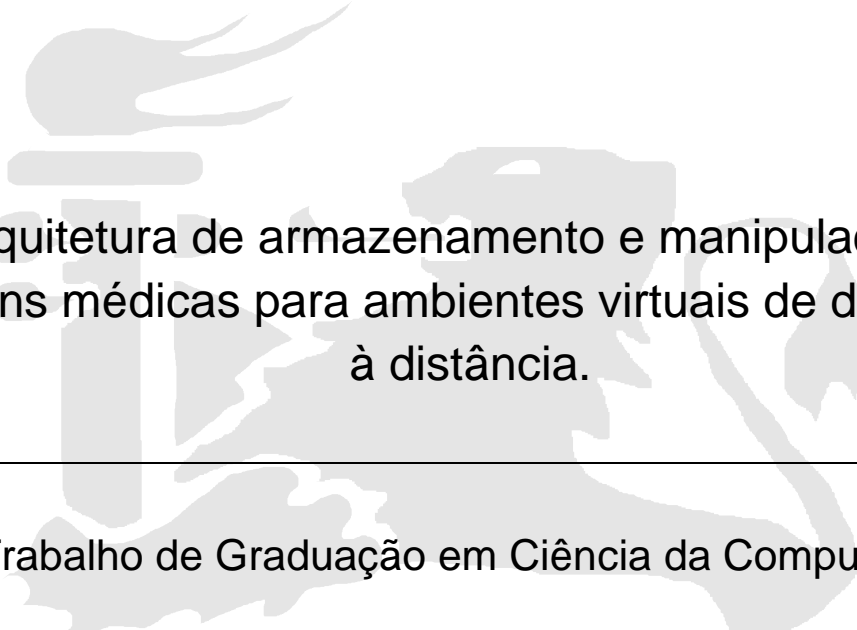


Universidade Federal de Pernambuco

Centro de Informática



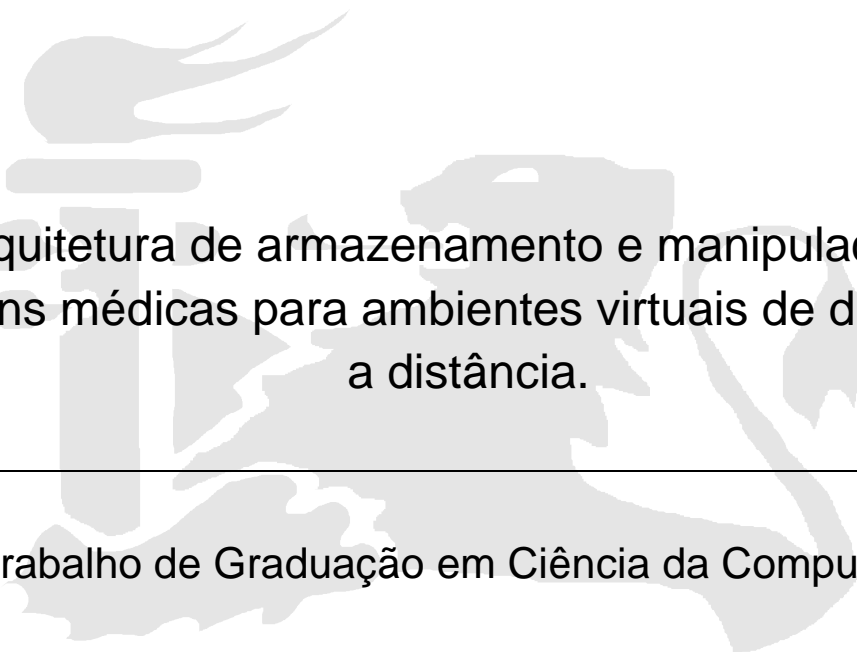
Arquitetura de armazenamento e manipulação de
imagens médicas para ambientes virtuais de diagnóstico
à distância.

Trabalho de Graduação em Ciência da Computação

Márcio Dias Costa

Recife, 25/06/2010

Universidade Federal de Pernambuco
Centro de Informática



Arquitetura de armazenamento e manipulação de
imagens médicas para ambientes virtuais de diagnóstico
a distância.

Trabalho de Graduação em Ciência da Computação

Monografia apresentada ao Centro de Informática da
Universidade Federal de Pernambuco, como requisito parcial
para obtenção do Grau de Engenheiro da Computação.

Orientador: *Fernando da Fonseca de Souza*

Co-Orientador: *Everton Rodrigues da Silva*

FOLHA DE APROVAÇÃO

Arquitetura de armazenamento e manipulação de
imagens médicas para ambientes virtuais de diagnóstico
a distância.

Trabalho de Graduação em Ciência da Computação

Banca Examinadora:

Fernando da Fonseca de Souza – Orientador

Francisco Luiz dos Santos – Avaliador

AGRADECIMENTOS

Agradeço primeiramente a Deus por me proporcionar saúde para realização de todo meu curso e deste trabalho.

Agradeço a minha família pelo apoio durante a realização do trabalho, e durante toda minha caminhada na Universidade.

Agradeço a minha namorada e amigos que ajudaram direta ou indiretamente na realização deste trabalho.

Agradeço ao meu orientador Fernando da Fonseca de Souza e meu co-orientador Everton Rodrigues da Silva, pela contribuição e orientação no desenvolvimento do trabalho.

