio de aplicação. O propósito é fornecer uma plataforma comum para as aplicações de bancos de dados de imagens e a flexibilidade e a extensibilidade necessárias para que os desenvolvedores consigam implementar aplicações de maneira simples usando esta plataforma. A chave para construir uma arquitetura genérica e independente do domínio das aplicações é a separação da função de processamento das imagens da função de recuperação das mesmas. Para validar as idéias propostas foi implementado o sistema WIRED (*Web Image REtrieval from Databases*) e duas aplicações protótipos de recuperação de imagens.

Abstract

Most of the solutions proposed in image database applications are limited to a specific application domain. The lack of generic models frequently introduces troubles to reseachers. As an attempt to solve this situation, this work defines a Generic Retrieval Architecture for Image Databases (GRAID). GRAID is adaptable to many application domains. The main goal of this work is to provide an extensible and flexible common platform to ease the work of the image application programmer. The key to construct a generic domain independent architecture is to separate the image retrieval from the image processing functions. To validate the GRAID architecture, the WIRED (Web Image REtrieval from Databases) system was implemented. Finally, two prototype image database applications based on the WIRED system are briefly described.

Agradecimentos

Ao meu esposo *Eduardo*, por ser meu suporte emocional, profissional e incondicional.

Ao meu amigo *Luis Mariano* por sua insuperável ajuda na elaboração deste trabalho.

Ao meu orientador, *Akebo*, porque sem sua aceitação no curso de mestrado nada tivesse acontecido.

A todos os que, de uma forma ou de outra, contribuíram com este trabalho.

Amigo puede ser quien bien repara
en la musa o engendro que yo aporte;
amigo, sí, es también quien me soporte,
pero amigo mayor es quien me ampara.
...

Amigo Mayor, Silvio Rodríguez

... En busca de un sueño
hermoso y rebelde
En busca de un sueño
que gana y que pierde ...

En busca de un sueño, Silvio Rodríguez

Aos meus pais e avó
Aos meus Montenegro e González

Índice

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Muitas pesquisas estão sendo encaminhadas no sentido de minimizar os problemas que o uso das imagens introduz em aplicações computacionais. As imagens demandam um grande espaço de armazenamento e processadores rápidos para seu gerenciamento. Com o rápido desenvolvimento da tecnologia VLSI (*Very Large Scale Integration*) e o surgimento de vários tipos de meios de armazenamento físico, tanto a velocidade dos processadores quanto a capacidade de armazenamento continuam prosperando. Como resultado disso e da grande necessidade de projetar aplicações de imagens nas mais diversas áreas, os Bancos de Dados de Imagens (BDIs) são cada vez mais comuns. Dentre as áreas de aplicação onde a recuperação de imagens é fundamental podem-se citar: Sistemas de Informação Geográficos (SIGs) [33, 40], Sistemas de Informações Médicas (SIMs) [19, 38], Sistemas de Informação em Escritórios [1, 29] e Sistemas de Bibliotecas Digitais Históricas e para o gerenciamento de Museus e Galerias de Arte [4, 30, 31].

Como conseqüência da grande variedade de áreas de aplicação das imagens, as aplicações de BDIs são desenvolvidas com considerações específicas ao seu domínio [14]. A natureza das aplicações e, portanto, as necessidades de recuperação e as estratégias de processamento das imagens mudam de uma aplicação para outra.

Os *Sistemas de Recuperação de Imagens* (SRIs) são sistemas cuja principal operação de manipulação dos dados imagens é a recuperação. Os *sistemas de recuperação de imagens baseada no conteúdo* são SRIs que analisam as características do conteúdo das imagens para estabelecer comparações de similaridade entre imagens. Normalmente, estes sistemas desenvolvem a recuperação de imagens baseado no conteúdo através das técnicas de processamento de imagens mais próximas aos seus requerimentos de recuperação.

Por exemplo, CHABOT [24] foi projetado para manipular um banco de dados de imagens com aproximadamente 500,000 imagens. Estas imagens pertencem a uma grande coleção do *Department of Water Resources* (DWR) no Estado da Califórnia, EUA, com paisagens de lagos, flores, pôr do sol, etc. CHABOT integra sistemas de bancos de dados relacionais com técnicas de processamento das cores no conteúdo das imagens.

Em [33], os autores explicam uma abordagem e seus resultados experimentais para a recuperação de imagens em sistemas de bancos de dados de imagens de sensoriamento remoto. Para isto introduziram uma organização em grupos das imagens do banco de dados. A organização é baseada nas similaridades destas imagens com um conjunto de imagens predefinidas. A similaridade foi baseada no processamento das características da textura das imagens. A primeira fase em uma recuperação é feita sobre o conjunto de imagens

predefinidas o que minimiza o número de buscas no banco de dados. Os grupos são criados em dependência do tipo de aplicação por um especialista da área.

O sistema QBIC [8, 17] é um sistema de recuperação de imagens baseado no conteúdo. O sistema oferece consultas sobre bancos de dados de imagens baseadas em imagens exemplos, desenhos definidos pelos usuários e padrões de texturas, cores e formas. Para isto, a equipe desenvolveu os algoritmos de processamento de imagens correspondentes.

O sistema PICTION [35] usa os subtítulos das fotografias para identificar faces nas imagens. As técnicas de análise do conteúdo empregadas são o reconhecimento de faces humanas e o processamento da linguagem natural.

Note que nas últimas três abordagens não há suporte para os diversos tipos de dados tradicionais. Muitos dos sistemas de recuperação de imagens baseada no conteúdo não fornecem mecanismos para integrar a recuperação das características das imagens com os diversos tipos de dados tradicionais.

Observa-se como as aplicações descritas acima precisaram desenvolver um conjunto de ferramentas para a recuperação dependentes do seu domínio de aplicação.

Por essas razões, não existe consenso no que realmente possa ser um Sistema de Gerenciamento de Bancos de Dados de Imagens (SGBDI) [14]. Um verdadeiro SGBDI deve incluir os dados tradicionais e suportar o conjunto de funcionalidades que faz das imagens mais um tipo de dado nativo no sistema. Assim, as operações sobre todos os tipos devem ser aplicadas de maneira uniforme e integrada. O desenvolvimento de sistemas de gerenciamento de imagens, como de qualquer outro objeto multimídia, enfrenta as dificuldades no armazenamento, apresentação e principalmente na interpretação, indexação e recuperação das imagens. Um dos desafios atuais é incluir nos SGBDs a capacidade de manipular as imagens (e dados multimída em geral) de uma maneira integrada e eficiente.

Para compreender o propósito dos SGBDIs é preciso separar as operações sobre as imagens que são independentes das operações dependentes do domínio de aplicação. Esta separação permite uma definição mais clara dos componentes de uma aplicação e em conseqüência, pode permitir a reutilização por várias aplicações. Isto facilita o desenvolvimento de aplicações.

1.1 Sistemas de Gerenciamento de Bancos de Dados

Os Sistemas de Gerenciamento de Bancos de Dados (SGBDs) são altamente recomendados para serem usados como base para aplicações que precisam trabalhar com grande quantidade de dados [23]. Estes sistemas oferecem independência de dados e um repositório consistente de dados para ser compartilhado por várias aplicações. Os dois modelos de Bancos de Dados (BDs) mais conhecidos são o Orientado a Objetos e o Relacional.

O primeiro modelo suporta relações complexas no banco de dados, permite definições mais ricas sobre o formato e semântica dos dados que pode explorar na hora da manipulação para otimizar suas funcionalidades. As técnicas de definição de classes, encapsulamento e herança provêem ao SGBD uma melhor compreensão dos dados e uma manipulação mais eficientemente dos dados não estruturados. Apesar disso, ele não oferece uma solução completa para a gerência dos dados, pois não consegue uma boa interpretação da informação multimídia e não resolve satisfatoriamente problemas como a indexação e recuperação [23].

Um dos pontos fortes do segundo modelo é o fato de ser baseado em um modelo matemático: a Álgebra Relacional [27]. Embora as estruturas desenvolvidas nesses sistemas

não sejam completamente adequadas para as imagens, eles são os sistemas mais utilizados e são usadas como base de pesquisa para novos modelos. Além disso, é muito importante para a maioria das aplicações não perder a integração com os dados tradicionais, mesmo que as imagens sejam o recurso principal nestes sistemas. Com este propósito, o modelo relacional foi estendido para suportar novos tipos de dados como os BLOBs (*Binary Large Objects*). Este recurso oferece uma mínima integração dos dados tradicionais com objetos (dados complexos) que nunca puderam ser representados no modelo relacional tais como os dados multimídia (imagem, audio, vídeo e texto). No entanto, esta extensão não fornece nenhuma capacidade ao SGBD para manipular e interpretar o conteúdo destes objetos.

Devido às operações e as representações das imagens mudarem de uma aplicação para outra, cada aplicação escolhe a abordagem que mais eficientemente suporte as suas operações. Isto atenta contra a construção de sistemas de BDIs de propósito geral baseados nos modelos de dados convencionais por causa do seu limitado conjunto de tipos de dados.

Como uma solução a este problema, introduziram-se os SGBDs extensíveis. Outros tipos de extensões foram feitas ao núcleo de um SGBD tais como: extensões à linguagem de consulta, capacidade para a criação de operadores e funções definidas pelos usuários. A idéia básica é oferecer facilidades para um usuário definir extensões específicas para a sua aplicação. Existem abordagens direcionadas à incorporação das imagens a sistemas e comercialmente há disponíveis alguns SGDBs que permitem o armazenamento de textos e imagens [6]. Estes sistemas são de propósito geral mas nenhum deles oferece como parte do SGBD um conjunto de ferramentas para a recuperação e interpretação dos dados complexos, em particular, para o tratamento destes dados como imagens.

1.2 Trabalhos Relacionados

Alguns trabalhos como AIR [13, 14] e MACS [3] foram projetados com o objetivo de unificar características nas aplicações de bancos de dados de imagens/multimídia. O primeiro, AIR (*Adaptive Image Retrieval*) descreve uma abordagem ao problema da recuperação de imagens. Os autores propõem um *framework* unificado para a recuperação em Bancos de Dados de Imagens. A sua abordagem é baseada na premissa de que é possível desenvolver um modelo de dados e um modelo de recuperação associado de maneira que satisfaçam uma área de aplicações de recuperação de imagens. AIR utiliza múltiplas representações lógicas como parte integral do modelo de dados. Estudam-se diversas áreas de aplicações de imagens a partir do ponto de vista dos requerimentos para a recuperação (Galerias de Artes, Desenho Interior, Desenho Arquitetural, Recuperação de Informação de Faces e Negócios Imobiliários). Com isto, estabelece-se uma taxonomia dos atributos das imagens e outra de tipos genéricos de recuperação. Estes tipos genéricos de recuperação são suportados por AIR. No entanto, nada garante que as taxonomias propostas satisfaçam os requerimentos de recuperação de qualquer aplicação que use BDIs, por exemplo as aplicações de Sistemas de Informação Geográficos.

Em MACS, por outro lado, os autores propõem um framework baseado na abstração da mídia para encapsular dados heterogêneos e propõem alguns princípios ao redor dos quais os sistemas de bancos de dados multimídia podem ser construídos independentes do seu domínio. MACS (*Media Abstraction Creation System*) é uma implementação desse ambiente que permite consultar bancos de dados multimídia e integração de dados de domínios heterogêneos. MACS implementa algoritmos para criar e consultar dados abstratos e per-

mite modificações incrementais às consultas. O sistema pode ser estendido para ter acesso a uma variedade de banco de dados relacionais, estruturas de acesso e pacotes de *software* em geral usando Hermes (*Heterogeneous Reasoning and Mediator System*), um ambiente para integrar diferentes pacotes ou softwares. As informações de MACS são armazenadas em um SGBD relacional.

Em resumo, a primeira abordagem consegue estabelecer taxonomias para a modelagem de atributos e de recuperação que servem de base na hora de projetar uma aplicação. A idéia da segunda abordagem é contribuir com a capacidade dos bancos de dados multimídia de trabalhar cooperativamente com outros dados e softwares já existentes. Para isto oferece a uma aplicação, a unificação de tipos dados heterogêneos em uma linguagem de consulta e a integração de vários pacotes e softwares.

Assim, as propostas acima não oferecem ferramentas reais para que um desenvolvedor consiga construir aplicações de BDIs com o menor esforço.

Como uma tentativa para solucionar esta situação, neste trabalho se define uma arquitetura para unificar as aplicações que trabalhem a Recuperação de Imagens (RI) sobre bancos de dados relacionais e suas extensões: GRAID (*Generic Retrieval Architecture for Image Databases*). O propósito é fornecer uma plataforma comum para as aplicações de BDIs e a flexibilidade e a extensibilidade necessárias para que os programadores consigam implementar aplicações de maneira simples usando esta plataforma.

No próximo capítulo são descritas as características de alguns Sistemas de Recuperação de Imagens (SRIs). A arquitetura proposta é especificada no Capítulo 3. Uma implementação simplificada, WIRED, é apresentada nos Capítulos 4 e 5 junto com dois protótipos de aplicações de RI sobre BDIs que usam o sistema para efeito de validação das idéias propostas. No Capítulo 6 são apresentadas as conclusões e propostas de trabalhos futuros para melhorar e estender GRAID. Este trabalho possui três apêndices. O primeiro (A) contém um glosário de termos e siglas utilizados ao longo deste trabalho. O apêndice B é a especificação da linguagem de consulta de WIRED. O terceiro e último apêndice (C) contém fragmentos de código Java das aplicações usando WIRED.

Capítulo 2

Sistemas de Recuperação de Imagens

A primeira seção deste capítulo descreverá brevemente as características da recuperação de imagens baseada em seu conteúdo. A segunda seção apresentará as abordagens incluídas na classificação aos sistemas de recuperação de imagens feita em [15]. As abordagens são mencionadas a seguir.

1. **SGBDCs - SGBDs convencionais como SRIs**. Nesta abordagem as imagens são representadas por palavras chaves ou atributos associados com elas.
2. **SPI/BD - Sistemas de Processamento de Imagens com funcionalidade para banco de dados como SRIs**. As imagens são manipuladas separadamente dos seus atributos.
3. **SGBDEEs - SGBDs Estendidos e Extensíveis como SRIs**. O propósito é fornecer extensões ao modelo de dados relacional e flexibilidade para permitir tipos de dados, operadores e funções definidos pelos usuários destes sistemas.
4. **SRIAs - SRIs Adaptáveis**. Explora o conhecimento dos diferentes usuários utilizando técnicas de refinamento sobre a consulta. Aprende da interação com eles e se adapta às interpretações de cada um para obter melhores resultados na recuperação.

2.1 Recuperação de imagens baseada em conteúdo

Muitas aplicações consideram a recuperação de imagens como uma das suas atividades principais. No passado, vários esforços foram feitos para estender técnicas de bancos de dados tradicionais e suportar a recuperação de imagens baseada no conteúdo. Atualmente, a técnica mais comum para integrar imagens em um banco de dados tradicional é armazenar estes dados junto com textos descritivos ou palavras chaves. Esta abordagem é exclusivamente baseada em textos. As palavras chaves são definidas pelos operadores, o processo de entrada dos descritores consome bastante tempo e a busca está sujeita a valores subjetivos e incompletos. A recuperação pode falhar se o usuário usar palavras diferentes das palavras chaves. Adicionalmente, algumas propriedades visuais como texturas e formas são difíceis ou quase impossíveis de descrever com palavras. A única solução para estes problemas é

o desenvolvimento de métodos automatizados capazes de analisar o conteúdo das imagens [9]. As primeiras abordagens foram desenvolvidas fundamentalmente por pesquisadores da área de bancos de dados enquanto a última por especialistas da área de processamento de imagens.

Na última abordagem, a recuperação de imagens baseada no seu conteúdo é feita através da extração das características do conteúdo (CIs) utilizando técnicas de processamento de imagens. Estas CIs incluem cor, textura, forma, relações espaciais, dentre outras. O processo de recuperação é automatizado mas o reconhecimento de objetos é computacionalmente caro, dificultoso e normalmente dependente do domínio. A maioria das aplicações desenvolvem técnicas para o preprocessamento das CIs [8, 17, 18, 24, 25, 32, 35]. As CIs são precomputadas e armazenadas como um atributo tradicional. No entanto, este atributo precisa ser analisado por algum algoritmo que interprete o seu valor. Como exemplo, o histograma de cores pode ser armazenado como um tipo `text` [24] mas este atributo precisa ser processado na hora da recuperação para conhecer seu valor e estabelecer as comparações.

A inclusão das CIs como critério para a recuperação de imagens introduz os seguintes problemas nos bancos de dados [18].

- **Algoritmos e estruturas de representação**. A extração das CIs exigem algoritmos e técnicas eficientes de processamento de imagens. Os resultados destes algoritmos devem ser representados em estruturas específicas que permitam a sua manipulação eficiente.

- **Definição de critérios de similaridade**. As consultas baseadas nas CIs, em geral, não podem ser processadas com critérios de casamento exato. A descrição de uma restrição sobre uma CI é por similaridade e portanto, a recuperação é imprecisa. Dois parâmetros são analisados na hora de testar o grau de certeza de um sistema que usa a recuperação de imagens baseadas no conteúdo [24]. Para isto é feito uma análise manual do conjunto de todas as imagens relevantes a uma dada consulta no banco de dados (*conjunto relevante*) e comparado com o conjunto resultado que o algoritmo de processamento do sistema automaticamente recupera (*conjunto resultado*). Para testar o sistema, escolhe-se um conjunto de consultas e aplicam-se os seguintes parâmetros:

  recall . Do conjunto relevante para a dada consulta, o valor do *recall* é a porcentagem de imagens desse conjunto que foi encontrado no conjunto resultado;

  precision . Do conjunto resultado, o valor de *precision* é a porcentagem de imagens desse conjunto que pertence ao conjunto relevante para a dada consulta.