RESUMO

A utilização de ambientes virtuais para diagnóstico médico a distância, baseado na cooperação entre profissionais, revelou-se como uma excelente alternativa para que a assistência à saúde supere parte das limitações apresentadas em seu modelo mais clássico. Dentre as novas técnicas de apoio ao diagnóstico, o uso de imagens vem se destacando pela sua ampla utilização e intensa expansão; pela complexidade da tecnologia aplicada e rápida renovação dessa tecnologia; e, finalmente, pelos resultados que oferece. Desta forma, um ambiente virtual com finalidade de dar suporte ao diagnóstico médico, deve ter entre seus componentes básicos, recursos para manipulação de imagens. O objetivo deste trabalho é propor uma arquitetura para armazenamento e manipulação de imagens médicas para atender às necessidades de um ambiente virtual de diagnóstico a distância.

Palavras-chave: Imagens médicas, ambiente virtual, arquitetura.

ABSTRACT

The virtual environment using for distance medical diagnosis based on the cooperation between professionals shown to be an excellent alternative to help health assistance to overcome part of the limitations from its more classical model. Amongst the new techniques for supporting diagnosis, image based ones have getting broad attention for its ample use and intense expansion; for the complexity of underlying technology and its fast renewal; and, finally, for the results that it can offer. Thus, a virtual environment aimed at supporting medical diagnosis must provide resources for image manipulation amongst its basic components. The main goal of this work is to present an architecture for storage and manipulation of medical images to fulfill the needs of a virtual environment for distance diagnosis.

Key words: Medical images, virtual environment, architecture

LISTA DE FIGURAS

Figura 1 - Tela principal do sistema (HENRIQUE NETO; OLIVEIRA; VALERI, 2010).	15
Figura 2 – SRIS-HC – Módulo I – Pesquisa de Exames	17
Figura 3 – SRIS-HC - Módulo II - Autenticação.....	17
Figura 4 – SRIS-HC - Módulo III - Consulta por Similaridade.....	18
Figura 5 - Principais partes presentes na extensão do protocolo DICOM (HENRIQUE NETO; OLIVEIRA; VALERI, 2010).	24
Figura 6 - Arquitetura do Oracle Multimídia DICOM (PELSKI, 2010).	36
Figura 7 - Objeto ORDDicom (PELSKI, 2010).....	38
Figura 8 - Imagem DICOM armazenada no banco de dados (PELSKI, 2010).	38
Figura 9 - Arquitetura Servlet	43
Figura 10 - Hierarquia Servlet.	44
Figura 11 - Exemplo método post.	44
Figura 12 - Exemplo do método <i>getReader()</i>	45
Figura 13 - Clico de uma Servlet.	46
Figura 14 - Exemplo do método <i>doFilter()</i>	47
Figura 15 - Arquitetura proposta.....	51
Figura 16 – Classes do pacote jdbc.	53
Figura 17 – Classes do pacote modelo.	54
Figura 18 - Classes do pacote dao.....	54
Figura 19 - Classes do pacote Filtro.....	56
Figura 20 - Classes do pacote servlet	56
Figura 21 - Modelo lógico do banco de dados.....	58
Figura 22 - Código do procedimento PRO_GERA_THUMB	59
Figura 23 - Código do procedimento PROC_INSERE_EXAME.	60
Figura 24 - Código do procedimento PROC_EXPORTA_DICOM.....	60
Figura 25 - Código do procedimento PROC_EXPORTA_THUMB.....	60
Figura 26 - Código do procedimento PROC_RETORNA_EXAMES.	61
Figura 27 - Código do procedimento PROC_RETORNA_EXAMES_SIMILARES. ...	62
Figura 28 - Gráfico comparativo entre o uso do PROC_RETORNA_EXAMES e API Java DICOM.....	63

Figura 29 - Tela de autenticação.....	64
Figura 30 - Tela de navegação do sistema.	65
Figura 31 - Tela de cadastro de médicos.	65
Figura 32 – Primeira tela do cadastro de exames.	66
Figura 33 – Segunda tela do cadastro de exames.	66
Figura 34 - Tela de pesquisa de exames.	67
Figura 35 - Tela de detalhamento dos exames.	67
Figura 36 - Tela de exames similares.	68

SUMÁRIO

FOLHA DE APROVAÇÃO.....	iii
AGRADECIMENTOS	iv
RESUMO.....	v
ABSTRACT	vi
LISTA DE FIGURAS	vii
1. INTRODUÇÃO	11
1.1 OBJETIVOS	13
1.1.1 OBJETIVO GERAL.....	13
1.1.2 OBJETIVOS ESPECÍFICOS	13
1.1.3 ESTRUTURA DO TRABALHO	13
2. ESTADO DA ARTE.....	15
2.1 ARMAZENAMENTO DE IMAGENS MÉDICAS COM O INTERBASE	15
2.2 SRIS-HC – SISTEMA DE RECUPERAÇÃO DE IMAGENS SIMILARES.....	16
2.3 RESUMO DAS CARACTERÍSTICAS.....	19
3. FUNDAMENTAÇÃO CONCEITUAL E METODOLOGIA DO TRABALHO	20
3.1 ESTUDO DE IMAGENS DIGITAIS.....	21
3.1.1 COMPACTAÇÃO.....	21
3.1.2 FORMATOS DE IMAGENS DIGITAIS ESTÁTICAS.....	21
3.1.3 FORMATOS DE IMAGENS DIGITAIS DINÂMICAS.....	22
3.1.4 O PADRÃO DICOM.....	22
3.2 ESTUDO DO ARMAZENAMENTO DE IMAGENS EM UM SGBD.....	29
3.2.1 VANTAGENS.....	30
3.2.2 DESVANTAGENS	31
3.2.3 CONSIDERAÇÕES	32
3.3 ESTUDO DO ORACLE 11g MULTIMÍDIA DICOM.....	33
3.3.1 REPOSITÓRIO DE MODELOS DE DADOS DICOM	34

3.3.2	ARQUITETURA.....	35
3.3.3	ARMAZENANDO O CONTEÚDO DICOM.....	37
3.3.4	MÉTODOS DO OBJETO ORDDicom.....	39
3.3.5	RECUPERAÇÃO DE IMAGEM POR CONTEÚDO	40
3.3.6	ORACLE MULTIMÍDIA DICOM JAVA API.....	41
3.4	ESTUDO DA TECNOLOGIA WEB.....	42
3.4.1	SERVLETS.....	42
3.4.2	JAVA SERVER PAGES (JSP).....	48
3.4.3	VANTAGENS NA UTILIZAÇÃO DE JSP E SERVLETS	48
3.5	METODOLOGIA EMPREGADA.....	49
4.	ARQUITETURA PROPOSTA E PROTÓTIPO	50
4.1	ARQUITETURA PROPOSTA.....	50
4.1.1	TECNOLOGIAS PROPOSTAS.....	50
4.1.2	FUNCIONAMENTO DA ARQUITETURA	50
4.2	DESENVOLVIMENTO DO PROTÓTIPO.....	52
4.2.1	RECURSOS DE HARDWARE E SOFTWARE	53
4.2.2	ESTRUTURA E FUNCIONAMENTO DAS CLASSES	53
4.2.3	TABELAS DO BANCO DE DADOS.....	58
4.2.4	PROCEDIMENTOS DO BANCO DE DADOS	59
4.2.5	DESEMPENHO DA API JAVA DICOM.....	63
4.2.6	FUNCIONALIDADES IMPLEMENTADAS	64
5.	CONCLUSÃO	69
5.1	TRABALHOS FUTUROS	69
	REFERÊNCIAS.....	70

1. INTRODUÇÃO

O avanço tecnológico tem viabilizado a melhoria dos serviços de saúde em todo o mundo, no qual se pode destacar o uso da computação como auxílio ao diagnóstico e troca de informações médicas. A atividade clínica sempre procura por diagnósticos eficientes e precisos, e, dentre as ferramentas de auxílio ao diagnóstico médico, o uso de imagens é uma das áreas mais promissoras da medicina moderna. Graças ao progresso da radiologia e a melhoria de novas metodologias como ultrassom e ressonância magnética, as imagens se tornaram uma excelente e essencial ferramenta de diagnóstico médico (HENRIQUE NETO; OLIVEIRA; VALERI, 2010) (CARITÁ et al., 2006).

Atualmente, grande parte dos exames é digitalizada, e precisa ser armazenada em uma base de dados para posterior consulta. No entanto, para que as imagens sejam usadas como base de informação para futuros diagnósticos, precisam ser recuperadas de forma eficiente, com mecanismos de filtros que permitam agrupá-las por patologia e/ou informações relevantes dos pacientes (HASEGAWA; AIRES, 2007). Diversos trabalhos são desenvolvidos com o objetivo de conseguir melhores formas de armazenamento, recuperação e manutenção dessas imagens. Esses trabalhos são impulsionados pelo aumento e a consolidação no uso de imagens médicas nas práticas profissionais e de ensino. Devido à disponibilidade de recursos de armazenamento e processamento, as imagens de exames dos equipamentos mais modernos já são digitais, e as demais imagens não digitalizadas passam pelo processo de digitalização. Esse processo tem evoluído bastante na redução de perdas, e se tornou uma forma bastante segura de preservação de acervos médicos (CARRARE et al., apud SILVA, Everton, 2010).

Apesar dos avanços na área de diagnóstico por imagem, o acesso aos tratamentos de saúde ainda possui limitações, como a dificuldade de acesso do paciente a médicos mais distantes, baixa velocidade na troca de informações, dificuldade de comunicação entre os médicos para compartilhamento de casos clínicos, dentre outras. Com a capacidade de sanar tais problemas, os ambientes virtuais para diagnóstico médico a distância se tornaram uma excelente opção para dinamizar a assistência à saúde, quebrando fronteiras com o uso de novas tecnologias. O avanço no uso de novas tecnologias, principalmente de recursos de

telecomunicação, para melhorar a assistência à saúde constitui a base da telemedicina (SILVA, Everton, 2010).

Em 1993, foi criado um padrão de imagens e informações chamado DICOM (Digital Imaging and Communication in Medicine) (DICOM, 2010). Esse padrão visa aumentar a interoperabilidade entre diferentes fontes de imagens médicas, facilitando seu armazenamento e comunicação. O armazenamento padronizado das imagens possibilita a criação de um ambiente virtual de diagnóstico médico a distância, no qual as imagens vão ser organizadas, catalogadas e disponibilizadas para consultas médicas (HENRIQUE NETO; OLIVEIRA; VALERI, 2010). Devido à necessidade de armazenamento, os ambientes virtuais de diagnóstico devem fazer uso de Banco de dados e não somente tecnologias de telecomunicações. Esses sistemas devem permitir fácil acesso e conter informações dos pacientes, doenças, e registros de exames. O compartilhamento das informações permite que médicos em locais geograficamente distantes dêem seu diagnóstico, melhorando a qualidade do serviço de saúde (SILVA, Everton, 2010).