- **Suporte nas linguagens de consultas para as CIs**. As linguagens de consultas devem fornecer recursos para representar as CIs que nem sempre podem ser expressos de forma textual. As restrições por similaridade devem ser manipuladas como parte da consulta. Existem várias abordagens na forma de representar uma consulta.

  - *Query by pictorial example.* A especificação da consulta é feita através de um modelo de imagem representando aquela que deseja recuperar.
  - *Consultas textuais.* A linguagem oferece possibilidades de descrever o exemplo de uma imagem através consultas por termos ou conceito. Por exemplo, definindo recursos textuais tais como *SomeOrange*, *sunset* [24], dentre outros.

- *Consultas visuais.* Ferramentas gráficas são fornecidas ao usuário para descrever as CIs que deseja recuperar. A linguagem de consulta em QBIC apresenta um conjunto de janelas que define visualmente uma consulta, por exemplo, as formas e posição dos objetos nas imagens podem ser representadas graficamente.

- **Suporte da indexação das CIs.** A execução eficiente de consultas sobre grandes bancos de dados de imagens baseadas em CIs depende da possibilidade de criar estruturas de indexação para elas. Dada a natureza inexata da recuperação, estas estruturas devem considerar o acesso rápido a imagens a partir do critério de similaridade adotado. Estruturas de indexação foram definidas em [10, 17, 18, 26].

- **Integração da recuperação de dados tradicionais e CIs.** Apesar das áreas de aplicação serem diversas, muitas delas consideram as imagens como seu recurso principal e percebem a necessidade da sua manipulação integrada a outros tipos de dados. No entanto, a integração das CIs com os diversos tipos de dados tradicionais não é resolvida em muitos sistemas [8, 10, 20, 33, 39].

2.2 Abordagens aos Sistemas de Recuperação de Imagens

Existem várias abordagens para implementar um Sistema de Recuperação de Imagens (SRI). A taxonomia aos SRIs proposta em [15] inclui as abordagens descritas nas seguintes subseções.

2.2.1 Sistemas de Bancos de Dados Convencionais como SRIs

Uma primeira abordagem aos SRIs foi sobre os SGBD convencionais. Estes sistemas foram inicialmente projetados para manipular somente dados atômicos. No entanto, esta abordagem tem sido uma das mais populares no desenvolvimento de SRIs. Sob este esquema, cada imagem é representada por palavras chaves ou atributos da imagem. Tipicamente, a linguagem de consulta e a estratégia de recuperação usadas são as tradicionais do SGBD. Os arquivos das imagens não são uma parte integral dos dados manipulados pelo SGBD. Esta abordagem é chamada de RI baseada em atributos.

A vantagem desta abordagem é a relação custo-efetividade pois toda a funcionalidade dos SGBDs convencionais está facilmente disponível. Contudo, a recuperação pode falhar se o usuário usar palavras diferentes das palavras chaves. Freqüentemente, algumas propriedades visuais como texturas e formas são difíceis ou quase impossíveis de descrever com palavras. Por outro lado, as imagens são objetos complexos e as relações entre eles são numerosas. A representação das relações entre estes objetos devem ser mapeadas em tabelas relacionais e a recuperação feita através de junções. Este último é um processo caro e torna o sistema ineficiente para aplicações com um grande repositório de imagens.

Como exemplo de sistemas dentro desta categoria se encontra a abordagem de Tang [37] onde é apresentado um sistema de gerenciamento organizado logicamente como um conjunto de relações. A linguagem SEQUEL somente é estendida para fornecer uma interface ao usuário e para prover facilidades de migração e de compressão para as imagens.

2.2.2 Sistemas de Processamento de Imagens sobre banco de dados como SRIs

Outra abordagem foi a construção de sistemas de processamento de imagens sobre bancos de dados. Os sistemas nesta categoria manipulam as imagens como dados cuja estrutura não é compreensível pelo sistema (dados complexos). Os atributos das imagens também podem ser armazenados e recuperados. Por conseguinte, existem dois modelos de dados associados aos sistemas nesta categoria: um para a visão dos dados complexos e outro para a visão dos dados associados às imagens.

Para a visão dos dados complexos, o modelo de dados empregado é uma das representações do nível físico: matricial ou vetorial. A manipulação dos dados complexos requer métodos especiais para compreender as semânticas e executar operações sobre eles. A especificação de consultas e a estratégia de recuperação são feitas através da interação dos usuários com o sistema usando um conjunto de comandos. Um comando, tipicamente, especifica uma operação de processamento do conteúdo da imagem para ser executada. Geralmente, a estratégia de recuperação é baseada em comandos ou funções equivalentes a um conjunto de comandos. Uma seqüência de comandos expressa as necessidades de um usuário.

Para a visão dos dados associados às imagens, os atributos externos ao conteúdo das imagens ou os resultados das interpretações das imagens (atributos semânticos) podem ser armazenados como dados de tipo alfanumérico (dados atômicos) usando ou não um SGBD convencional. Nos sistemas onde os dados atômicos são limitados aos dados externos ao conteúdo das imagens, o modelo de dados usado é simplesmente um conjunto de atributos ou palavras chaves armazenados no cabeçalho do arquivo de dados da imagem e não se integram com SGBDs convencionais. O sistema de arquivos do sistema operacional é acrescentado com alguma funcionalidade para armazenar, editar e recuperar esses dados no cabeçalho do arquivo de dados da imagem. Por outro lado, os sistemas que incluem SGBDs convencionais utilizam o modelo relacional para modelar e recuperar os atributos semânticos, assim como os atributos externos ao conteúdo das imagens.

Os usuários nesta categoria devem ser especialistas do domínio da aplicação e ter familiaridade com o processamento de imagens oferecido pelo sistema. Esse tipo de sistemas oferece como parte integral do sistema operações de análise das imagens tanto de propósito geral quanto de domínio específico.

Os seguintes sistemas pertencem a esta categoria.

QBIC (*Query by Image Content*). O sistema QBIC é um sistema de recuperação de imagens baseado no conteúdo. Usa a análise das imagens para o processamento de consultas para um banco de dados de imagens. A abordagem usada para a análise das imagens é a extração prévia das CIs [12]. A extração é feita semi-automaticamente. O sistema oferece consultas sobre bancos de dados de imagens baseadas em imagens exemplos, desenhos definidos pelos usuários e padrões de texturas, cores e formas. No entanto, a integração das CIs com os diversos tipos de dados tradicionais suportados em um SGBD relacional não é resolvida neste sistema.

No modelo de dados, existem dois tipos de dados principais: as imagens (ou cenas) e os objetos das imagens. Uma cena tem zero ou mais objetos. Os objetos são identificados manualmente ou semi-automaticamente. As CIs para capturar os seus atributos são computadas durante a etapa de entrada de dados no banco. As consultas são baseadas nas

cenas (*scene-based*) ou nos objetos (*object-based*) e são processadas por métodos que oferecem aproximações baseados em similaridade. A interface com o usuário é gráfica e permite aos usuários especificar visualmente consultas por exemplos, refinamento dos resultados e navegação no banco de dados.

O QBIC não inclui a combinação das consultas visuais de conteúdo com métodos de busca tradicionais, nem os operadores lógicos nas consultas. Porém, estes elementos fazem parte do trabalho futuro no sistema.

PICTION. Srihari examina em [35] o papel das informações textuais como informação colateral para a extração de características das imagens e a sua recuperação pelo conteúdo em um ambiente de banco de dados integrando textos e imagens. O sistema PICTION usa os subtítulos (*captions*) das fotografias para identificar faces (rostos) nas fotos. Com isto, mostra como informações dos textos e as imagens podem ser usadas para computar similaridade entre uma consulta e as imagens de um banco de dados. Este sistema é baseado na extração prévia das CIs de maneira automática. O sistema emprega duas técnicas principais: processamento da linguagem natural para obter informação dos subtítulos das fotos e o reconhecimento de objetos para localização de faces. Por exemplo, a idéia é extrair informação dos subtítulos das fotos de um jornal para ser usada para a recuperação da foto ou para a identificação das pessoas nela. PICTION usa as restrições espaciais (geométricas ou topológicas), características (propriedades dos objetos; por ex.: gênero, cor do cabelo) e contextuais (por ex.: as pessoas na foto mencionadas no subtítulo, e outras características da cena como o lugar - apartamento, aeroporto, etc.) derivadas dos subtítulos para rotular rostos gerados pelo localizador de faces. O sistema aplica-se para projetos de domínio específico: fotografias com subtítulos. As técnicas de análise do conteúdo foram feitas somente para o reconhecimento de faces humanas. Ambos os processamentos da linguagem e de imagens são complexos.

Abordagem de Lu [18]. Os autores propõem um tipo de estrutura de indexação chamada de *three-tier color index*. Esta nova estrutura baseada em cor consiste de três níveis de abstração sucessivos:

1. classificação por cores dominantes;
2. R-tree [16]: árvore de busca balanceada utilizada para indexar dados espaciais (por exemplo, imagens);
3. histograma de cores de múltiplos níveis: estrutura quad-tree [34] utilizada para manipular informação sobre a posição das cores em uma imagem.

As imagens passam por um processamento nestes três níveis. Cada nível constrói seu conjunto resultado, o qual é passado ao nível seguinte. O resultado final é composto pelas imagens que formam o conjunto resultado do último nível.

Foi implementado um protótipo sobre um banco de dados de pinturas para demonstrar a efetividade da técnica de indexação proposta. A CI extraída foi a cor. A extração foi feita de maneira precomputada armazenando o histograma de múltiplos níveis. Além da estrutura de indexação por cor (*three-tier color index*), outras duas estruturas foram implementadas: B^+tree para atributos estruturados e um *signature file* para as descrições baseadas em textos. A recuperação por combinação destas estruturas foi também implementada. Um

amplo número de consultas baseadas em texto e em conteúdo pode ser processado. Por exemplo, é possível recuperar uma imagem baseada em uma imagem exemplo mais atributos tradicionais e palavras chaves.

2.2.3 Sistemas de Bancos de Dados Estendidos e Extensíveis como SRIs

Um grande volume de pesquisa tem sido dedicado a estender o modelo de dados relacional com o objetivo de solucionar as dificuldades impostas pela estrutura tabular plana que impõe a primeira forma normal. O modelo de dados resultante é caracterizado pela adição de componentes específicos de uma aplicação ao núcleo de um SGBD existente, tais como relações aninhadas, campos procedurais e extensões à linguagem de consulta. Neste esquema, as imagens são armazenadas no sistema como dados atômicos, mas este fato é mantido transparente para o usuário. O usuário enxerga as imagens como dados complexos ou estruturados. A linguagem de consulta é a mesma do SGBD. No entanto, o poder expressivo da linguagem é incrementado devido ao fato dos usuários poderem especificar nomes de procedimentos para valores de atributos na formulação de uma consulta. Observe-se que este incremento no poder da linguagem de consulta traz como conseqüência uma perda da performance já que um nome de procedimento, por exemplo, pode implicitamente especificar várias operações de junção. A estratégia de recuperação é exatamente a mesma do SGBD. Alguns destes mecanismos foram implementados nas linguagens de consulta Quel [7] e Postquel [36].

Devido às operações e às representações das imagens mudarem de uma aplicação para outra, cada aplicação escolhe o SGBD que mais eficientemente suporte as suas operações. Isto dificulta a construção de aplicações de BDIs de propósito geral baseados nos modelos de dados convencionais por causa do seu limitado conjunto de tipos de dados. Como uma solução para este problema, introduziu-se os SGBDs extensíveis. A idéia básica é oferecer facilidades para um usuário definir extensões específicas para a sua aplicação. Um SGBD extensível precisa suportar tipos de dados abstratos. Existem duas abordagens essenciais para o suporte de tipos de dados abstratos nestes sistemas. A primeira considera o modelo de dados fixo e implementa este modelo de dados como um sistema com interfaces bem definidas para extensões dos usuários. Este é o caso de POSTGRES e POSTGRESQL [36, 11], os quais são baseados no modelo relacional. A segunda abordagem considera o modelo de dados variável e oferece um conjunto de ferramentas para a construção de uma aplicação de domínio específico como no caso dos SGBDs extensíveis GENESIS [2] e EXODUS [5].

Novas características foram criadas para melhorar a representação dos dados, tais como atributos de tipo conjunto (`SET`), campos procedurais, BLOBs e definição de novas operações e operadores. As linguagens de consulta dos SGBDs convencionais foram estendidas para incluir estas características. A estratégia de recuperação para considerar estas novas características na especificação da linguagem de consulta está sendo ativamente pesquisada. No entanto, nenhum deles oferece um conjunto de ferramentas para a recuperação e interpretação dos tipos de dados abstratos, em particular, para o tratamento destes dados como imagens.

Alguns dos sistemas que fizeram extensões para suportar tipos de dados complexos como as imagens são brevemente descritos a seguir.

DB2. DB2[6] é um SGBD relacional cliente/servidor desenvolvido com fins comerciais pela

IBM. Este SGBD fez extensões para fornecer às aplicações ferramentas para armazenar e recuperar dados multimídia (*large objects*). As aplicações podem importar/exportar imagens e seus atributos. O DB2 *Image Extender* armazena atributos tais como o tamanho em bytes, formato, número de cores, etc. das imagens. O sistema suporta os formatos de imagens GIF, JPEG, BMP e TIFF. As consultas/recuperações podem ser baseadas tanto nos atributos mantidos pela aplicação (atributos tradicionais) quanto nos atributos mantidos pelo sistema DB2 (atributos das imagens). Adicionalmente, o sistema oferece recuperações com o QBIC que além de apresentar consultas visuais por exemplos, inclui critérios de buscas textuais para especificar uma cor ou textura. Estes dois tipos de recuperações não podem ser usados por uma aplicação em uma mesma consulta. As restrições da consulta não incluem métodos de análise pelo conteúdo da imagem. Para isto, usa-se o QBIC como outro tipo de consulta.

PostgreSQL. O PostgreSQL é um sistema relacional estendido de domínio público com características de orientação a objetos. O conceito fundamental do PostgreSQL são as classes. Uma classe é uma coleção de instâncias de objetos. Cada instância tem uma coleção de atributos e cada atributo é de um tipo de dado específico. Assim sendo, uma tabela é chamada de classe, uma tupla de instância e os campos são os atributos. Então, uma tabela pode herdar de outra com as mesmas características da herança dos modelos orientados a objetos. Todo acesso às tabelas é através de instruções SQL. Permite também funções, tipos de dados, operadores e métodos de acesso definidos pelo usuário. O PostgreSQL pode manipular grandes quantidades de informação, tem arquitetura cliente-servidor e possui uma API para Java.

MySQL. O sistema de domínio público MySQL[41] faz uma extensão de um modelo relacional e da sua linguagem de consulta. As extensões introduzidas na SQL estão mais encaminhadas ao usuário final. Em particular, não possui funções para o trabalho com o conteúdo dos BLOBs. Estes tipos de dados não podem ser indexados. Este SGBD relacional estendido não implementa tipos de dados nem operadores definidos pelo usuário.

Várias aplicações utilizam estas propriedades dos SGBDs extensíveis tal como:
CHABOT. Este trabalho integra sistemas de bancos de dados relacionais com técnicas de análise do contéudo das imagens. O sistema demonstra que critérios para a recuperação por atributos podem ser combinados com critérios baseados no conteúdo das imagens para conseguir bons resultados na recuperação. Na versão atual, a análise do conteúdo das imagens é baseada em histogramas de cores e técnicas de otimização de consultas são usadas para minimizar o número de histogramas considerado quando uma consulta é processada. O sistema usa o SGBD relacional Postgres. Ele usa as características extensíveis de Postgres para integrar os dados tradicionais e a análise baseada no conteúdo com o propósito de definir *consultas por conceito*. Uma consulta por conceito envolve informações contextuais tal como *céu azul*. O usuário define este conceito incluindo nas restrições da sua consulta a palavra *céu* no campo de descrição ou imagens com a cor *azul* como predominante. As consultas são definidas com informações textuais, inclusive os atributos relacionados com a análise contextual das imagens. Apesar de que as consultas podem expressar conceitos com palavras, o sistema não oferece as restrições das consultas baseadas em amostras de imagens. Dentre os atributos que a tela de busca oferece, têm vários onde a recuperação

é restrita a um conjunto de opções definidos pelo sistema, o qual limita as consultas do usuário.

2.2.4 SRIs Adaptáveis

Existem outros tipos de SRIs que levam em consideração as interpretações das imagens pelos diferentes usuários. Um conjunto de imagens resultante de uma recuperação pode mudar de um usuário a outro para uma dada consulta. Com este objetivo, alguns SRIs oferecem interfaces de consultas flexíveis para ter em conta as diferentes interpretações das imagens. Adicionalmente, o sistema pode solicitar aos usuários o refinamento (*relevance feedback*) iterativo da consulta desde o resultado inicial e assim, aprender incrementalmente da perspetiva de usuários e conseguir melhores resultados individuais. Atualmente, várias pesquisas estão baseadas nesta abordagem.

Como exemplos de sistemas dentro desta categoria se encontram:

Abordagem de Picard [28]. Os autores descrevem as suas pesquisas sobre qual modelo usar para a análise de imagens em uma biblioteca de imagens e como combinar estes modelos para satisfazer as requisições dos usuários. Um dos problemas no processamento do conteúdo das imagens é quando o conteúdo da imagem é difícil de descrever objetivamente. Dentre as soluções propostas nesta pesquisa, o usuário não precisa selecionar parâmetros do modelo senão simplesmente escolher imagens exemplos. A pesquisa se baseia na subjetividade das requisições dos usuários e possui dois componentes.

1. Modelo de Inferência e Combinação: o sistema infere automaticamente a melhor combinação de modelos para representar os dados de interesse a um usuário. A análise tenta inferir as CIs comuns às imagens escolhidas por um usuário e as usa para predizer outras imagens de interesse para ele. A inferência é sempre baseada no conjunto atual de exemplos positivos e negativos dados por um usuário.

2. Instrução e Generalização - Subjetividade na Modelagem: o sistema aprende continuamente durante a interação com cada usuário. O conhecimento aprendido em um problema pode ser usado para resolver outro problema.

IDQS (*Image Database Query System*). Este sistema descrito em [39] foi projetado para explorar o conhecimento dos usuários utilizando técnicas de refinamento iterativo sobre as informações relevantes em tempo de execução. Após a tentativa inicial de recuperação, o refinamento é dado pela aceitação ou rejeição desses resultados. Esta informação é usada como uma coleção de exemplos positivos e negativos que são analisados e classificados. O processo iterativo faz o sistema aprender cada vez mais com as informações dos usuários e melhorar o processo de recuperação. IDQS consiste de três etapas: um preprocessamento para estabelecer a segmentação e os vetores de CIs para cada imagem, as seções de consultas com ciclos de refinamento, e um treinamento *off-line* para construir uma biblioteca com informações referentes aos objetos envolvidos no processo de recuperação.

2.3 Sumário

Os SRIs assumem uma função importante na maioria das aplicações de bancos de dados de imagens. Inicialmente, a proposta foi manipular as imagens dentro do ambiente dos

SGBDCs através da extração manual de seus atributos. O maior problema da abordagem baseada em atributos é que as consultas são limitadas a valores de atributos predeterminados e portanto, não é simples para os usuários formular suas consultas adequadamente. Para melhorar esta situação armazenou-se também as características geométricas das imagens como dados atômicos em tabelas relacionais. Esta última abordagem introduz uma abertura entre a conceitualização de uma consulta pelos usuários e a consulta que é especificada ao sistema. Nas abordagens posteriores as imagens são tratadas como dados complexos através da criação dos tipos de dados abstratos, mas foi ao introduzir a interpretação das imagens como uma parte integrante dos SRIs que se superaram as limitações da recuperação baseadas em atributos das imagens.

Na tabela 2.1 é mostrada uma comparação entre as abordagens aos SRIs descritas neste capítulo. Os aspectos a comparar são os problemas encontrados na recuperação de imagens baseadas nas CIs.

Aspectos	SGBDCs	SPI/BD	SGBDEEs	SRIAs
Recuperação por atributos tradicionais	Sim	Sim	Sim	Geralmente Não
Recuperação por processamento das CIs	Não	Sim	É possível	Sim
Recuperação integrada em uma consulta	Não	Nem sempre	É possível	Geralmente Não
Sistemas de domínio específico	Não	Sim	Não	Sim
Sistemas de propósito geral	Sim	Não	Sim	Não
- Com funcionalidades para imagens	Não	Não	Não	Não
- Incorporação de algoritmos de p. i. [a]	Não	Não	É possível	Não
- Reusabilidade de algoritmos de p. i.	Não	Não	Não oferece mecanismos	Não

[a] p. i. processamento de imagens

Tabela 2.1: Tabela comparativa das diferentes abordagens aos SRIs

Nesta área ainda existem limitações nos sistemas para satisfazer os requerimentos referentes à complexidade da representação das imagens, à diversidade das técnicas para a especificação de consultas (por exemplo: SQL, QBE) e ao conjunto de operações requeridas para englobar todos os domínios. Além disso, a maioria dos problemas na recuperação de imagens é originada pelas necessidades de domínio específico e por isto as soluções encontradas são limitadas somente a um conjunto de domínios de aplicação.

Este trabalho define uma plataforma comum para unificar os Sistemas de Recuperação de Imagens. A idéia é permitir que as aplicações pertencentes a alguma das abordagens anteriores possam ser adaptadas e incorporadas a um sistema que implemente a arquitetura. Para isto, a inclusão dos algoritmos de processamento de imagens será separada da recuperação em si. No próximo capítulo será apresentada a especificação de GRAID (*Generic Retrieval Architecture for Image Databases*), uma arquitetura orientada a satisfazer os requerimentos expostos acima.

Capítulo 3

Arquitetura GRAID

Este capítulo apresentará a arquitetura GRAID (*Generic Retrieval Architecture for Image Databases*). GRAID é uma arquitetura que pretende ser genérica o suficiente para ser adaptável a qualquer domínio de aplicação. Para isto, é preciso separar as operações sobre as imagens que são independentes das operações dependentes do domínio de aplicação. Esta separação permite uma definição mais clara das operações de uma aplicação.

O principal objetivo é especificar, no maior nível de detalhes possível, as operações que não dependem da natureza de uma aplicação, deixando para o desenvolvedor da aplicação a tarefa de especificar apenas os detalhes que dizem respeito à natureza da mesma.

Uma primeira seção explica as motivações que levaram a propor esta abordagem e descreve uma idéia geral da arquitetura. A segunda seção expõe os princípios de projeto de GRAID. Já a terceira seção especifica a estrutura e o funcionamento da arquitetura.

3.1 Descrição geral do problema

Para o propósito deste trabalho, a noção de banco de dados de imagens é estreitamente relacionada à de bancos de dados no sentido tradicional. A falta de modelos de propósito geral para recuperação de imagens para aplicações de BDIs e a falta de funcionalidades nos SGBDs tradicionais para manipular imagens trazem, freqüentemente, grandes dificuldades aos pesquisadores.

Como uma tentativa para solucionar esta situação, esta abordagem define a arquitetura GRAID para unificar as aplicações que fazem recuperação sobre BDIs. GRAID deverá ser apropriada para qualquer domínio de aplicação.

Esta abordagem é definida sobre os SGBDs relacionais e suas extensões, aproveitando assim as facilidades oferecidas por estes sistemas na manipulação de dados tradicionais e oferecendo para eles funcionalidade na interpretação e recuperação de imagens. Para isto, GRAID foi projetada definindo uma camada sobre os SGBDs já existentes. Na figura 3.1 é apresentada uma visão geral dos SGBDIs como uma extensão ou camada entre a aplicação e o SGBD.

Conforme mostra a figura 3.1, define-se uma arquitetura de quatro camadas. A primeira camada especifica o repositório de imagens e seus dados associados. A camada dois resolve o gerenciamento dos dados tradicionais através de um SGBD relacional ou alguma das suas extensões. A terceira camada é a camada dos sistemas de gerenciamento de imagens e a

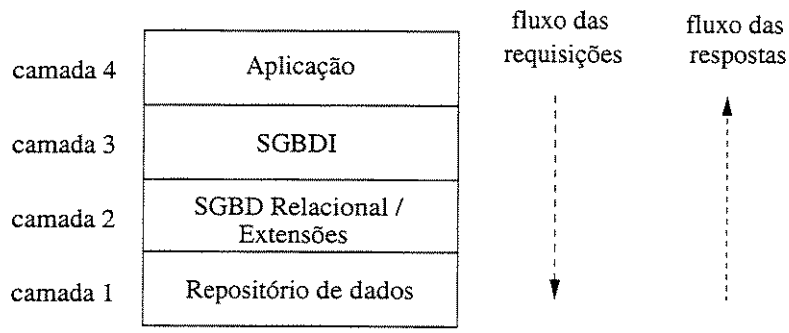

Figura 3.1: Modelo em camadas para os SGBDIs.

camada quatro especifica a aplicação, geralmente de domínio específico.

Para realmente oferecer aos desenvolvedores um tipo de dado *imagem*, a terceira camada deve ser um verdadeiro SGBDI, suportando o conjunto de operações que façam das imagens mais um tipo de dado disponível para as aplicações. O gerenciamento de imagens inclui um conjunto de operações que manipulam as imagens e seus atributos tais como a inserção, exclusão, atualização, recuperação de imagens e seus atributos associados no BDI e os algoritmos de processamento e indexação de imagens.

De todas estas operações, a recuperação de imagens é uma das mais utilizadas. Neste trabalho vai ser estudada em particular a operação de RI, definida como a *recuperação de imagens e seus atributos associados.*

O propósito é oferecer ao desenvolvedor de aplicações meios simples para criar suas aplicações associadas a bancos de dados de imagens. Neste contexto, chama-se de *aplicações de RI* aos sistemas de recuperação de imagens que usam como base BDIs.

Uma aplicação de RI poderá ser incorporada à arquitetura se é projetada para usar as funcionalidades oferecidas. Assim sendo, ela poderá reutilizar os algoritmos de processamento de imagens, repositórios de imagens e até modelos dos dados das imagens já incluídos.

O seguinte exemplo mostra como várias aplicações podem ser agrupadas sobre uma plataforma comum. *Suponha-se três aplicações de RI. A primeira é construída para os pesquisadores em um Museu, a segunda para os estudantes de uma Escola de Artes e a terceira para uma Agência de Turismo que deseja oferecer consultas a seus clientes interessados em conhecer previamente fotos dos lugares mais atraentes de uma região. Ambas as aplicações do Museu e da Escola de Artes estão precisando de um repositório de pinturas impressionistas. No entanto, as necessidades de recuperação são diferentes para estas duas aplicações. Mais especificamente, a primeira precisa de um algoritmo para reconhecer os objetos nas pinturas e de atributos com palavras chaves enquanto a segunda precisa recuperar as pinturas pela porcentagem de cores contida nelas. Os atributos relevantes a esta aplicação são os atributos externos às pinturas como os pintores e a data da pintura. Por outro lado, a Agência de Turismo está precisando para sua aplicação de um algoritmo para calcular a porcentagem de cores contidas nas paisagens das fotos.*

Um mesmo repositório de pinturas impressionistas poderia ser utilizado pelas aplicações do Museu e da Escola de Artes. Por sua vez, o algoritmo que processa as cores das imagens poderia ser utilizado pelas aplicações da Escola de Artes e da Agência de Turismo.

Note que apesar da arquitetura ser independente de domínio, as aplicações continuarão sendo dependentes do seu domínio mas podem fazer uso desta arquitetura e utilizar, no caso de conveniência, componentes já existentes.

Observe na figura 3.2 como seria o fluxo dos dados através do modelo, uma vez que as aplicações são incorporadas a GRAID. No esquema, o fluxo para cada aplicação tem um traço diferente. Indicados com círculos, note-se como a nova arquitetura permitirá que várias aplicações tirem proveito de GRAID e como uma conseqüência, reutilizam o banco de dados de pinturas impressionistas e o algoritmo do histograma de cores. No exemplo, as pinturas são recuperadas pelo Museu e a Escola de Artes, e a implementação do histograma de cores é usada pela Escola de Artes e a Agência de Turismo.

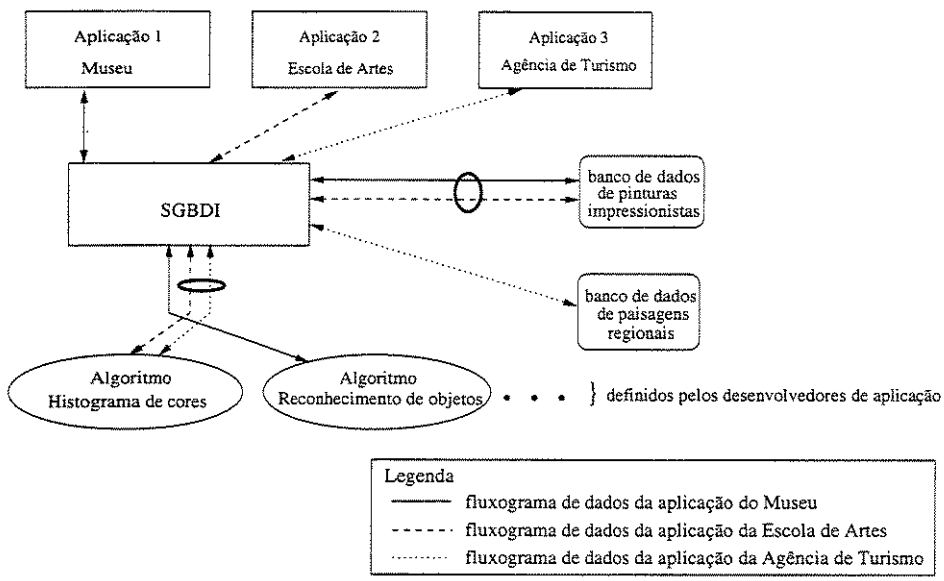

Figura 3.2: Fluxograma de aplicações de RI sobre uma plataforma comum.

3.2 Princípios de projeto de GRAID

Os princípios de projeto de GRAID podem ser resumidos nos itens mostrados a seguir.

- Definir uma arquitetura de propósito geral para as aplicações de RI sobre SGBDs relacionais e suas extensões.
- Permitir a recuperação integrada dos tipos de dados tradicionais com as imagens.
- Facilitar a incorporação de novos algoritmos de processamento de imagens ao sistema e permitir a reusabilidade dos algoritmos já existentes.

Para recuperar informação relacionada às imagens, um SGBD precisa conter mais funcionalidades que as tradicionais. Isto inclui algum suporte para os algoritmos de processamento do conteúdo das imagens. A chave para construir uma arquitetura independente de domínio é a separação das funções de processamento e de recuperação de imagens.

3.3 Especificação de GRAID

A figura 3.3 apresenta o esquema geral da arquitetura. Como mostrado, o fato mais relevante no esquema é a comunicação da aplicação com o SGBD através do SGBDI. Em particular, as consultas de seleção feitas pela aplicação serão atendidas por GRAID.

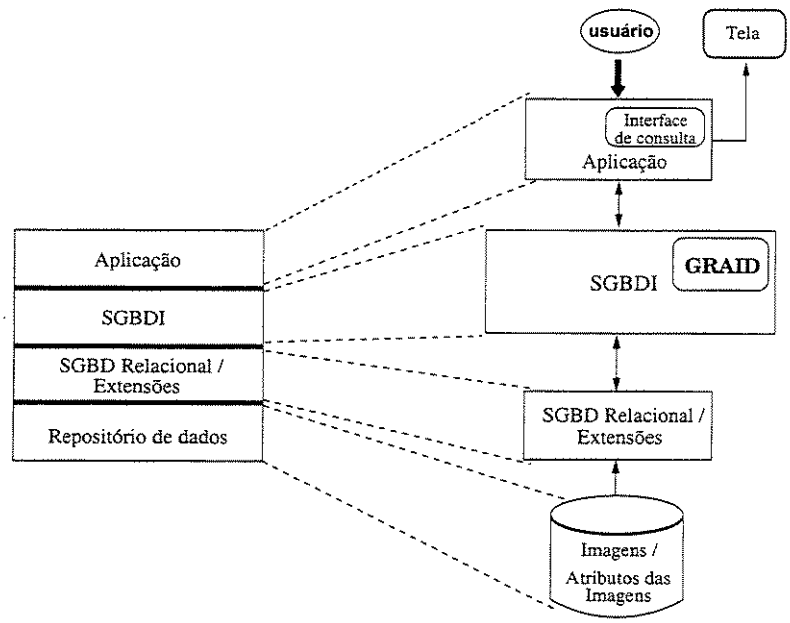

Figura 3.3: Esquema geral de GRAID.

Conforme mostra a figura 3.4, a arquitetura inclui seis componentes.

1. **Interface de Aplicação (IA)**. Comunica a aplicação com GRAID.
2. **Interpretador de Consulta (IC)**. Processa as consultas, busca e combina os resultados.
3. **Interpretador de Imagens (I^2)**. Interface dos algoritmos de processamento para a análise do conteúdo das imagens. Adicionalmente, reúne os resultados do processamento de imagens correspondente a uma dada consulta.
4. ***Driver* de SGBD**. Estabelece a relação entre GRAID e o SGBD utilizado pela aplicação.
5. **Interface de Configuração**. É a interface para que os desenvolvedores de aplicações consigam incorporar seus algoritmos de processamento de imagens em GRAID.
6. **Repositório de Algoritmos**. Mantém uma tabela com todos os algoritmos de processamentos de imagens existentes em GRAID. O objetivo é permitir a reusabilidade destes algoritmos.

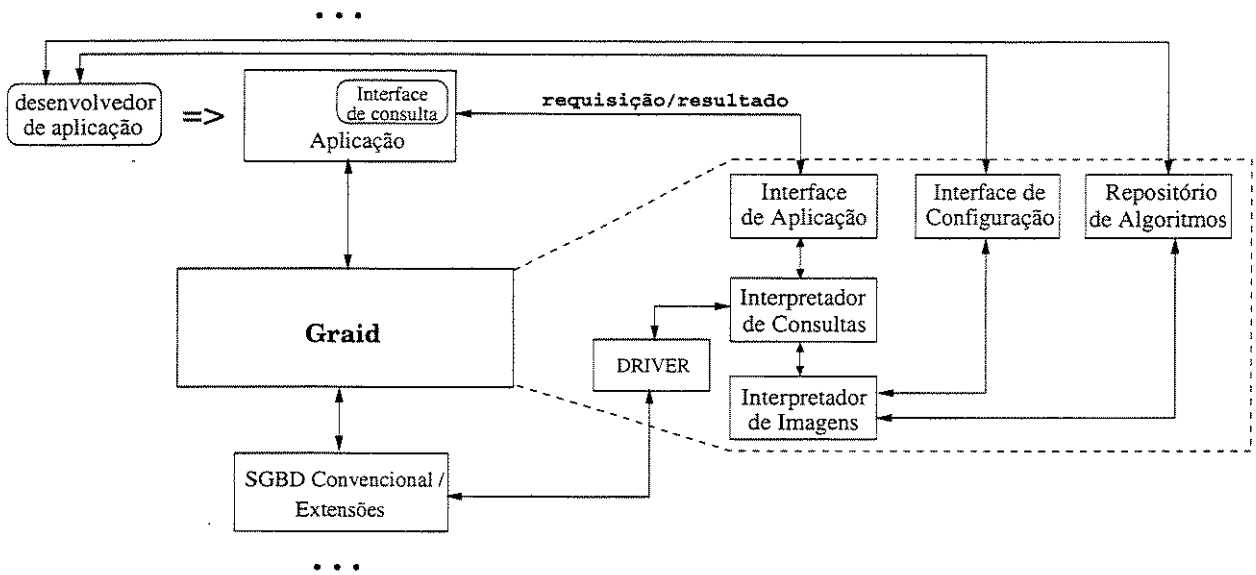

Figura 3.4: Componentes de GRAID.