Um sistema virtual de diagnóstico a distância possui algumas características essenciais para seu sucesso e viabilidade de uso. Dentre elas, podem-se destacar a segurança no armazenamento, transmissão sem perda de informações, autenticação, autorização, e prevenção contra perdas. Para que essas características sejam cumpridas, as informações devem ser armazenadas em um banco de dados após sua transmissão. Desta forma podem ser recuperadas sempre que necessário. Outra funcionalidade necessária é um mecanismo de autenticação, que assegura a identidade do usuário e as restrições de acesso ao conteúdo que possa visualizar. (SILVA, Everton, 2010).

A construção de um ambiente virtual requer um estudo das tecnologias de armazenamento e comunicação existentes, escolhendo de forma precisa as ferramentas adequadas para cada finalidade. O avanço dos sistemas de banco de dados, que passaram a dar suporte ao armazenamento e manipulação de imagens, aliado ao surgimento da web, foi definitivo para viabilizar a construção desses ambientes.

Este trabalho se propõe a elaborar uma arquitetura baseada nas tecnologias mais recentes de banco de dados e sites web, para orientar a construção de um ambiente virtual de diagnóstico médico a distância com ênfase em imagens médicas.

1.1 OBJETIVOS

Nesta seção serão apresentados os objetivos do trabalho.

1.1.1 OBJETIVO GERAL

Propor uma arquitetura para armazenamento e manipulação de imagens médicas que atenda às necessidades de um ambiente virtual de diagnóstico.

1.1.2 OBJETIVOS ESPECÍFICOS

- Definir a melhor alternativa de armazenamento de imagens para um ambiente virtual de diagnóstico a distância;
- Definir a melhor alternativa de desenvolvimento da interface do ambiente virtual com o usuário, levando em consideração a escolha da solução de banco de dados;
- Desenvolver um protótipo de um sistema de armazenamento e manipulação de imagens médicas para um ambiente virtual de diagnóstico a distância, a fim de evidenciar a viabilidade da integração entre as tecnologias propostas.

1.1.3 ESTRUTURA DO TRABALHO

Este trabalho está dividido nos seguintes capítulos:

- O capítulo 2 apresenta uma análise de dois dos principais sistemas de armazenamento de imagens médicas encontrados na literatura. O capítulo tem como objetivo a análise de aspectos em comum, e as diferenças entre os sistemas.
- O capítulo 3 apresenta a metodologia utilizada para o desenvolvimento do trabalho e a fundamentação teórica das tecnologias necessárias na construção de um sistema de armazenamento e manipulação de imagens médicas.

- O capítulo 4 apresenta a arquitetura proposta para um sistema de armazenamento e manipulação de imagens médicas para um ambiente virtual de diagnóstico a distância e a metodologia de desenvolvimento do protótipo.
- O quinto e último capítulo descreve as conclusões tiradas do trabalho e apresenta as perspectivas de trabalhos futuros relacionados.

2. ESTADO DA ARTE

Neste capítulo serão abordados dois dos principais sistemas existentes na área de armazenamento e visualização de imagens médicas, bem como sua arquitetura e quais tecnologias utilizaram.

O estudo desses sistemas foi relevante para o desenvolvimento da arquitetura proposta neste trabalho, tendo em vista que permitiu tanto análise de técnicas utilizadas em sistemas de diagnóstico médico, como a visualização do que ainda é necessário fazer para contribuir com a criação de ambientes virtuais de diagnóstico a distância.

2.1 ARMAZENAMENTO DE IMAGENS MÉDICAS COM O INTERBASE

Projeto que desenvolveu um sistema de armazenamento e visualização de imagens JPEG (JPEG, 2010). O sistema também possui armazenamento DICOM, mas, sua visualização não é possível. O sistema de banco de dados utilizado é o Interbase 6.0 (INTERBASE, 2010) que é um sistema de banco de dados relacional (SGBDR) e possui como forma de armazenamento de imagens o formato BLOB - Binary Large Object (HENRIQUE NETO; OLIVEIRA; VALERI, 2010). A Figura 1 mostra a tela principal do sistema.

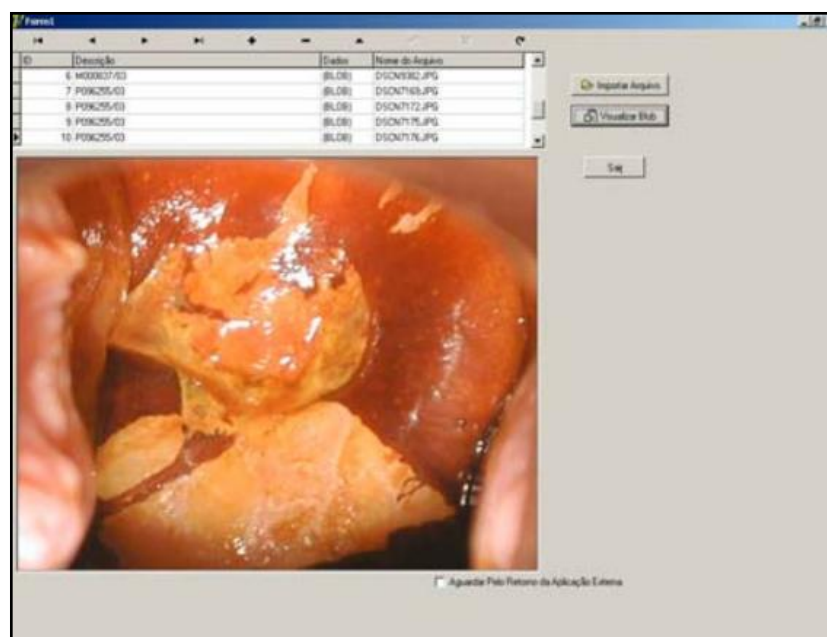


Figura 1 - Tela principal do sistema (HENRIQUE NETO; OLIVEIRA; VALERI, 2010).