3.3.1 Interface de Aplicação

Para implementar as aplicações é necessária uma interface de programação. Esta interface fornece as ferramentas necessárias aos programadores para implementar aplicações que respondam ao modelo de programação do sistema. A Interface de Aplicação estabelece os parâmetros para a iniciação e envio das requisições ao Interpretador de Consulta. Ela devolve para a aplicação o conjunto resultado da requisição.

Uma requisição pode ser elaborada de várias maneiras. Uma das formas mais comuns para construir uma requisição é com uma linguagem de consulta. GRAID deixa para os implementadores da arquitetura a forma de elaboração de uma requisição. O único requisito de GRAID com respeito às requisições é a sua independência quanto aos SGBDs.

A falta de ferramentas nas linguagens de consultas dos SGBDs relacionais para trabalhar com tipos de dados complexos e a heterogeneidade de linguagens de consultas (apesar da SQL ser a mais usada) foram as causas para definir a arquitetura independente da linguagem de consulta dos SGBDs.

Como conseqüência, esta independência permitirá definir requisições com funcionalidades mais específicas às imagens. Além disso, uma requisição em uma camada superior aos SGBDs poderia definir semânticas para suportar consultas que incluam bancos de dados criados em diferentes SGBDs.

Uma requisição deve especificar no mínimo três parâmetros fundamentais na hora de construir uma consulta:

- **Lista de informações destinos**. Lista de atributos de saída.
- **Lista das tabelas**. Tabelas do banco de dados que intervêm na requisição.
- **Lista das restrições do usuário**. As restrições que os usuários impõem ao conjunto

resultado. Envolvem atributos tradicionais e/ou restrições de análise do conteúdo das imagens. Este parâmetro é opcional na requisição.

3.3.2 Interpretador de Consulta

O Interpretador de Consulta é chamado pela aplicação de RI através da Interface de Aplicação. Ele é o encarregado de processar as consultas, separar as cláusulas tradicionais daquelas que requerem processar algoritmos de imagens e definir um plano de execução para uma dada consulta. Para isto, foram definidos vários sub-componente do IC, como observado na figura 3.5.

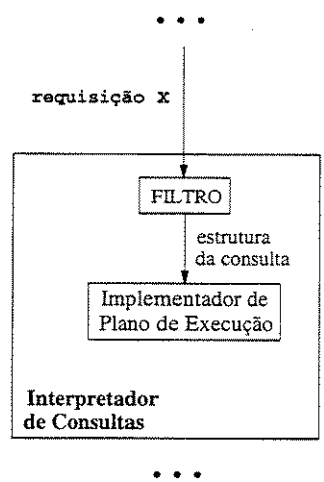

Figura 3.5: Interpretador de Consultas.

1. **Filtro**. Separa as cláusulas tradicionais daquelas que requerem algoritmos de processamento de imagens e define um plano de execução que envolva o menor número possível de registros no processamento de imagens. Pressupõe-se que as restrições de uma requisição estejam relacionadas através de operadores lógicos. Portanto, é possível dividir a lista de restrições em cláusulas atômicas e criar um plano de execução, conforme mostram os algoritmos em [7]. A forma de distinguir os dois tipos de restrições é dependente da estrutura da requisição. O plano de execução para o processamento das consultas depende dos operadores envolvidos. O algoritmo para o processamento das cláusulas das consultas é decidido na implementação de GRAID.

2. **Estrutura da consulta**. Estrutura de dados na qual é definida o plano de execução para uma dada consulta. Geralmente é construída em forma de árvore.

3. **Implementador do plano de consulta**. Processa a estrutura da consulta. Para isto, envia as requisições tradicionais para o Driver de SGBD e as restrições de processamento para o Interpretador de Imagens. Faz a combinação dos resultados.

3.3.3 Driver de SGBD

O Driver estabelece a relação entre GRAID e o SGBD utilizado pela aplicação. Ele é chamado pelo IC para atender as consultas tradicionais requisitadas pela aplicação.

Dado que nem todas os SGBDs possuem a mesma linguagem de consulta, a sua função é mapear as requisições na linguagem de consulta do dado SGBD, estabelecer a comunicação com o mesmo e obter os resultados das consultas, os quais mapeia em resultados em GRAID.

Em princípio, este elemento consegue a independência de GRAID com os SGBDs pois é o único componente a ser mudado de um SGBD para outro. Isto é, a introdução do Driver como um componente de GRAID, implica que uma mudança de SGBD pela aplicação de RI não causa mudanças no resto dos componentes de GRAID.

3.3.4 Interpretador de Imagens

O I^2 é um modelo de interfaces a partir do qual vão ser colocadas as diferentes implementações dos algoritmos para a análise das imagens. Ele é chamado pelo Interpretador de Consulta para executar os algoritmos de processamento de imagens. O esquema dos sub-componentes do I^2 é mostrado na figura 3.6. Duas estruturas foram definidas para

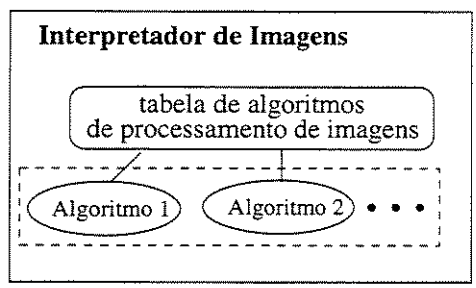

Figura 3.6: Interpretador de Imagens.

conseguir a incorporação de novos algoritmos a GRAID.

1. Uma tabela de algoritmos de processamento de imagens para armazenar os nomes dos algoritmos existes em GRAID. Esta tabela é usada para construir o repositório de algoritmos.

2. Uma estrutura para armazenar as implementações desses algoritmos.

Uma implementação de GRAID deverá oferecer instruções claras aos desenvolvedores de aplicação sobre como implementar novos algoritmos no sistema e como reutilizar algoritmos já existentes.

3.3.5 Repositório de Algoritmos e Interface de Configuração

Um dos princípios de projeto de GRAID é facilitar a incorporação de novos algoritmos de processamento de imagens ao sistema e permitir a reusabilidade dos algoritmos já existentes. Para conseguir este objetivo foram definidos em GRAID dois componentes.

O Repositório de Algoritmos é criado para permitir a reusabilidade dos algoritmos já existentes. Ele usa a tabela criada pelo Interpretador de Imagens para construir um repositório de algoritmos de processamento de imagens. A função deste componente é ajudar ao desenvolvedor de aplicações na procura dos algoritmos adequados para sua aplicação, a forma em que devem ser implementados os algoritmos e a estrutura das requisições.

A Interface de Configuração é criada para facilitar a incorporação de novos algoritmos de processamento de imagens ao sistema pelos desenvolvedores de aplicação. Uma vez incorporados ao sistema, estes algoritmos formam parte do repositório. Qualquer algoritmo de processamento de imagens disponível em GRAID pode ser incluído nas requisições.

Estes dois componentes foram projetados para interagirem com os desenvolvedores de aplicações antes da execução da aplicação de RI.

3.4 Fluxogramas de Requisições e Aplicações em GRAID

O fluxo de uma requisição em GRAID é mostrado na figura 3.7. As requisições em GRAID serão chamadas de maneira abstrata de `requisição X` conforme é observado na figura. Os passos são detalhados a seguir.

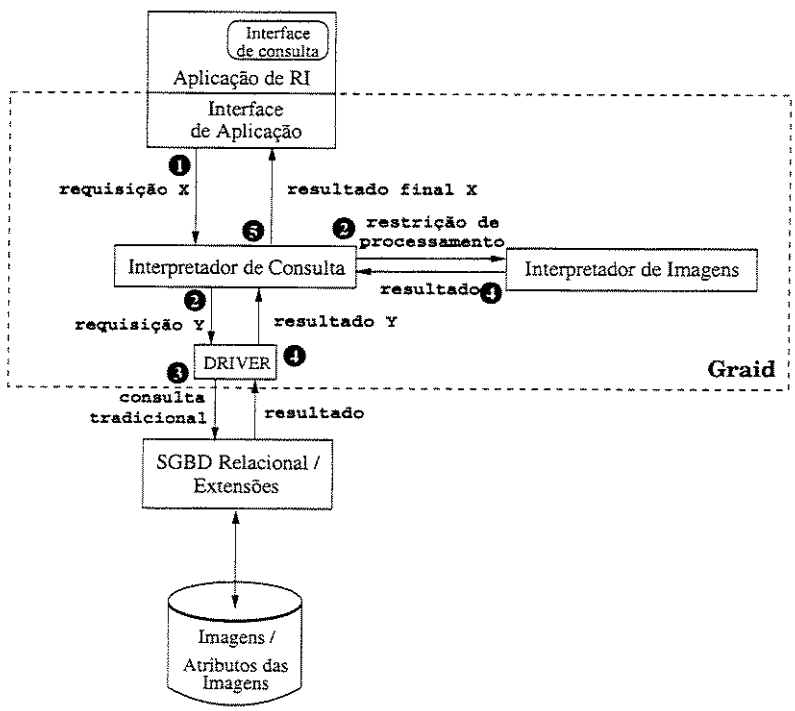

Figura 3.7: Fluxograma de uma requisição em GRAID.

Passos

1. A aplicação de RI envia uma requisição ao Interpretador de Consulta através da Interface de Aplicação.

2. Uma vez que o IC recebe a `requisição X`,

   - o filtro a analisa, separa as restrições e armazena o plano de execução na estrutura da consulta.

   - o implementador do plano de execução processa a estrutura da consulta.

   . Envia as restrições correspondentes ao Interpretador de Imagens. Cada restrição contém as operações de processamento de imagens requisitadas e as informações necessárias para o processamento.

   . Envia as requisições tradicionais ao Driver de SGBD.

3. O Driver de SGBD converte estas requisições em uma consulta na linguagem do SGBD base e as envia para este.

4. Os resultados são enviados do Interpretador de Imagens ao Interpretador de Consultas e do SGBD ao Driver.

   - O Driver de SGBD converte os resultados em resultados em GRAID e os envia ao Interpretador de Consultas.

5. O Implementador do plano de execução no IC combina os resultados e os envia à aplicação de RI através da IA.

Observe na figura 3.8 o fluxo dos dados através de GRAID para as aplicações do exemplo do Museu, da Escola de Artes e da Agência de Turismo. No esquema, o fluxo para cada aplicação tem um traço diferente. Indicados com círculos, note-se como a nova arquitetura permitiu que várias aplicações tirassem proveito do banco de dados de pinturas impressionistas e do algoritmo do histograma de cores. No exemplo, as pinturas são recuperadas pelas aplicações do Museu e da Escola de Artes, e o algoritmo de histograma de cores é usado pelas aplicações da Escola de Artes e da Agência de Turismo.

Como conseqüência, a arquitetura permite que um algoritmo de processamento de imagens possa ser usado por mais de uma aplicação e uma dada aplicação pode utilizar os algoritmos de processamento que necessitar.

3.5 Interfaces de GRAID para os desenvolvedores de aplicações de RI

Os desenvolvedores de aplicações de RI usarão implementações de GRAID. Eles não terão que se preocupar com os detalhes da recuperação (por exemplo, junção dos resultados do processamento de imagens com os resultados da consulta tradicional, quando for o caso). O SGBD escolhido pela aplicação não precisa suportar tipos de dados abstratos cujas operações precisam ser definidas pelos usuários pois GRAID oferece a capacidade para manipular as imagens e incorporar os algoritmos de processamento de imagens necessários. Uma aplicação de RI pode ir incorporando aos poucos seus algoritmos de processamento de imagens, caso o desenvolvedor esteja interessado em construir novas versões da aplicação com novas operações de processamento de imagens.

Para os desenvolvedores de aplicações de RI usarem implementações de GRAID, eles precisam conhecer previamente:

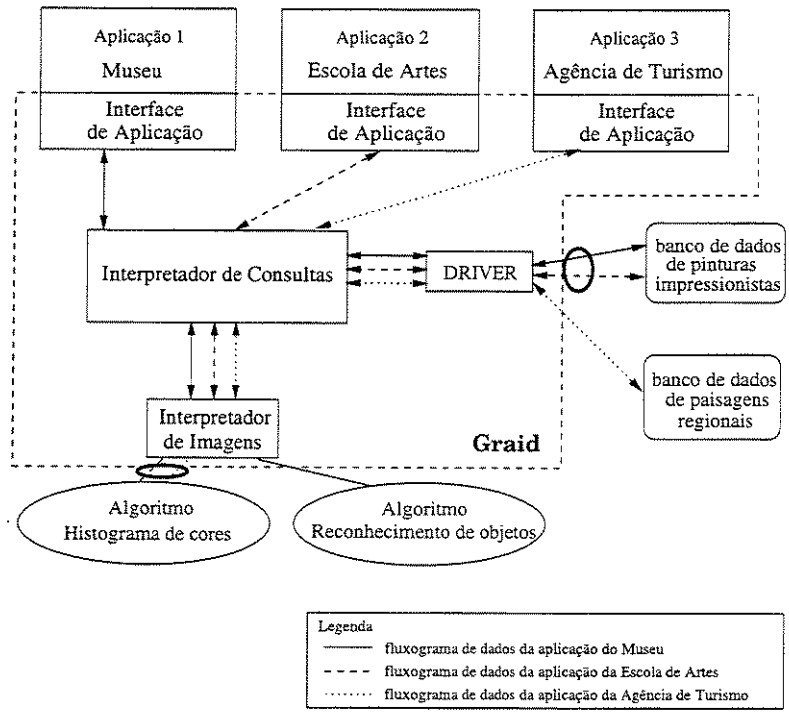

Figura 3.8: Fluxograma das aplicações de RI em GRAID.

- O modelo de dados do seu banco de dados.
- A forma em que devem ser implementados os algoritmos de processamento de imagens definidos para a aplicação.
- O mecanismo para formar as requisições de GRAID.

Adicionalmente, a aplicação precisa especificar o Driver associado ao SGBD que será utilizado.

Três componentes de GRAID são diretamente utilizados pelo desenvolvedor de aplicações.

- O repositório de algoritmos é utilizado na hora de projetar a aplicação, com ele pode conhecer os algoritmos que já existem em GRAID, a forma em que devem ser implementados os algoritmos e a estrutura das requisições.
- A interface de configuração é utilizada para registrar as implementações dos algoritmos em GRAID.
- A interface de aplicação é utilizada para implementar a aplicação de RI.

3.6 Sumário

Este capítulo apresentou GRAID, uma arquitetura para unificar as aplicações que precisam da Recuperação de Imagens sobre Bancos de Dados Relacionais e suas extensões. A chave

Aspectos	**SGBDCs**	**SPI/BD**	**SGBDEEs**	**SRIAs**	GRAID
Recuperação por atributos tradicionais	Sim	Sim	Sim	Geralmente Não	Sim
Recuperação por processamento das CIs	Não	Sim	É possível	Sim	Sim
Recuperação integrada em uma consulta	Não	Nem sempre	É possível	Geralmente Não	Sim
Sistemas de domínio específico	Não	Sim	Não	Sim	Não
Sistemas de propósito geral	Sim	Não	Sim	Não	Sim
- Com funcionalidades para imagens	Não	Não	Não	Não	Sim
- Incorporação de algoritmos de p. i. [a]	Não	Não	É possível	Não	Sim
- Reusabilidade de algoritmos de p. i.	Não	Não	Não oferece mecanismos	Não	Sim

[a] p. i. processamento de imagens

Tabela 3.1: Tabela comparativa incluindo GRAID

para a construção desta arquitetura foi a separaração da função de recuperação da função do processamento de imagens. GRAID oferece uma plataforma comum com repositórios de imagens, modelos de dados e algoritmos de processamento reusáveis. A arquitetura engloba componentes que são independentes do tipo da aplicação e oferece mecanismos para que as aplicações possam colocar e utilizar suas especificidades. O próximo capítulo apresenta o sistema WIRED (*Web Image REtrieval from Databases*), uma implementação de GRAID.

Para completar a comparação feita entre as abordagens aos SRIs descritas no capítulo 2, a tabela 3.1 mostra uma análise comparativa das características de GRAID em relação às abordagens descritas nesse capítulo. É possível observar na tabela que, diferente das outras abordagens, GRAID resolve satisfatoriamente os problemas enunciados para a recuperação de imagens.

Capítulo 4

Sistema WIRED e Aplicações Protótipos

Neste capítulo descreve-se uma primeira versão simplificada de GRAID. Este sistema foi implementado com o objetivo de testar a validade dos mecanismos básicos propostos. WIRED (*Web Image REtrieval from Databases*) é uma ferramenta que implementa algumas das funcionalidades de GRAID. Em particular, foram implementados os seis componentes de GRAID para suportar algumas das suas funcionalidades e duas aplicações protótipos que usam WIRED e um algoritmo de processamento das cores das imagens para validar o sistema.

4.1 Princípios de projeto do sistema

Além dos três princípios de projetos já definidos para GRAID, foram definidos mais quatro princípios de projeto para o sistema WIRED. Os princípios de projeto de WIRED podem ser resumidos nos itens mostrados a seguir.

- Implementar um sistema de propósito geral para as aplicações de RI.
- Permitir a integração dos tipos de dados tradicionais com as imagens.
- Facilitar a incorporação de novos algoritmos de processamento de imagens ao sistema e permitir a reusabilidade dos algoritmos já existentes.
- Ser independente do SGBD utilizado na implementação.
- Ser simples de usar.
- Definir uma interface para o acesso ao sistema através da Web.
- Acessar de forma independente de plataforma.

Para implementar o sistema WIRED e as aplicações protótipos, de forma a atender os princípios de projeto listados acima foram utilizados os seguintes elementos.

Linguagem da implementação Java. Tanto o sistema quanto as aplicações são baseados na linguagem Java [21, 22], o que permitirá uma implementação independente de plataforma e possibilitará a manipulação dos dados armazenados no BDI através da Web. O Driver de SGBD utilizado foi a interface JDBC de Java para trabalhar com SGBDs, demonstrando assim a independência de WIRED com respeito ao SGBD escolhido pelo desenvolvedor de aplicação. Atualmente, a maioria dos SGBDs relacionais disponíveis possuem uma interface JDBC.