2.2 SRIS-HC – SISTEMA DE RECUPERAÇÃO DE IMAGENS SIMILARES

O SRIS-HC é um sistema de recuperação de imagens similares que foi desenvolvido com a finalidade de demonstrar a viabilidade de recuperação de imagens por conteúdo (Rosa, et al 2010). O estudo utilizou como fonte de dados exames do Hospital das Clínicas da Faculdade de Medicina de Ribeirão Preto da Universidade de São Paulo (HCFMRP/USP). O Hospital já possuía um sistema de Laudo Eletrônico, que é um sistema de informação em Radiologia (RIS) e contém todas as informações textuais de exames radiológicos dos pacientes. O SRIS-HC foi desenvolvido como uma extensão ao sistema de Laudo Eletrônico, a fim de possibilitar a consulta e recuperação de imagens por similaridade (Rosa, et al 2010).

No sistema as imagens são recuperadas através da extração de características extraídas das mesmas. Sendo as características de distribuição de cor, distribuição de brilho da imagem, presença de formas e texturas, as mais utilizadas. Essas informações são valores de alta dimensionalidade. Devido a isso, o sistema de indexação deve está preparado para esse tipo de dados. Para que seja possível essa comparação, são utilizadas duas técnicas de extração: os histogramas tradicionais e métricos (Rosa, et al 2010).

O processo de armazenamento no banco de dados é dividido em duas etapas: a primeira é a descompactação dos arquivos DICOM e a extração das informações relevantes; na segunda etapa, as informações textuais e as imagens são persistidas nos respectivos campos e tabelas do banco de dados. O formato que as imagens são inseridas é o formato gbdi, que é um formato nativo do Oracle 8.0 (ORACLE 8g, 2010). Visando agilizar o processo de armazenamento das imagens, a extração das características das mesmas para armazenamento no banco pode ser feito posteriormente à sua inserção, esse processo é chamado de armazenamento parcial (Rosa, et al 2010).

SRIS-HC MÓDULO I – Pesquisa de Exames

Esse módulo, cuja tela é mostrada na Figura 2, é semelhante ao módulo existente no sistema de Laudo Eletrônico, com o diferencial de apresentar fotos em miniatura dos exames.

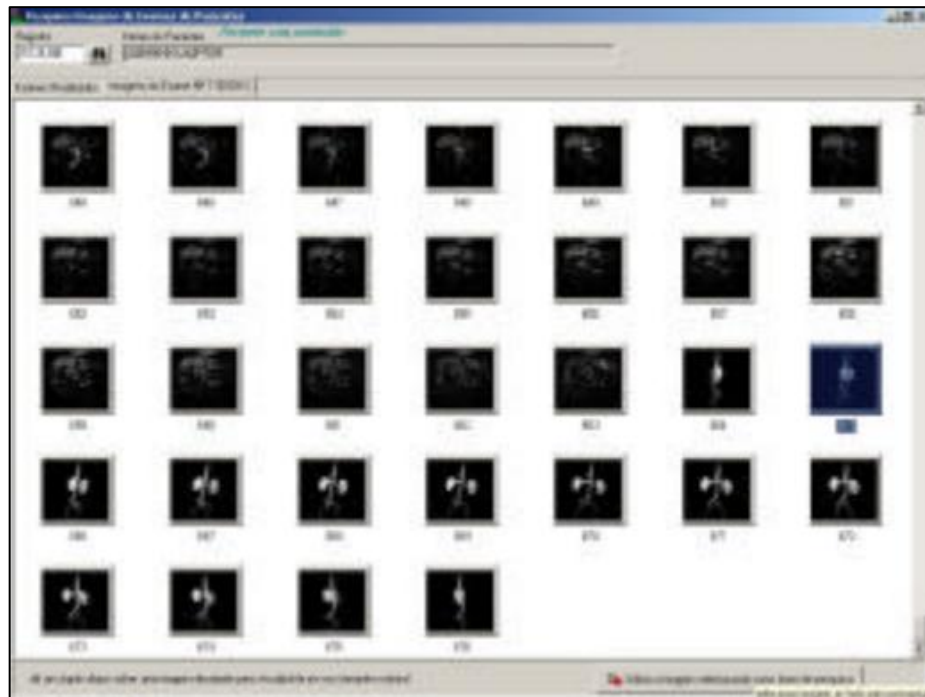


Figura 2 – SRIS-HC – Módulo I – Pesquisa de Exames

SRIS-HC - MÓDULO II - Autenticação

Esse módulo, cuja tela é mostrada na Figura 3, é o responsável pela segurança do sistema, que necessita de um *login* de usuário e senha para a conexão do usuário.



Figura 3 – SRIS-HC - Módulo II - Autenticação

SRIS-HC - MÓDULO III – Consulta por Similaridade

Esse módulo (Figura 4) é o núcleo do sistema SRIS-HC e é o que permite fazer consultas por similaridade das imagens médicas. O usuário deve inserir uma imagem para que o sistema retorne suas semelhantes, de acordo com parâmetros fornecidos previamente pelo mesmo.