SGBD PostgreSQL. PostgreSQL [11] é um sistema Relacional Estendido com características de Orientação a Objetos. Todo acesso ao banco de dados é através de consultas SQL. Permite também a criação de funções, tipos de dados, operadores e métodos de acesso definidos pelos usuários. No entanto, estas funcionalidades de PostgreSQL não foram utilizadas, e procurou-se sempre explorar as funcionalidades de WIRED. A arquitetura de PostgreSQL é cliente-servidor e possui uma interface JDBC. Possui capacidade para o armazenamento no banco de dados de objetos binários e um conjunto de operações para sua manipulação (*read, write, open, lseek*).

Linguagem de Consulta SQL estendida. As requisições foram definidas com a linguagem de consulta SQL com suporte para chamadas a funções. Foi implementado um subconjunto das capacidades da instrução SELECT. Mais detalhes são apresentados no apêndice B.

A comunicação entre os componentes em WIRED pode ser feita de duas formas, segundo mostram a figura 4.1 (a) e a figura 4.1 (b). Na figura 4.1 (a), o Interpretador de Imagens não precisa se comunicar com o SGBD pois o Interpretador de Consulta envia as informações necessárias para sua execução. Na figura 4.1 (b), o Interpretador de Consulta envia para o Interpretador de Imagens o que deve fazer e este último procura as informações necessárias contactando o SGBD.

WIRED implementa a comunicação entre os componentes como no item (a) da figura 4.1. A figura 4.2 mostra a estrutura geral do protótipo. As duas aplicações do protótipo utilizam o sistema WIRED para recuperar informações relacionadas com imagens.

4.2 Sistema WIRED

Nas próximas seções serão discutidos os aspectos mais relevantes da implementação de WIRED.

4.2.1 Interface de Aplicação

As operações incluídas na interface de programação para chamadas ao Interpretador de Consulta são as seguintes.

- **Iniciação**. Inicia uma instância do Interpretador de Consulta para a aplicação. O nome do driver do SGBD precisa ser especificado para executar esta operação. Neste caso, as aplicações especificam o driver de PostgreSQL para JDBC.

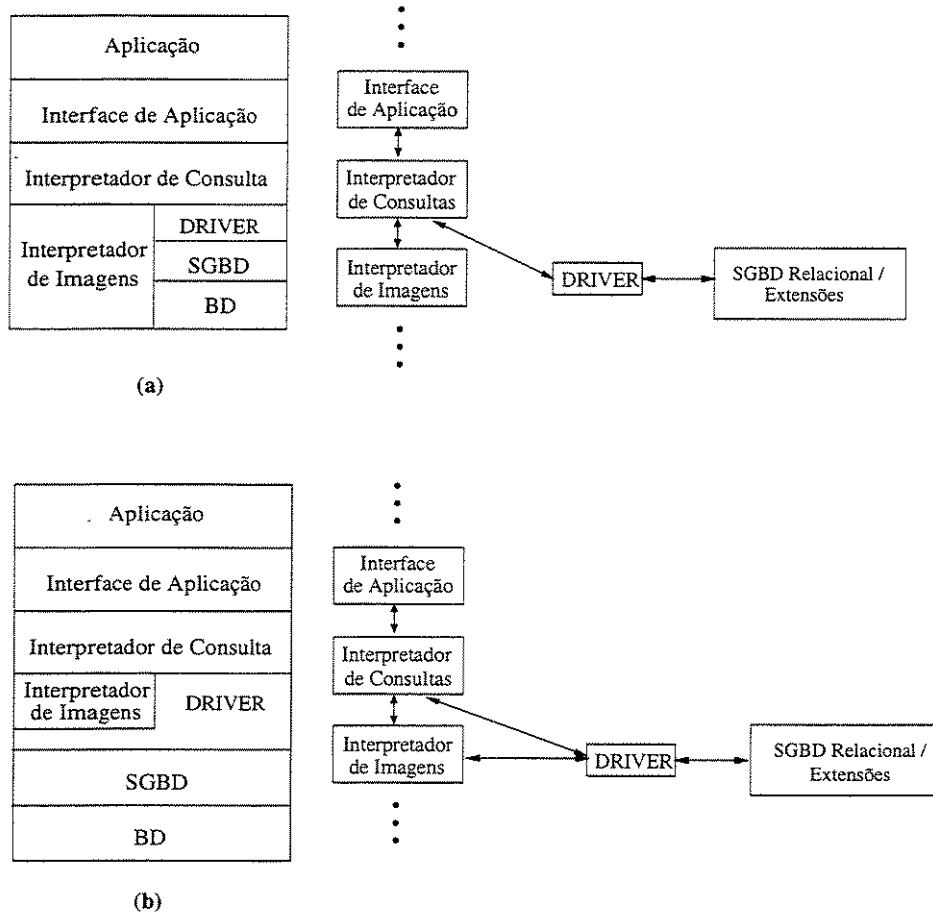

Figura 4.1: Comunicação entre os componentes em WIRED.

- **Requisição**. Esta operação é utilizada pela aplicação para enviar as requisições ao Interpretador de Consulta. A requisição é feita usando a linguagem de consulta SQL com suporte para chamadas a funções.
- **Encerramento**. Encerra a comunicação da aplicação com WIRED.

4.2.2 Interpretador de Consulta

Nesta seção são mostrados o fluxo das requisições em WIRED e dois exemplos de plano de execução das requisições.

4.2.2.1 Fluxo de requisições através de WIRED

O esquema da figura 4.3 representa o fluxograma da recuperação de uma consulta em WIRED. Uma requisição feita por um usuário é passada da aplicação a WIRED através da Interface da Aplicação. O Interpretador de Consulta recebe a requisição, decompõe a cláusula das restrições (se necessário) em cláusulas atômicas usando o FILTRO, e envia a

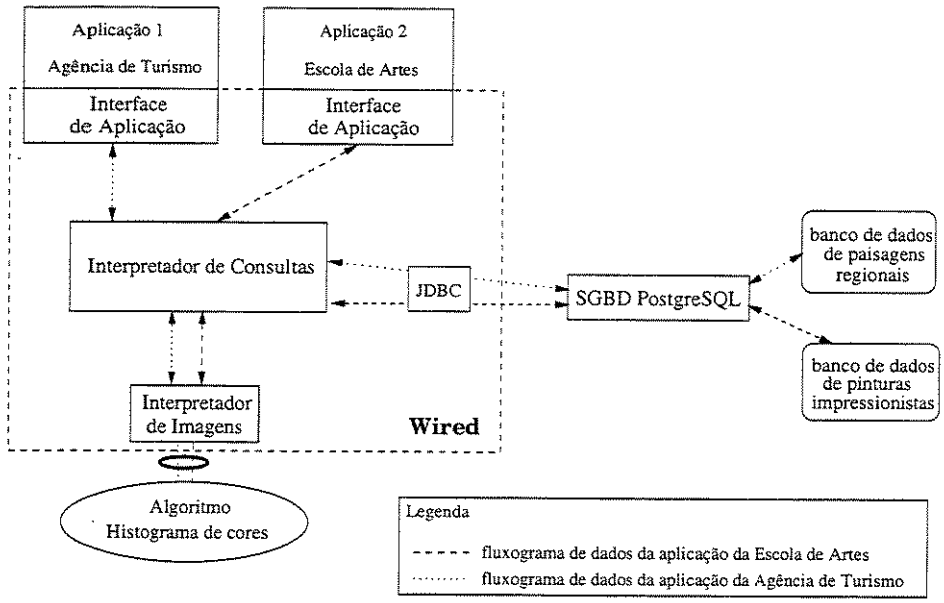

Figura 4.2: Estrutura geral do protótipo.

informação correspondente ao SGBD e/ou ao Interpretador de Imagens. No exemplo da figura, a cláusula das restrições na requisição é:

```
description=''flores'' OR Histogram(''SomeYellow'')
```

```
cláusula 1: description=''flores''
cláusula 2: Histogram(''SomeYellow'')
```

O Interpretador de Consultas identifica a primeira como uma cláusula que pode ser resolvida pelo SGBD enquanto a segunda requer um algoritmo que reconheça a cor amarela nas imagens. Logo, a primeira cláusula é enviada ao SGBD através do Driver de SGBD e a segunda ao Interpretador de Imagens junto com as informações necessárias para o processamento. Estes últimos processam a informação e devolvem os resultados ao Interpretador de Consulta que se encarrega de combiná-los e enviá-los à aplicação que está aguardando o resultado final. O plano de execução para o processamento das consultas depende dos operadores envolvidos. Por exemplo, neste caso, como o operador utilizado é o `OR`, o envio das cláusulas é simultâneo para ambos os sistemas (SGBD e I^2). Se o operador fosse `AND`, o Interpretador de Consultas seria mais eficiente se enviasse a primeira cláusula ao SGBD e o conjunto resultado desta operação fosse enviado ao Interpretador de Imagens.

4.2.2.2 Planos de execução das requisições em WIRED

As requisições são elaboradas utilizando a linguagem de consulta SQL. Suponha-se as seguintes requisições em WIRED:

1.
```
// Seleciona as fotos com flores amarelas
select foto
```

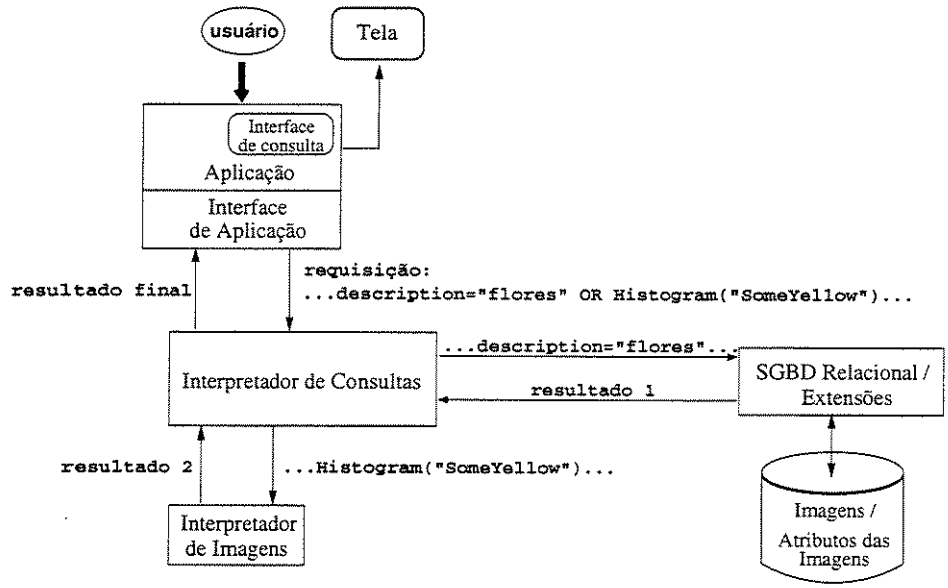

Figura 4.3: Fluxograma de uma requisição em WIRED.

```
from bd_paisagens
where description=''flores'' AND Histogram(''SomeYellow'')
```

2. `// Seleciona as fotos com flores ou com elementos da cor amarelo`
```
select foto
from bd_paisagens
where description=''flores'' OR Histogram(''SomeYellow'')
```

Dois exemplos de planos de execução são mostrados na figura 4.4. O primeiro exemplo é o plano de execução de uma consulta que utiliza o operador `AND` (requisição 1) e o segundo exemplo utiliza o operador `OR` (requisição 2).

Os passos dos planos de execução são descritos a seguir.

Passo 1 . O IC faz a filtragem das cláusulas obtendo como resultado duas cláusulas:
cláusula 1 - `description=''flores''`
cláusula 2 - `Histogram(''SomeYellow'')`

Para a primeira consulta, o plano de execução da requisição para o operador `AND` é detalhado pelos passos 2 ao 5 da figura 4.4 (a).

Passo 2 . O campo da cláusula 1 é reconhecido como atributo tradicional e a consulta enviada para o SGBD (através do Driver de SGBD).

Passo 3 . O SGBD processa a consulta e envia os resultados para o IC (através do Driver de SGBD).

Passo 4 . O IC envia o conjunto resultante do Passo 3 junto com a cláusula 2 para o I^2.
Quando o IC reconhece uma cláusula de processamento de imagens (cláusula 2), ele

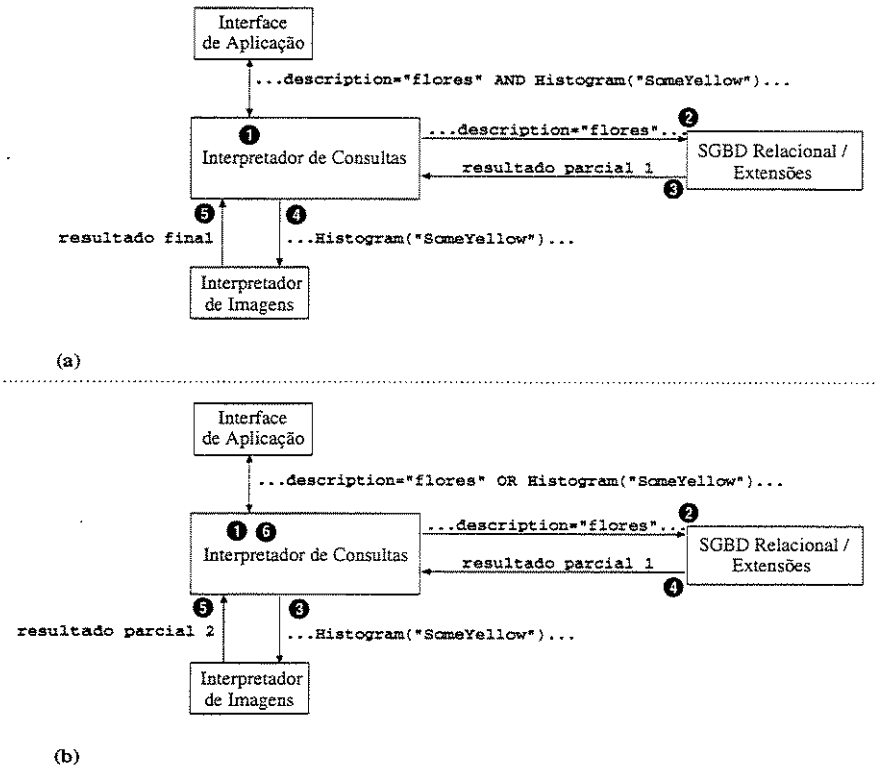

Figura 4.4: Planos de Execução. (a) operador `AND`. (b) operador `OR`.

confire se o algoritmo requisitado e os parâmetros associados são válidos. Assim, ele é capaz de determinar os dados que são necessários para o processamento e os busca no SGBD. O IC então constrói uma consulta tradicional formada pela cláusula tradicional da consulta original (caso ela exista) e uma cláusula adicional que requisita os dados necessários para o algoritmo de processamento de imagens. Dentre os dados recuperados pela consulta, aqueles relacionados com o algoritmo de processamento de imagens são posteriormente enviados ao I^2.

Passo 5 . O I^2 faz o processamento correspondente e envia os resultados para o IC. O conjunto resultante deste passo é o resultado da consulta enviada pela IA.

Para a segunda requisição, o plano de execução para o operador `OR` é como descrito nos passos 2 ao 6 da figura 4.4 (b).

Passo 2 . O campo da cláusula 1 é identificado como atributo tradicional e enviado para o SGBD.

Passo 3 . O campo da cláusula 2 é identificado como restrição de processamento e enviado para o I^2. O IC executa o mesmo procedimento do passo 4 do plano de execução do operador `AND` para obter as informações necessárias relacionadas ao processamento de imagens.

Passo 4 . O SGBD processa a consulta e envia os resultados para o IC.

Passo 5 . O I^2 faz o processamento correspondente e envia os resultados para o IC.

Passo 6 . Neste caso o IC combina os resultados parciais 1 e 2 e obtém o resultado final da consulta enviada pela IA.

A combinação dos resultados parciais acarreta um aumento do custo da performance devido a que esta operação envolve operações de junção para mostrar um conjunto de resultados sem repetições, no caso do operador `OR`.

4.2.3 Driver de SGBD

O Driver usado por WIRED é o JDBC. Em particular, cada aplicação de RI precisa especificar a interface do seu SGBD com JDBC.

4.2.4 Interpretador de Imagens

As questões introduzidas para implementar o Interpretador de Imagens são listadas a seguir.

- Como programar um algoritmo de processamento de imagens para WIRED?
- Como incluir um algoritmo de processamento ao sistema?
- Como carregar o algoritmo adequado?
- Como executar o algoritmo para obter os resultados certos?

O sistema resolve estas questões em quatro etapas.

- **Implementação de Algoritmos**. Nesta versão, os algoritmos de processamento de imagens incorporados devem ser métodos públicos com valor de retorno *boolean*.
- **Registro de Algoritmos**. Para registrar um novo algoritmo foi necessário definir uma estrutura para a tabela de processamento de imagens. Nesta versão, a tabela é um arquivo de configuração chamado de `algorithms.txt`. Este arquivo contém um nome de classe de algoritmo por linha. Os passos seguidos nesta etapa são:
  1. Copiar no diretório `wired/` que precisa ser adicionado na variável de ambiente da máquina virtual Java CLASSPATH, o arquivo `.class` onde se encontra o código do algoritmo. O sistema não permite duas classes diferentes com o mesmo nome.
  2. Modificar o arquivo de configuração, incluindo uma nova linha com o nome do arquivo que contém o algoritmo.
- **Iniciação do Sistema**. Ao se carregar o sistema, o Interpretador de Imagens lê do arquivo `algorithms.txt` todos os nomes de arquivos de algoritmos disponíveis para o sistema. Carrega as classes correspondentes, e determina os métodos com valor de retorno *boolean* que são públicos e possíveis nomes de algoritmos. Finalmente, o sistema cria um objeto da classe carregada e o armazena. Este objeto será o encarregado de executar os algoritmos de processamento de imagens implementados na classe. Nesta etapa a aplicação precisa especificar a interface de SGBD que o Driver deve carregar para estabelecer a comunicação.

- **Processamento da Requisição.** O Interpretador de Consulta determina a parte da consulta que deve ser enviada ao Interpretador de Imagens e envia uma requisição. Este último procura entre os métodos registrados o nome especificado na consulta, faz uma validação dos tipos dos parâmetros e executa o algoritmo. Se houver algum erro, o sistema o informa à aplicação. Se existirem dois métodos de algoritmos com o mesmo nome, a aplicação deverá resolver a situação colocando como prefixo do nome do método o nome da classe a que pertence a função requisitada.

As duas primeiras etapas são feitas estaticamente, enquanto as outras duas são feitas dinamicamente em tempo de execução, permitindo assim, mudanças nos códigos e nomes das classes dos algoritmos.

A forma em que está implementado o Interpretador de Imagens demonstra a capacidade para incluir novos algoritmos de processamento de imagens no sistema.

4.2.5 Repositório de Algoritmos

O Repositório de Algoritmos é uma interface gráfica que mostra as classes e as interfaces dos métodos de algoritmos já incluídos em WIRED. Além disso, oferece uma ajuda para mostrar a sintaxe da linguagem de consulta e o conjunto de instruções que o programador deve seguir para implementar seu algoritmo. A figura 4.5 mostra a interface gráfica do Repositório de Algoritmos. À esquerda da tela encontra-se a lista de classes e à direita as interfaces dos algoritmos de processamento de imagens da classe selecionada.

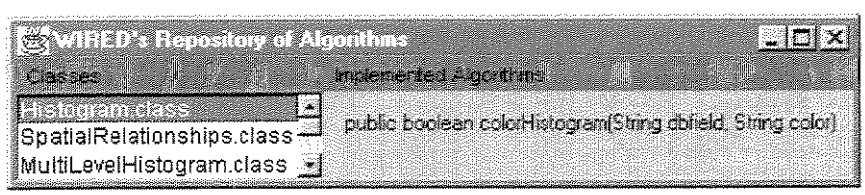

Figura 4.5: Interface do Repositório de Algoritmo

4.2.6 Interface de Configuração

A Interface de Configuração é uma interface gráfica onde o desenvolvedor pode escolher os nomes dos arquivos das classes que deseja incorporar. Este componente implementa a etapa de registro de algoritmos. Uma vez incorporados ao sistema, estes algoritmos formam parte do repositório. A figura 4.6 mostra a interface gráfica da Interface de Configuração. A janela inferior na tela é a caixa de diálogo utilizada para importar as classes.

4.2.7 Interfaces para o desenvolvedor de aplicações

Para a interação do desenvolvedor de aplicações com WIRED, foram definidas as três interfaces projetadas em GRAID.

- Repositório de Algoritmos.
- Interface de Configuração.

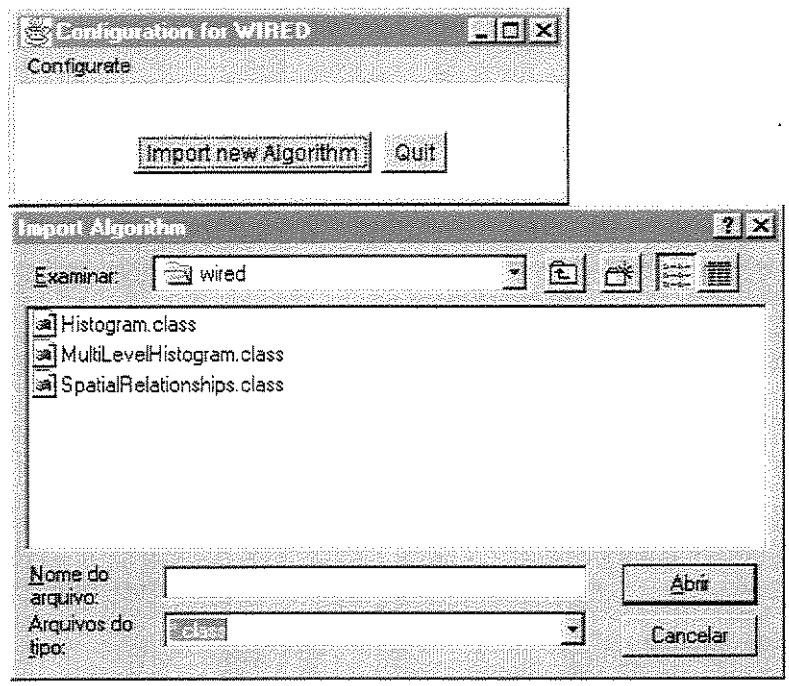

Figura 4.6: Interface do Componente de Configuração

- Interface de Aplicação.

4.2.7.1 Modo de uso de WIRED

Para programar algoritmos que respondam a este modelo, os programadores devem seguir os seguintes passos:

1. **Etapa de implementação de algoritmos.**
   A interface do Repositório de Algoritmos permite conhecer os algoritmos já implementados em WIRED.

   Para implementar novos algoritmos de processamento de imagens: os algoritmos devem ser métodos públicos e com valor de retorno *boolean* em classes Java. Os parâmetros dos métodos podem ser nomes de atributos no bancos de dados, constantes alfanuméricas, arquivos de imagens ou qualquer tipo de dados que precisar. O exemplo mostra a interface de um algoritmo de Histograma de Cores:

   `public class Histogram`
   Classe que contém o algoritmo de processamento de imagens implementado.

   ```
   public boolean colorHistogram(String dbfield, String color)
   { ... }
   ```
   `dbfield` é o nome do atributo do banco de dados que contém as características preprocessadas, no caso, `histogram`.

`color` é a cor requisitada nas imagens. No caso, são admitidos os seguintes critérios: *SomeGreen*, *SomeRed*, *SomeYellow*, *SomeBlue*, *SomeWhite*, *SomeBlack*.

2. **Etapa de registro de novos algoritmos.**
   Usar o Componente de Configuração para incorporar algoritmos em WIRED.

Invocar as operações definidas na interface de programação.

3. **Etapa de iniciação do sistema.**
   Utilizar a Interface de Aplicação para especificar o driver JDBC do SGBD utilizado pela aplicação.

4. **Etapa de envio de requisição.**
   Utilizar a Interface de Aplicação e a linguagem de consulta SQL para enviar e construir as requisições. Qualquer algoritmo de processamento de imagens disponível no repositório de WIRED pode ser incluído nas requisições.

5. **Etapa de encerramento do sistema.**
   Utilizar a Interface de Aplicação para encerrar a comunicação da aplicação com WIRED.

4.3 Aplicações Protótipos

As aplicações protótipos são duas aplicações de Recuperação de Informações relacionadas com imagens. Foram implementadas duas aplicações: Sistema de Recuperação de Paisagens da Agência de Turismo e Sistema de Recuperação de Pinturas da Escola de Artes. O objetivo foi explorar as capacidades do sistema WIRED e demonstrar a incoporação de algoritmos de processamento de imagens (neste caso, um algoritmo de histograma de cores) em WIRED e seu uso pelas duas aplicações.

4.3.1 Estruturas dos Bancos de Dados

Os bancos de dados das aplicações foram definidos de maneira simples, como mostram as tabelas 4.1 e 4.2.

Em ambas as aplicações foi introduzido um campo `histogram` para armazenar o histograma de cores para as imagens. O armazenamento desta informação do conteúdo é precomputada para cada imagem na hora da inserção da imagem no banco de dados.

4.3.2 Especificação e Processamento de Consultas

Ambos os sistemas especificaram como interface de SGBD o driver de PostgreSQL para JDBC. A estrutura das requisições foi feita utilizando a linguagem definida por WIRED. As aplicações geram os pedidos usando a especificação da linguagem de consulta de WIRED e usam a interface de programação definida por WIRED para enviar as requisições e obter os resultados. O processamento das consultas é feito por WIRED. Foi implementada uma

Sistema de Recuperação de Paisagens da Agência de Turismo

Atributo	Tipo	Descrição
filename	text	nome do arquivo de imagem importado ao banco de dados
sizebytes	int4	tamanho em bytes da imagem
entry_date	abstime	data de entrada do registro ao sistema
location	text	lugar capturado na imagem
shot_date	date	data em que foi tirada a fotografia
color	char	foto em branco e preto (B), em cores (C)
photographer	text	nome do fotógrafo
description	text	descrição breve do lugar representado na foto
histogram	text	histograma de cores
landscape	oid	identificador de objeto de PostgreSQL

Tabela 4.1: Atributos do banco de dados `dblandscapes`

Sistema de Recuperação de Pinturas da Escola de Artes

Atributo	Tipo	Descrição
filename	text	nome do arquivo de imagem importado ao banco de dados
sizebytes	int4	tamanho em bytes da imagem
entry_date	abstime	data de entrada do registro ao sistema
century	char4	século em que foi criada a obra de arte
artist	text	nome do autor da obra
título	text	título da obra
description	text	descrição breve da obra
histogram	text	histograma de cores
picture	oid	identificador de objeto de PostgreSQL

Tabela 4.2: Atributos do banco de dados `dbpictures`

classe `Histogram`. Esta classe foi incorporada a WIRED e é carregada em tempo de execução como parte do mecanismo de consulta para analisar esta informação armazenada no campo `histogram` do banco de dados.

4.3.3 Interfaces de Usuário

As duas aplicações apresentam uma interface gráfica tal como apresenta a figura 4.7. Esta figura mostra a interface de usuário do Sistema de Recuperação de Paisagens da Agência de Turismo. A interface do Sistema de Recuperação de Pinturas da Escola de Artes é similar.

Na parte superior da tela é apresentado conjunto de atributos que podem ser visualizados como resultado da consulta tais como a paisagem, a localização e descrição da paisagem, o fotógrafo e a data da foto. O usuário deve escolher os atributos que deseja examinar.

Na segunda secção (superior direita) encontram-se os critérios para restringir os resultados de recuperação. Os atributos envolvidos nas restrições foram:

- Localização. É um campo texto para colocar a região de interesse.
- Palavras chaves. É um campo texto para colocar palavras chaves associadas às paisagens que deseja recuperar.

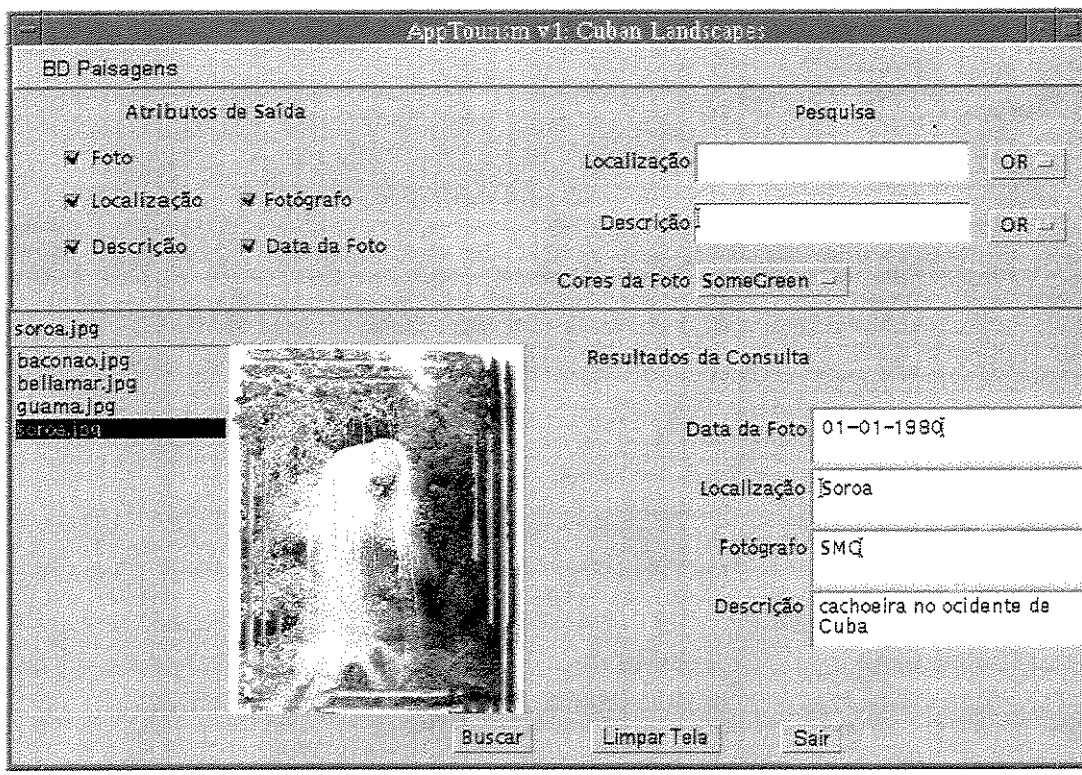

Figura 4.7: Interface de Usuário do Sistema de Recuperação de Paisagens da Agência de Turismo.

- Cores nas paisagens. É um lista com os seguintes valores para as cores das fotos: *SomeGreen*, *SomeRed*, *SomeYellow*, *SomeBlue*, *SomeWhite*, *SomeBlack*. Por exemplo, uma pessoa que deseja olhar as fotos de paisagens campestres pode escolher o item *SomeGreen*.

Clicando no botão `Buscar`, a secção inferior da tela apresenta os resultados da consulta. Esta terceira secção tem três sub-secções:

1. mostra a lista dos nomes de arquivos das imagens resultados. Os nomes de arquivos, que foram armazenados no banco de dados, agem como seletores de registros do conjunto resultado da consulta;
2. visualiza a imagem (quando for requisitada);
3. visualiza os atributos tradicionais escolhidos como atributos de saída da consulta.

4.4 Sumário

Este capítulo apresentou o sistema WIRED, uma implementação da arquitetura GRAID proposta no capítulo 3, e duas aplicações protótipos: as aplicações da Agência de Turismo e da Escola de Artes. Todas as implementações foram feitas em Java com o objetivo de oferecer um sistema independente de plataforma e possibilitar a manipulação dos dados

armazenados no BDI através da Web. Os bancos de dados foram criados em PostgreSQL e a linguagem definida foi uma extensão da SQL para suportar chamadas a funções. O objetivo foi demonstrar a capacidade de WIRED de unificar aplicações por:

- ser um sistema de propósito geral para aplicações de RI;
- ter capacidade de incorporar algoritmos de processamento ao sistema e sua execução;
- permitir reusabilidade dos algoritmos;
- independer do SGBD utilizado pela aplicação de RI.

Para isto, as aplicações foram projetadas para serem incorporadas em WIRED e conseguiram utilizar o sistema de maneira que somente os detalhes de domínio específico foram implementados por elas. Foi incorporada a WIRED uma classe que implementa um algoritmo de histograma de cores, utilizado pelas duas aplicações. A interface usada para a comunicação com o SGBD foi JDBC, em particular a interface de PostgreSQL para JDBC. Isto demonstra que WIRED independe do SGBD escolhido pelo desenvolvedor de aplicação pois uma mudança de SGBD pela aplicação somente implica a alteração da interface especificada na iniciação da comunicação com WIRED.

Foram definidos três componentes para a interface de WIRED com o desenvolvedor de aplicação: o repositório de algoritmos que mostra as interfaces dos algoritmos já implementados, a forma em que devem ser implementados os algoritmos e a estrutura das requisições; a interface de configuração que registra as classes que implementam algoritmos para o sistema; e a interface de aplicação encarregada da interação da aplicação com WIRED.

Capítulo 5

Implementação de WIRED

Este capítulo apresenta as principais classes que foram definidas para implementar WIRED e as aplicações protótipos. Descrevem-se os fluxos de controle da execução assim como os detalhes de implementação mais relevantes.

5.1 Componentes do sistema

A figura 5.1 mostra o relacionamento entre as diferentes classes que foram definidas para implementar o sistema WIRED.

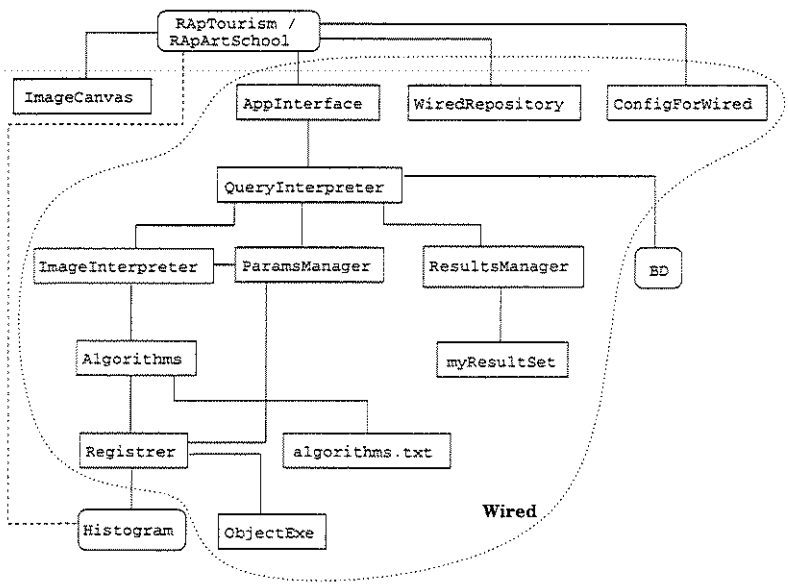

Figura 5.1: Relacionamento entre as classes de WIRED.

A seguir apresenta-se por cada componente de WIRED, uma breve descrição das classes usadas na implementação.

Repositório de Algoritmos

- `public class WiredRepository`
  A classe `WiredRepository` é um tipo de ajuda que o sistema oferece para mostrar as características que devem apresentar as aplicações programadas para WIRED. Possui uma interface que mostra a lista das classes já incluídas em WIRED. Para cada classe, mostra as interfaces dos métodos dos algoritmos implementados. Tem opções para mostrar as instruções que um programador deve seguir para implementar um algoritmo para o sistema e para mostrar a sintaxe da linguagem de consulta de WIRED.