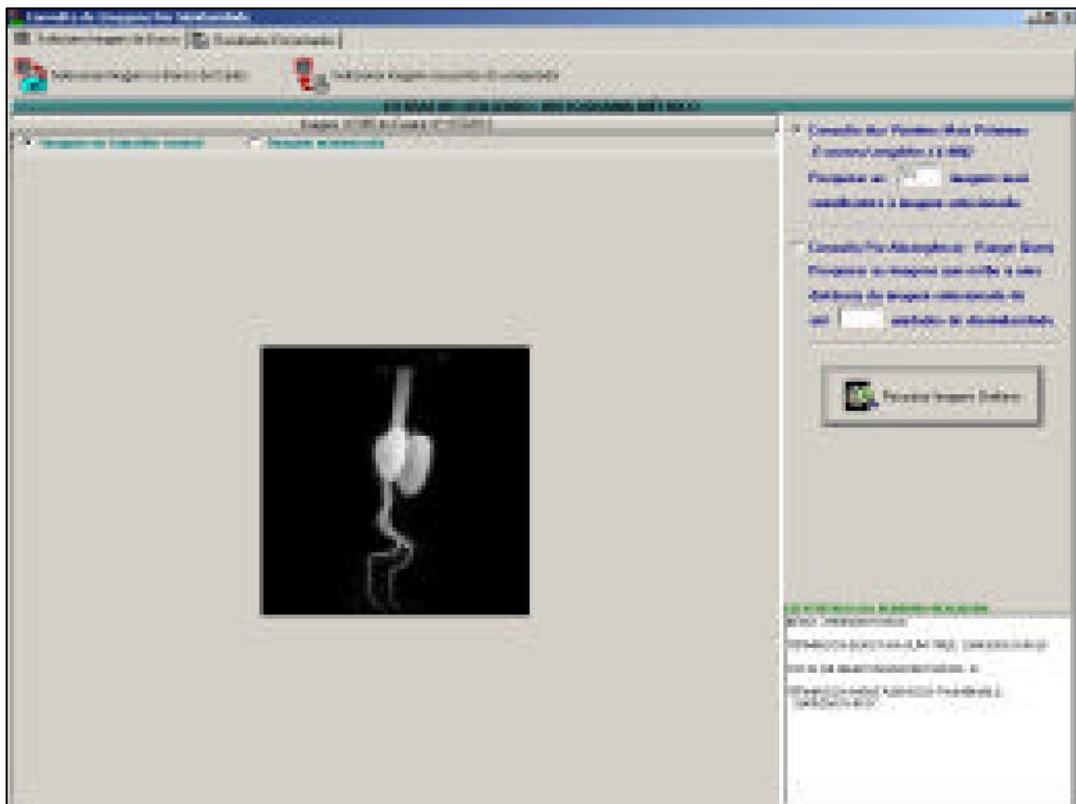


Figura 4 – SRIS-HC - Módulo III - Consulta por Similaridade

2.3 RESUMO DAS CARACTERÍSTICAS

Nesta seção será apresentado um quadro com as principais características críticas ao funcionamento de um sistema de armazenamento de imagens médicas. O quadro 1 apresenta estas características, o SRIS-HC corresponde ao “Sistema de recuperação de imagens similares”, e o INTERBASE corresponde ao “Armazenamento de Imagens médicas com o Interbase”.

Quadro 1 – Resumo das principais características dos sistemas apresentados.

Característica	SRIS-HC	INTERBASE
Autenticação do Usuário.	Possui	Não Possui.
Recuperação de Imagens por conteúdo.	Possui	Não Possui
Armazenamento de imagens no formato DICOM	Não Possui	Possui
Visualização de <i>thumbnails</i> dos exames armazenados.	Possui	Não Possui

Ficou claro que pelo conjunto de características que possui o SRIS-HC é o sistema mais completo dentre as duas soluções apresentadas. No entanto, seu conjunto de características não é satisfatório no suporte e manipulação do conteúdo DICOM. O quadro abaixo apresenta um conjunto de características desejáveis para um ambiente virtual de diagnóstico a distância, e que não estão presentes nos sistemas analisados.

Quadro 2 – Características desejáveis.

Característica
Armazenamento de forma nativa do conteúdo DICOM no SGBD.
Recuperação de imagens DICOM através dos seus metadados.
Utilização da plataforma WEB, possibilitando o uso do sistema em qualquer plataforma.

3. FUNDAMENTAÇÃO CONCEITUAL E METODOLOGIA DO TRABALHO

Para elaboração da proposta de uma arquitetura de armazenamento e manipulação de imagens médicas, foi realizado primeiramente um estudo acerca das tecnologias envolvidas em sistemas similares de armazenamento e distribuição de imagens médicas. Os tópicos analisados foram:

- Imagens Digitais - um estudo acerca das características inerentes as imagens digitais foi realizado, bem como de seus formatos mais utilizados. Além do padrão DICOM, que visa regulamentar sua distribuição e armazenamento;
- Armazenamento de Imagens - realizou-se um estudo acerca das possibilidades de armazenamento de imagens em SGBD, verificando-se as vantagens e desvantagens do seu uso. Também foi realizado um estudo sobre o Oracle 11g (ORACLE 11g, 2010), que se mostrou ser o único sistema de banco de dados com suporte a imagens no formato DICOM; e
- Tecnologias Web - a tecnologia JSP (JSP, 2010) e Servlets (SERVET, 2010) que possui melhor integração com o Oracle 11g, foi analisada, assim como os aspectos relativos à sua arquitetura e vantagens no seu uso.

3.1 ESTUDO DE IMAGENS DIGITAIS

A imagem digital pode ser descrita como uma matriz cujos índices de linhas e colunas identificam um ponto na imagem, e o correspondente valor do elemento da matriz informa o nível de cinza naquele ponto. Os elementos da matriz são chamados de elementos de imagem (em inglês, *picture element*), conhecidos como *pixels* (GONZALEZ; WOODS apud HENRIQUE NETO; OLIVEIRA; VALERI, 2010).

A informação da cor de cada *pixel* da matriz é feita através de bits, e o número de bits é fator determinante para a quantidade de cores que poderão ser representadas na imagem. Estudos mostram que para algumas aplicações, como vídeo-endoscopia, médicos não vêem diferença entre uma imagem de 8, 16, ou 24 bits. Então, sempre que possível deve-se trabalhar com o menor número de bits possível, uma vez que a quantidade de bits é diretamente proporcional ao tamanho da imagem, demandando mais espaço de armazenamento e processamento dos equipamentos de visualização. Quando não for possível a utilização de imagens menores, o uso da compactação de imagens é indicado (FRANCESCHI, 2006).

3.1.1 COMPACTAÇÃO

Em uma imagem digital existem muitas informações repetidas. Utilizando essa característica, os algoritmos de compactação localizam áreas de redundância de informações, e as gravam em blocos e não ponto a ponto, a fim de reduzir o espaço ocupado no seu armazenamento. O grau de compressão de uma imagem é obtido pela divisão da resolução da imagem original, pela resolução da imagem resultante na compressão (FRANCESCHI, 2006).

3.1.2 FORMATOS DE IMAGENS DIGITAIS ESTÁTICAS

Imagens digitais estáticas são imagens de um único quadro, ou seja, não são vídeos, não têm movimento ao longo do tempo.

Devido à diversidade de fontes de imagens digitais como câmeras digitais e scanners, existe uma variedade de formatos para seus arquivos. Os mais utilizados são: RAW (RAW, 2010), PSD (PhotoShop Image) (PSD, 2010), BMP (Bitmap) (BMP, 2010), GIF (Graphics Interchange Format) (GIF, 2010), PNG (Portable Network

Graphics) (PNG, 2010), TIFF (Tagged Image File Format) (TIFF, 2010), JPEG (Joint Photographic Expert Group) (JPEG, 2010). Os arquivos podem ser constituídos em duas partes: o cabeçalho que contém informações sobre a imagem, e os dados sobre os *pixels* que constituem a imagem. Dentre os dados existentes no cabeçalho, destacam-se as informações sobre o tipo da imagem, seu esquema de cores, e dados espaciais, como largura e altura. Arquivos de imagens podem ser constituídos apenas de informação “pura” sobre a imagem, sem a presença de cabeçalho, como arquivos no formato RAW (FRANCESCHI, 2006).

3.1.3 FORMATOS DE IMAGENS DIGITAIS DINÂMICAS

As imagens dinâmicas são imagens formadas por mais de um quadro. Desta forma, são mais complexas que as estáticas por apresentarem sua representação nos eixos cartesianos x, y, e tempo. Os principais formatos para essas imagens são QuickTime Movie® (QUICKTIME, 2010), AVI (Audio Video Interleave) (AVI, 2010) e MPEG (Motion Picture Experts Group) (MPEG, 2010). Por possuírem maior tamanho que as imagens estáticas, a utilização de técnicas de compactação para imagens dinâmicas é fundamental. Os CODEC¹ para imagens dinâmicas fazem esse trabalho de compactação, que pode ser por meio de redução de informações em cada quadro do filme, reduzindo a quantidade de quadros por segundo, ou utilizando a técnica de redundância temporal (STEWART apud FRANCESCHI, 2006).

3.1.4 O PADRÃO DICOM

O DICOM - *Digital Imaging and Communication in Medicine* é um conjunto de normas para padronização da comunicação e do armazenamento de informações e imagens médicas. Um dos grandes diferenciais do DICOM é que a imagem e informações adicionais são encapsuladas em um único arquivo. A imagem é baseada no formato JPEG com ou sem compressão, e as informações são dados de pacientes, exames, entre outros. Para que os dados sejam armazenados de forma estruturada, são utilizadas “*tags*” que identificam e organizam as informações (HENRIQUE NETO; OLIVEIRA; VALERI, 2010).

¹ Dispositivos de hardware e software que codificam ou decodificam sinais.

3.1.4.1 Histórico

Na década de 1980, se iniciava uma crise de interoperabilidade entre os equipamentos de imagens médicas. A NEMA - *National Electrical Manufacturers Association* (NEMA, 2010) e o ACR - *American College of Radiology* (ACR, 2010), produtores de equipamentos de radiologia digital, formaram um comitê chamado ACR-NEMA, com o objetivo de desenvolver um padrão para imagens digitais e transmissão destas imagens entre os equipamentos em medicina. Este padrão recebeu o nome de DICOM - *Digital Imaging and Communication in Medicine* (DICOM, 2010).

As duas primeiras versões do DICOM eram bastante específicas nos detalhes de sua implementação. A versão 1.0 especificava até o tipo de conector físico que deveria ser utilizado nas ligações de um sistema digital. Devido a essas minúcias, a versão 1.0 não teve aplicabilidade, já que obrigava que os fabricantes alterassem suas linhas de produção para atender aos requisitos exigidos pelo padrão. A versão 2.0 também não obteve sucesso pelo mesmo motivo, no entanto, motivou aos grandes fabricantes a adaptarem seus produtos à interoperabilidade, que foi vista como um diferencial na venda dos equipamentos (FRANCESCHI, 2006).