Componente de Configuração

- `public class ConfigForWired`
  Fornece uma interface para incorporar classes de algoritmos em WIRED.

Interface de Aplicação

- `public class AppInterface`
  Esta classe fornece os métodos necessários aos programadores para interagir com WIRED.

  `public AppInterface (driver-JDBC, endereco-servidor, login-usuário, senha-usuário)`
  `{ ... }`
  Inicia um Interpretador de Consulta (`QueryInterpreter`) para a aplicação. A aplicação especifica os parâmetros iniciais para carregar o sistema (`driver-JDBC, endereco-servidor, login-usuário, senha-usuário`).

  `public ResultsManager sendRequest (consulta)`
  `{ ... }`
  Método utilizado pela aplicação para enviar as requisições (`consulta`) para o Interpretador de Consulta. A requisição é feita usando a linguagem de consulta SQL com suporte para chamadas a funções.

  `public void closeWIRED()`
  `{ ... }`
  Encerra WIRED para a aplicação. Libera a memória utilizada por WIRED.

- `public class imageTools extends Canvas`
  Contém o método `setImage` para mostrar uma imagem.

Interpretador de Consulta

- `public class QueryInterpreter`
  Faz o processamento e execução da consulta.

`public QueryInterpreter (driver-JDBC, endereco-servidor, login-usuário, senha-usuário)`
`{ ... }`
Carrega o manipulador de SGBD para JDBC (`driver-JDBC`).
Estabelece a conexão com o banco de dados no endereço `endereco-servidor` com autenticação para `login-usuário, senha-usuário`.
Inicia um Interpretador de Imagens (`ImageInterpreter`) para a aplicação.
No exemplo da Aplicação da Agência de Turismo, são especificados os seguintes parâmetros.

```
driver-JDBC: postgresql.Driver
endereco-servidor:  jdbc:postgresql:dblandscapes
```

`public ResultsManager processRequest (consulta)`
`{ ... }`
Chama ao método `Filter`.
Se existe restrição na consulta, determina se é uma chamada a uma função de processamento de imagens.
Se é uma restrição que requer processamento de imagens, executa o algoritmo para cada registro do banco de dados, buscando previamente as informações necessárias no SGBD e utilizando o método `paramsValue` de `ParamsManager`.
Senão, envia a consulta tradicional para o servidor do SGBD.
Devolve o conjunto dos resultados.

`protected void Filter (consulta)`
`{ ... }`
Analisa sintaticamente a consulta e a divide na lista de atributos de saída, restrições e tabelas. Utiliza o método `paramRecognition` de `ParamsManager`.

- `public class ResultsManager`
  Contém um conjunto de métodos para manipular os resultados da consulta. O conjunto de resultados é armazenado em uma estrutura do tipo `Vector`. Cada item do vetor é do tipo `myResultSet`.

  `public void insertResult (ResultSet rs, attributes)`
  `{ ... }`
  Insere no conjunto de resultados a tupla `rs` depois de ser convertido a `myResultSet` com o método `convertIn` de `myResultSet`.

  `public boolean moreElements ())`
  `{ ... }`
  `public myResultSet next ())`
  `{ ... }`
  Estes dois métodos são definidos como uma interface para a aplicação percorrer o conjunto resultado. Inicialmente um indicador é posicionado antes do primeiro ele-

mento do conjunto resultado.
O método `next` devolve o resultado do tipo `myResultSet` na posição do indicador.
O método `moreElements` verifica se tem mais elementos a partir da posição do indicador.

```
public boolean isEmpty ())
{ ... }
```
Verifica se o conjunto de resultados é nulo.

```
public myResultSet getElement (indice)
{ ... }
```
Devolve o resultado do tipo `myResulSet` na posição `indice` do conjunto de resultados.

- `public final class myResultSet`
  Define dinamicamente a estrutura de um item do resultado. Está formado pelos valores dos atributos da lista de saída na consulta.

```
public void convertIn (ResultSet rs, atributos)
{ ... }
```
Converte uma tupla do resultado da interface JDBC (`ResultSet rs`) ao tipo `myResultSet` escolhendo somente os atributos de saída cujos nomes dos campos do banco de dados estão armazenados em `atributos`.

```
public String getString (indice)
{ ... }
```
Devolve o valor do campo na posição `indice` da tupla resultado (`myResultSet`) como um `string`.

```
public public byte[] getBytes (indice)
{ ... }
```
Devolve o valor do campo na posição `indice` da tupla resultado (`myResultSet`) como um `array de bytes`.

- `public class ParamsManager`
  Esta classe possui métodos para reconhecer e validar os parâmetros dado a chamada a uma função. Até o momento, os parâmetros admitidos são: um nome de atributo do banco de dados ou uma constante de um dos tipos atômicos tradicionais.

```
public void paramRecognition (lista)
{ ... }
```
Reconhece o tipo de dado de cada um dos parâmetros da `lista` de parâmetros de uma chamada a função de processamento de imagens. Os tipos podem ser: um nome de atributo do banco de dados ou uma constante alfanumérica. Se um dos parâmetros

é um atributo do banco de dados, determina o tipo de dado do atributo no banco de dados.

```
public void paramsValue (ResultSet rs)
{ ... }
```
Se um dos parâmetros do algoritmo de processamento de imagens é um atributo do banco de dados, este método coloca o valor do atributo na tupla atual `rs` na posição desse parâmetro na chamada do algoritmo antes da execução do algoritmo.

Interpretador de Imagens

- `public class ImageInterpreter`
  Encarregada de registrar os métodos de processamento de imagens e executá-los quando for requisitado.

```
public ImageInterpreter ()
{ ... }
```
Chama ao método `loadFunctions` da classe `Algorithms`.

```
public boolean executeFunction (nomeAlgoritmo,parametros)
{ ... }
```
Busca a referência do objeto onde está armazenado o método `nomeAlgoritmo` com os `parametros` chamando a `callFunction` de `Algorithms`.
Executa o método e retorna para `processRequest` de `QueryInterpreter` o valor de retorno do método; no caso, `true` ou `false`.

- `public class Algorithms`
  Contém os métodos para a manipulação dos métodos de processamento de imagens.

```
public void loadFunctions
{ ... }
```
Carrega as classes cujos nomes se encontram em `algorithms.txt`. Para cada classe carregada, cria um objeto dessa classe.

```
public void registerFunctions(classe)
{ ... }
```
Usa os mecanismos de introspeção para obter os métodos públicos declarados em `classe` que possuem como valor de retorno um `boolean`, supondo que os métodos que satisfazem estas condições são algoritmos de processamento de imagens. Insere no Registro de classes a referência à classe do algoritmo com a lista ordenada dos nomes dos métodos que são algoritmos usando `putIn` da classe `Register`.

```
public ObjectExe callFunction (nomeAlgoritmo,tiposParametros)
{ ... }
```

Verifica se existe um método registrado que responda ao `nomeAlgoritmo` e parâmetros `tiposParametros` do método requisitado fazendo uma chamada ao método `lookForMethod` de `Registrer`.

- `public class Registrer`
  Contém um conjunto de métodos para manipular o Registro de classes onde estão armazenadas as referências das classes. O conjunto de resultados é armazenado em uma estrutura do tipo `Vector`.

  `public void putIn (algoritmo)`
  `{ ... }`
  Insere no Registro de classes o objeto `algoritmo` do tipo `ObjectExe` com a referência à classe do algoritmo e a lista ordenada dos nomes dos métodos que são algoritmos já determinada em `registerFunctions` da classe `Algorithms`.

  `public ObjectExe lookForMethod (methodName, paramstype)`
  `{ ... }`
  Busca no Registro de classes um método com nome `methodName` e com o tipo de dado de cada parâmetro conforme especifica a lista `paramstype`. Devolve o objeto do tipo `ObjectExe` que contém a referência à classe desse algoritmo.

- `public class ObjectExe`
  Define a estrutura de um registro. Está formado pelas referências ao objeto da classe e ao método de processamento de imagens. A referência ao objeto da classe é armazenada para sempre executar o método sobre um mesmo objeto, de outra forma, criaria um objeto por cada requisição. Os métodos `getObjectExe` e `setObjectExe` foram definidos para manipular um objeto deste tipo.

- `algorithms.txt`
  Arquivo de configuração que tem em cada linha, o nome de uma classe que implementa um algoritmo de processamento de imagens.
  Por exemplo:
  `Histogram`
  `SpatialRelationships`
  `MultiLevelHistogram`

Aplicações

Alguns fragmentos de código Java das aplicações que apresenta o uso da interface de programação de WIRED e a manipulação dos resultados estão detalhadas no apêndice C.

- `public class RApArtSchool`
  Aplicação da Escola de Arte para fazer recuperações sobre um banco de dados de Obras de Arte (`dbpictures`).

- `public class RApTourism`
  Aplicação da Agência de Turismo para fazer recuperações sobre um banco de dados de Paisagens (`dblandscapes`).

- `public class Histogram`
  Classe que contém o algoritmo de processamento de imagens implementado.

  ```
  public boolean colorHistogram(String dbfield, String color)
  { ... }
  ```
  `dbfield` é o nome do atributo do banco de dados que contém as características preprocessadas, no caso, `histogram`.
  `color` é a cor requisitada nas imagens. No caso, são admitidos os seguintes critérios: *SomeGreen*, *SomeRed*, *SomeYellow*, *SomeBlue*, *SomeWhite*, *SomeBlack*.

Os pacotes e classes de Java utilizados na implementação de WIRED foram:

`java.sql.*`: interfaces `ResultSet` e `Statement` para a manipulação de consultas e resultados interagindo com o banco de dados.

`java.util.*`: classe `Vector` para armazenar o conjunto resultado e a referência aos métodos de processamento de imagens.

`java.lang.*`: classe `Class` para obter informação sobre um objeto de classe, seus métodos, etc.

`java.lang.reflect.*`: classe `Method` para obter informação sobre o próprio método tais como parâmetros, valor de retorno, etc.

5.2 Etapas da execução de WIRED

Na seção 4.2.4 do capítulo 4 apresentaram-se quatro etapas para a utilização de WIRED. As duas últimas etapas, a etapa de Iniciação do Sistema e a de Processamento da Requisição, estão envolvidas na execução de WIRED. A seguir são mostrados os esquemas de fluxo de controle destas duas etapas considerando o relacionamento das classes e métodos descritos acima.

Iniciação do sistema

A figura 5.2 mostra o fluxo de controle na etapa de iniciação do sistema. As flechas estão associadas a chamadas a funções. A classe origem da flecha faz uma chamada à função na classe destino da flecha.

Processamento da requisição

A figura 5.3 mostra o fluxo de controle no momento de processamento de uma requisição.

5.3 Detalhes de implementação de WIRED

Alguns detalhes da implementação de WIRED considerados relevantes são listados a seguir.

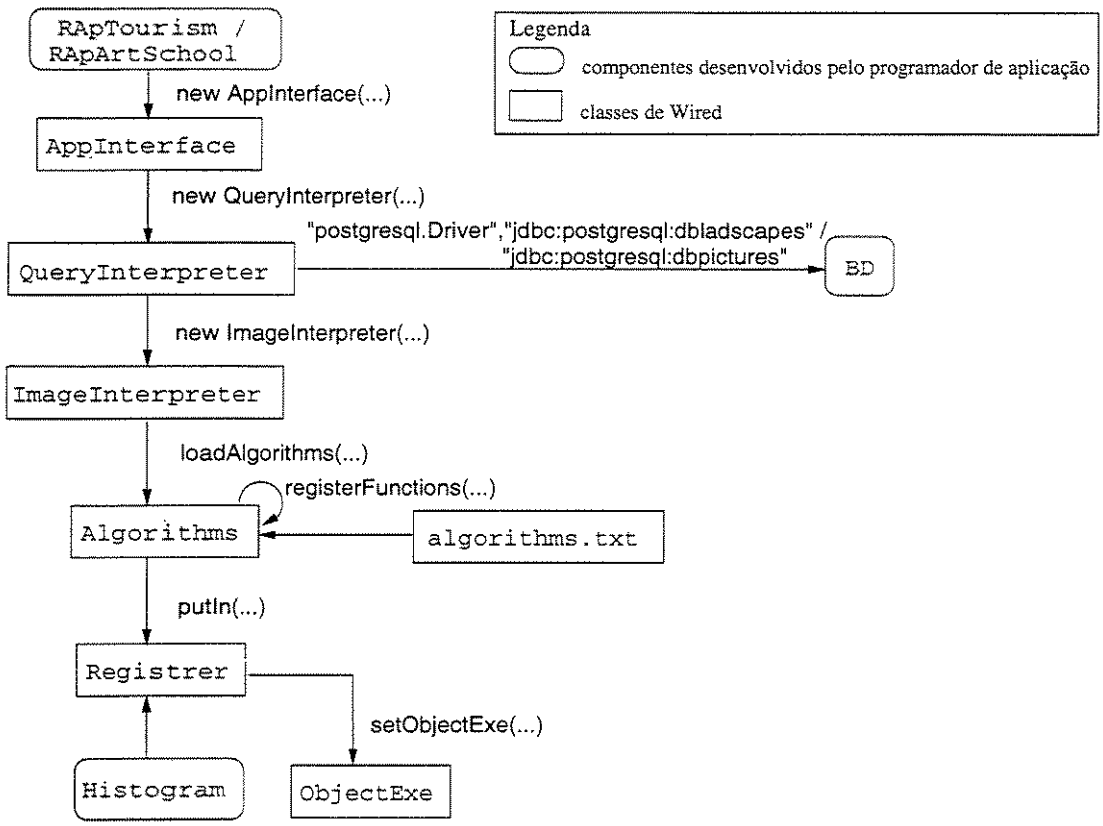

Figura 5.2: Fluxo de controle na iniciação do sistema WIRED.

Manipulação das imagens

O PostgreSQL armazena as imagens como um objeto binário. Através da interface de JDBC deste sistema, pode obter-se um vetor de bytes que contém a imagem em forma binária. Utilizando a classe `ImageCanvas`, este vetor de bytes é traduzido e mostrado como imagem.

Mecanismos de introspeção de Java

As classes `Class` e `Methods` já definidas em Java permitem o acesso a detalhes internos de classes e seus métodos. Elas foram utilizadas para implementar os seguintes métodos:

- `loadFunctions` de `Algorithms`: cria um objeto para cada nome de classe obtido da leitura do arquivo `algorithms.txt`.
- `registerFunctions` de `Algorithms`: uma vez instanciadas, é possível obter os métodos cujo tipo do valor de retorno é `boolean` e os tipos de dados dos parâmetros destes métodos.
- `executeFunction` de `ImageInterpreter`: executa o método requisitado como algoritmo de processamento de imagens que faz parte de um dos objetos de classe criados.

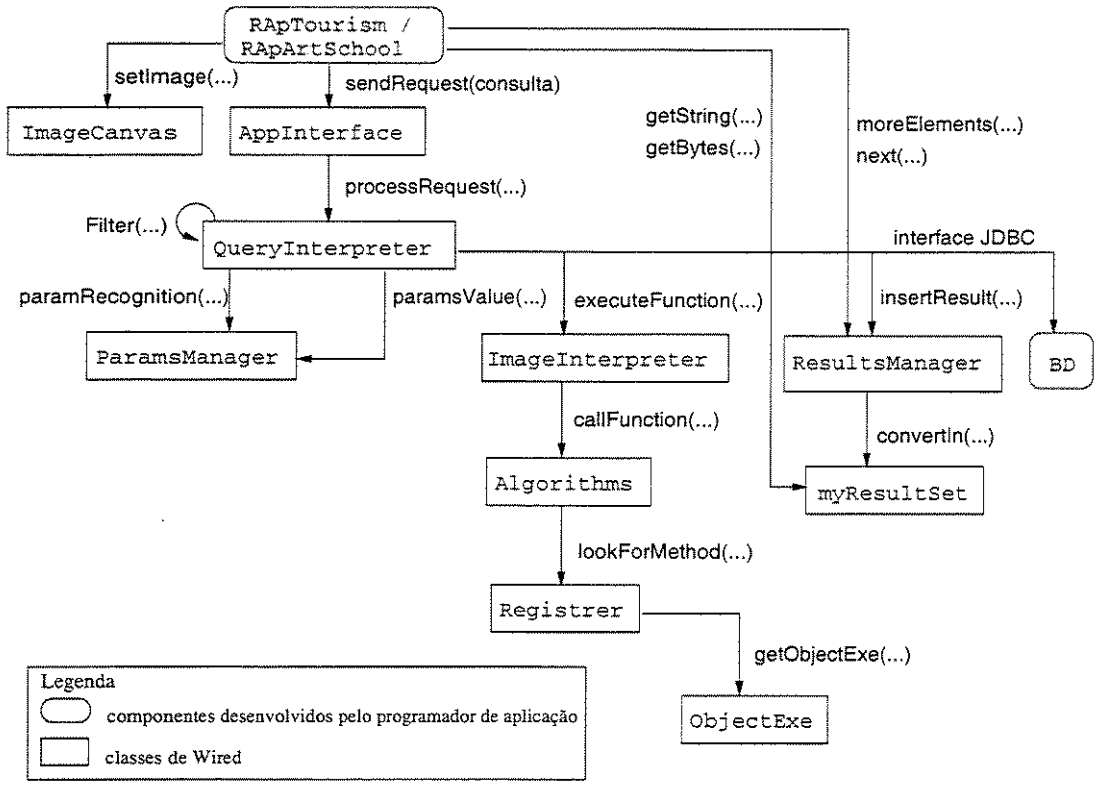

Figura 5.3: Fluxo de controle no processamento de uma requisição em WIRED.

Estrutura de dados para os resultados

Foi preciso criar uma estrutura para armazenar os resultados das requisições (`myResultSet`) e um manipulador (`ResultsManager`) pois a interface oferecida por JDBC (`ResultSet`) não permite alterações na interface que oferece para os resultados de uma consulta. Como conseqüência, não era possível incluir ou excluir os resultados do processamento das imagens. A nova estrutura `myResultSet` armazena os atributos de saída de uma consulta. O manipulador `ResultsManager` insere em uma estrutura deste tipo cada tupla do conjunto de resultados.

Tipos de dados dos objetos binários

Cada SGBD atribui um nome diferente para o tipo de dado associado aos objetos binários (BLOBs). Por exemplo, o nome do tipo dos objetos binários em PostgreSQL é `oid` e em MySQL é `BLOB`. Isto dificulta a identificação do tipo de dado de um determinado atributo. Porém, WIRED precisa controlar esta situação, principalmente porque as imagens requerem um tratamento especial de leitura. Uma primeira solução foi colocar o símbolo _ no início do nome dos atributos de tipo BLOB. WIRED reconhece estes atributos como imagens.

5.4 Sumário

Neste capítulo foram descritos o conjunto de classes que foi definido para implementar WIRED e as aplicações protótipos, além dos detalhes de implementação mais relevantes de WIRED.

Para programar aplicações que respondam ao modelo proposto, os programadores necessitam da interface de programação (`AppInterface`). No caso de WIRED, esta interface consta de três métodos fundamentais: o construtor `AppInterface` que deve ser chamado explicitamente para estabelecer a conexão com o Interpretador de Consulta e o banco de dados; o método `sendRequest` que fornece os mecanismos necessários para obter os resultados da consulta requisitada e o método `closeWIRED` que encerra explicitamente o sistema. Outras interfaces complementárias foram implementadas para facilitar o trabalho com WIRED: as classes `WiredRepository` e `ConfigForWired`.

Capítulo 6

Conclusões e Trabalho Futuro

A recuperação de imagens é uma das operações mais importantes na maioria das aplicações de BDIs. A maioria dos problemas na recuperação de imagens são originadas pelas necessidades de domínio específico e por isto as soluções encontradas são limitadas somente a um conjunto de domínios de aplicação.

Como uma tentativa para solucionar esta situação, neste trabalho definiu-se uma arquitetura geral para unificar as aplicações que necessitem fazer recuperações de imagens sobre bancos de dados relacionais e suas extensões. O objetivo principal é oferecer ao desenvolvedor de aplicações meios simples para adequar o sistema às suas necessidades particulares e, caso seja necessário, estendê-lo com novas funcionalidades.

Esta arquitetura chamada GRAID foi motivada pela falta de modelos de propósito geral para recuperação de imagens para aplicações de BDIs. Por outro lado, a nossa noção de banco de dados de imagens é estreitamente relacionada à noção de bancos de dados no sentido tradicional. Por essa razão, esta abordagem é definida sobre os SGBD Relacionais e suas extensões, aproveitando assim as facilidades oferecidas por estes sistemas na manipulação de dados tradicionais e oferecendo para eles funcionalidade na interpretação e recuperação de imagens. Para recuperar informação relacionada às imagens, um SGBD precisa conter mais funcionalidades que as tradicionais. Isto inclui algum suporte para os algoritmos de processamento do conteúdo das imagens. O princípio de projeto de GRAID é manter a independência de domínio através da separação da função de recuperação da função do processamento de imagens.

A arquitetura aproveita seu propósito geral para criar uma plataforma comum às aplicações de RI com repositórios, modelos de dados e algoritmos de processamento de imagens. GRAID engloba componentes que são independentes do tipo da aplicação e oferece mecanismos para que as aplicações possam colocar e utilizar suas especificidades.

A contribuição prática deste trabalho foi uma implementação simplificada de GRAID, o sistema WIRED, e duas aplicações protótipos: Sistema de Recuperação de Paisagens da Agência de Turismo e Sistema de Recuperação de Pinturas da Escola de Artes. Todas as implementações foram feitas em Java para oferecer um sistema independente de plataforma e possibilitar a manipulação dos dados armazenados no BDI através da Web. Os bancos de dados foram criados em PostgreSQL e a linguagem definida foi a SQL com suporte para chamadas a funções. O objetivo foi demonstrar a capacidade de incorporar algoritmos de processamento ao sistema, a reusabilidade dos algoritmos e a independência de WIRED com respeito ao SGBD utilizado pela aplicação. Foi incorporado um algoritmo de histograma de

cores a WIRED que foi utilizado pelas duas aplicações. A interface usada para a comunicação com o SGBD foi o JDBC. O trabalho detalha também uma interface de programação que permite implementar aplicações de RI que respondam ao modelo proposto.

Muito ainda pode ser feito para que GRAID se torne uma plataforma cada vez mais atraente para os programadores de aplicações de BDIs. Vários pontos de pesquisa foram identificados para serem resolvidos em GRAID. Dentre eles encontram-se a definição de uma linguagem de consulta e o desenvolvimento de mecanismos de indexação como parte da arquitetura de recuperação. Novos modelos de dados e estratégias de recuperação estão sendo estudados. GRAID pode tornar-se um SGBD de Imagens com a incorporação de um modelo de dados adequado e de outras operações de gerenciamento de imagens. Pesquisas em andamento estão sendo encaminhadas neste sentido.

Apêndice A

Glossário

Nesta seção descrevem-se brevemente alguns dos termos que foram usados durante o trabalho. Às vezes um mesmo conceito é associado a termos diferentes. Este apêndice fornece um conjunto de termos consistentes baseados em trabalhos já existentes. O principal objetivo é oferecer ao leitor maior clareza no vocabulário utilizado para uma melhor compreensão do trabalho.

Sigla	**Significado**
BDI	Banco de Dados de Imagens
BLOB	Binary Large OBjects
CI	Característica da Imagem
GRAID	Generic Retrieval Architecture for Image Databases
IA	Interface de Aplicação
IC	Interpretador de Consulta
I^2	Interpretador de Imagens
JDBC	Java DataBase Connectivity
ODBC	Open DataBase Connectivity
QBE	Query By Example
RI	Recuperação de Imagens
SGBD	Sistema de Gerenciamento de Bancos de Dados
SGBDR	Sistema de Gerenciamento de Bancos de Dados Relacional
SGBDI	Sistema de Gerenciamento de Bancos de Dados de Imagens
SQL	Structured Query Language
SRI	Sistema de Recuperação de Imagens
WIRED	Web Image REtrieval from Databases

Termo	**Significado**
aplicação de BDI	Aplicação que trabalha com BDIs.
aplicação de RI	SRI que usa como base BDI.
atributos externos à imagem	Atributos de uma imagem que não descrevem o seu conteúdo. Por exemplo: data de criação, autor.
atributos semânticos	Atributos de uma imagem que se derivam do seu conteúdo.
características da imagem	Elementos do conteúdo das imagens que podem ser processados. Por exemplo: cor, textura, forma e relações espacias dos objetos na imagem.
dados atômicos	Dado cujo tipo de dado é alfanumérico, refere-se aos dados dos SGBDs tradicionais.
dados complexos	Dados cuja estrutura não é compreensível pelo SGBD. Para manipular este tipo de dados é essencial métodos especiais para compreender as semânticas e executar operações sobre eles. Estes dados são referenciados também como: *byte string*, *long field* e BLOB.
dados estruturados	Dados heterogêneos sobre um objeto que precisam ser armazenados e recuperados juntos. Similar à noção das estruturas nas linguagens de programação.
recuperação de imagens	Recuperação de imagens e seus atributos associados. A consulta pode ser baseada em CIs, atributos semânticos e atributos externos à imagem.
sistema de recuperação de imagens (SRI)	Sistema cuja principal operação de manipulação dos dados imagens é a recuperação.

Apêndice B

Especificação da Linguagem de Consulta de WIRED

Foi implementado um subconjunto da instrução SQL SELECT com extensões para suportar chamadas a funções. Uma consulta em GRAID possui a seguinte estrutura:

```
SELECT <ATRIBS> FROM <TABELAS> [WHERE <CONDS>]

<ATRIBS> : atributo [, <ATRIBS>]

<TABELAS> : tabela [, <TABELAS>]

<CONDS> : atributo <OP> <VALOR> | <FUNCAO> ( <PARAMETROS> )
          /* cláusula tradicional ou cláusula de processamento */

<OP> : > | < | = | >= | <= | !=

<VALOR> : int | char | date | string | boolean

<FUNCAO> : [nome_classe.]nome_funcao

<PARAMETROS> : parametros [, <PARAMETROS>] | nulo

atributo = nome de um atributo no banco de dados

tabela = nome de uma tabela do banco de dados

nome_classe = nome da classe a que pertence o algoritmo de processamento de imagens

nome_funcao = nome de uma função de processamento de imagens carregada no sistema

parametros = atributo | constante | parametros [, parametros]

nulo = nenhum atributo

constante = constante de tipo <VALOR>
```

Apêndice C

Código das Aplicações

Para entender melhor como os diferentes métodos da interface de programação são utilizados na implementação, esta seção mostra com um exemplo como é programada uma aplicação para WIRED.

Uso da Interface de Programação de WIRED

```
{ ...
try {
        // iniciação
        AppInterface appint = new AppInterface("postgresql.Driver",
                        "jdbc:postgresql:dblandscapes",user,password);
...
        // envio da requisição
        // ''rm'' estrutura de dados do conjunto resultado
        ResultsManager rm = appint.sendRequest("SELECT location,description
             FROM landscapes WHERE colorHistogram(histogram,\"SomeGreen\")");
...
        // encerrando conexão
        appint.closeWIRED();
... }
```

Manipulação dos Resultados

```
{ ...
        // ler cada item ''rs'' do conjunto resultado ''rm''
        myResultSet rs;
        // se ''rm'' tem resultados, percorrê-lo
        if (!rm.isEmpty()) {
            while (rm.moreElements()) {
                rs = rm.next();
                // mostrar os resultados
                System.out.println("Location: " + rs.getString(1));
                System.out.println("Description: " + rs.getString(2)); }
            rm.close();
        }
        else
            System.out.println("Results: None");
... }
```

Bibliografia

[1] Constantin Arapis e Dimitris Thanos. Managing telemeetings. Electronic commerce objects, Centre Universitaire d'Informatique, University of Geneva, Julho 1998.

[2] D. Batory, J. Barnett, J. Garza, K. Smith, K. Tsukuda, B. Twichell, e T. Wise. Genesis: an extensible database management system. *IEEE Transactions on Software Engineering*, 14(11):1711–1729, Novembro 1988.

[3] Anne Brink, Sherry Marcus, e V.S. Subrahmanian. Heterogeneous multimedia reasoning. *IEEE Computer*, 28(9):33–39, Setembro 1995.

[4] Joseph A. Busch, Jim Blackaby, Robin Dowden, Joseph Busch, e Beth Sandore. Digital libraries in museums and galleries. Em *DL'97: Proceedings of the 2nd ACM International Conference on Digital Libraries*, Panels, página 268, 1997.

[5] Michael J. Carey, Daniel Frank, M. Muralikrishna, David J. DeWitt, Goetz Graefe, Joel E. Richardson, e Eugene J. Shekita. The architecture of the exodus extensible dbms. Em *Proceedings of IEEE/ACM International WorkShop on Object-Oriented Database Systems*, páginas 52–65, Setembro 1986. Pacific Grove, CA.

[6] Don Chamberlin. *A Complete Guide To DB2 Universal Database.* Morgan Kaufmann Publishers, Julho 1998.

[7] Rames Elmasry e Shamkant B. Navathe. *Fundamentals of Database Systems.* Addison-Wesley, 1994.

[8] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, e Peter Yauker. Query by image and video content: The qbic system. *IEEE Computer*, 28(9):23–31, Setembro 1995. IBM Research Division.

[9] Yihong Gong. *Intelligent Image Databases. Towards Advanced Image.* Kluwer Academic Publishers, 1998.

[10] R. Gray. Content-based image retrieval: Color and edges. Technical report, Department of Computer Science, Darmouth College, 1995.

[11] The PostgreSQL Development Group. Postgresql programmer's guide. Ed. by Thomas Lockhart, Fevereiro 1998. University of California, Berkeley.

[12] V. N. Gudivada e V. V. Raghavan (Guest Eds.). Content-based image retrieval systems. *IEEE Computer- Special Issue*, 28:18–22, Setembro 1995.

[13] Venkat N. Gudivada. *A Unified Framework for Retrieval in Image Databases.* PhD thesis, University of Southwestern Louisiana, Layayette, LA, 1993.

[14] Venkat N. Gudivada, V. V. Raghavan, e K. Vanapipat. Unified approach to data modeling and retrieval for a class of image database applications. Em S. Jajodia e V. S. Subrahmanian, editores, *Multimedia Database Systems: Issues and Research Directions*, páginas 37–78. Springer-Verlag, 1995.

[15] Venkat N. Gudivada e Vijay V. Raghavan. Pictorial retrieval systems: A unified perspective and research issues.

[16] Antonin Guttman. R-trees: A dynamic index structure for spatial searching. Em *Proceedings of ACM-SIGMOD International Conference on Management of Data, Boston*, páginas 47–57, Junho 1984.

[17] Denis Lee, Ron Barber, Wayne Niblack, Myron Flickner, Jim Hafner, e Dragutin Petkovic. Complex queries for a query-by-content image database. Technical report, IBM Research Division, Fevereiro 1994.

[18] Hongjun Lu, Beng-Chin Ooi, e Kian-Lee Tan. Efficient image retrieval by color contents. Em *Proceedings of 1^{th} International Conference ADB-94. Applications of Databases*, páginas 95–108. Lecture Notes in Computer Sciences 819, 1994.

[19] R. Machiraju, A. Gaddipati, e R. Yagel. Wavelet based automatic identification of structures in medical datasets. Technical report, Ohio State University, 1996. OSU-CISRC-2/96-TR09.

[20] Mehrotra e Gary. Similar-shape retrieval in shape data management. *IEEE Computer*, 28(9):57–62, Setembro 1995.

[21] Merlin e Conrad Hughes, Michael Schoffner, e Maria Winslow. *Java Network Programming.* Manning Publications Co., 1997.

[22] Patrick Naughton. *The Java Handbook.* Osborne McGraw-Hill, 1996.

[23] K.C. Nwosu, B. Thuraisingham, e P. Bruce Berra. *Multimedia Database Systems. Design and Implementation Strategies.* Kluwer Academic Publishers, 1996.

[24] Virginia E. Ogle e M. Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, Setembro 1995.

[25] A. Pentland, Rosalind W. Picard, e S. Sclaroff. Photobook: Tools for content-based manipulation of image databases. Em *Proceedings of SPIE Conference on Storage and Retrieval of Image and Video Databases II*, páginas 34–47, San Jose, CA, fevereiro 1994.

[26] E. Petrakis e C. Faloutsos. Similarity searching in large image databases. Technical report, Department of Computer Science, University of Maryland, 1994. CS-TR-3388.

[27] Frederick E. Petry. *Fuzzy Databases. Principles and Applications.* Kluwer Academic Publishers, 1996.

[28] Rosalind W. Picard, Thomas P. Minka, e Martin Szummer. Modeling user subjectivity in image libraries. Em *Proceedings of the IEEE International Conference on Image Processing (ICIP'96)*, volume 2, páginas 777–780, Lausanne, Switzerland, Setembro 1996.

[29] Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, e Henry Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays. Em Michael Cohen, editor, *SIGGRAPH 98 Conference Proceedings*, Annual Conference Series, páginas 179–188. ACM SIGGRAPH, Addison Wesley, Julho 1998.

[30] Christian Rauber, Joe O Ruanaidh, e Thierry Pun. Secure distribution of watermarked images for a digital library of ancient papers. Em *DL'97: Proceedings of the 2^{nd} ACM International Conference on Digital Libraries*, Digital Scholarship, páginas 123–130, 1997.

[31] Christian Rauber, Peter Tschudin, Serguei Startchik, e Thierry Pun. Archival and retrieval for large image databases: Application to an historical watermarks archive. Em *Proceedings of the IEEE International Conference on Image Processing (ICIP'96)*, volume 2, Lausanne, Switzerland, Setembro 1996.

[32] Gholamhosein Sheikholeslami e Aidong Zhang. Feature visualization and analysis for image classification and retrieval. Em *Proceedings of 2^{nd} International Conference on Visual Information Systems, San Diego*, Dezembro 1997. Department of Computer Science, State University of New York at Buffalo.

[33] G. Sheikoleslami, A. Zhang, e L. Bian. Geographical image classification and retrieval. Em *Proceedings of the 5^{th} International Workshop on Advances in Geographic Information Systems (GIS-97)*, páginas 58–61, New York, Novembro 1997. ACM Press.

[34] Abraham Silberschatz, Henry F. Korth, e S. Sudarshan. *Database System Concepts*. McGraw-Hill, 3^{rd} Edition, 1997.

[35] Rohini K. Srihari. Automatic indexing and content-based retrieval of captioned images. *IEEE Computer*, 28(9):49–56, Setembro 1995.

[36] Michael Stonebraker e Greg Kemnitz. The postgres next-generation database management system. *Communication of the ACM*, 34(10), Outubro 1991.

[37] Gregory Y. Tang. Management system for an integrated database of pictures and alphanumerical data. *Computer Graphics and Image Processing*, 16(3):270–286, Julho 1981.

[38] Mark C. Tsai e Kenneth L. Melmon. Digital library for education and medical decision making. Em *DL'98: Proceedings of the 3^{rd} ACM International Conference on Digital Libraries*, páginas 311–312, 1998.

[39] M. E. J. Wood, N. W. Campbell, e B. T. Thomas. Iterative refinement by relevance feedback in content-based digital image retrieval. Em *Proceedings of ACM Multimedia 98*, páginas 13–20, Bristol, UK, Setembro 1998. ACM.

[40] Yichun Xie e George D. Graettinger. An integrated ARCVIEW expert system for analyzing contaminated sediments in the great lakes basin. Em *Proceedings of 1996 ESRI User Conference*, Palm Springs, California, 1996.

[41] Randy Jay Yarger, George Reese, and et al. *MySQL and mSQL*. O'Reilly and Associates, Maio 1999.