Em 1993, foi apresentada a versão 3.0 do DICOM. Esta versão fornece informações às modalidades radiológicas de tomografia, ultra-sonografia e ressonância magnética. As modalidades não associadas à radiologia deveriam utilizar o padrão através de um módulo chamado captura externa (EC). Tal módulo permitia que imagens de qualquer formato fossem armazenadas como arquivo DICOM. A versão 3.0 trouxe como novidade o incentivo ao uso de redes, objetivando a criação de meios de comunicação entre os equipamentos e os hospitais (FREITAS, 2002; FRANCESCHI, 2006).

Com o passar dos anos, o DICOM foi se tornando o padrão em diversas outras modalidades de exames além da radiologia. O grande marco para tal evolução foi o anúncio formal da ACR-NEMA, que orientou todas as organizações que utilizavam imagens médicas a trabalhar na construção de extensões para o DICOM 3.0.

3.1.4.2 O DICOM 3.0

O DICOM pode ser considerado como um padrão de comunicação entre aplicações, dispositivos e sistemas heterogêneos, contendo informações detalhadas de todas as etapas que devem ser realizadas para a intercomunicação de equipamentos de aquisição de imagens médicas, estações de trabalho, unidades de arquivos, dentre outros equipamentos, de diversos fabricantes (HENRIQUE NETO; OLIVEIRA; VALERI, 2010) (CUNHA, 2010) (FREITAS, 2002).

Para facilitar o desenvolvimento e extensão do padrão, o DICOM 3.0 é um documento estruturado em diversas partes, e propicia adaptabilidade devido ao seu modelo de informação orientado para objeto. Além da definição de objetos para imagens o padrão também possui objetos para pacientes, relatórios e outros grupos de dados. As diferentes partes são independentes na sua atualização e edição, no entanto, são totalmente compatíveis entre si (CUNHA, 2010) (FRANCESCHI, 2006).

Existem 16 partes no DICOM 3.0, que representam diferentes pontos de vista do protocolo. Devido a relevância para este trabalho, a seguir serão descritas as partes 3, 6 e 10, das partes mostradas na Figura 5.

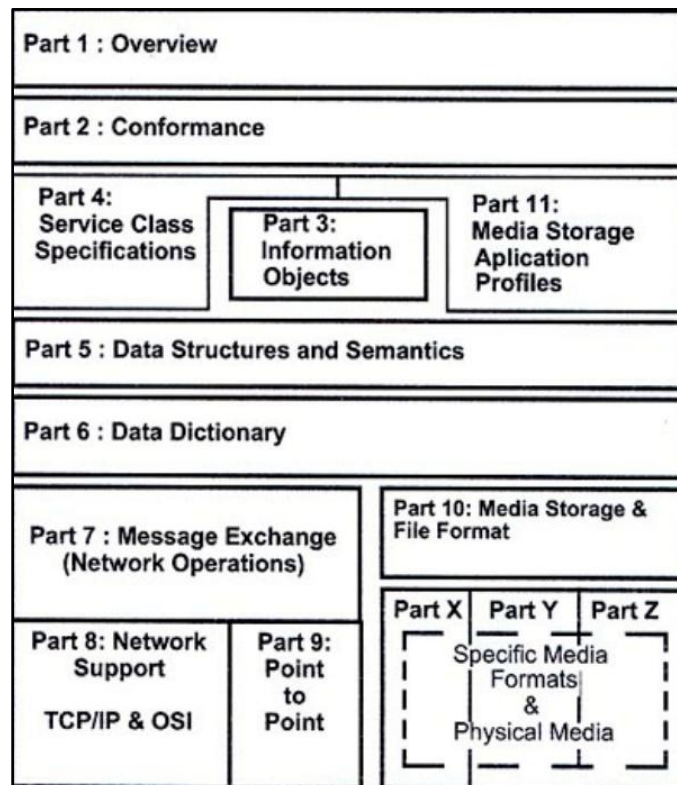


Figura 5 - Principais partes presentes na extensão do protocolo DICOM (HENRIQUE NETO; OLIVEIRA; VALERI, 2010).

3.1.4.2.1 DICOM Parte 3 - Definição de Objeto de Informação – IOD

O padrão DICOM trabalha com a definição de objeto de informação IOD – *Information Object Definition* (DICOM, 2010). O IOD é um modelo de dados orientado a objetos, que especifica objetos reais. Na realidade, um IOD não especifica diretamente um objeto real, mas uma classe de objetos reais que partilham das mesmas propriedades e atributos. Existem dois tipos de classes: normalizadas e compostas (FREITAS, 2002) (CUNHA, 2010).

➤ IOD NORMALIZADO

O IODs normalizados são aqueles que representam apenas uma única entidade do modelo DICOM, ou seja, são as classes que incluem apenas atributos inerentes a entidades do mundo real representadas. Como exemplo segue um detalhamento da classe Paciente IOD no Quadro 3 (CUNHA, 2010) (SILVA, Ronaldo, 2010).

➤ Paciente IOD

A definição do objeto de informação Paciente é a representação das informações de um paciente do mundo real que foi submetido a um exame. Segue abaixo alguns dos seus principais atributos.

Quadro 3 - Principais atributos do IOD Paciente (SILVA, Ronaldo, 2010).

Módulo	Atributo	TAG	Descrição do Atributo
Patient Identification	Patient's Name	(0010,0010)	Nome completo do paciente
Patient Identification	Patient's ID	(0010,0020)	Código do paciente emitido pelo hospital
Patient Identification	Patient's Birth Name	(0010,1005)	Nome de nascimento do paciente
Patient Demographic	Patient's Birth Date	(0010,0030)	Data de Nascimento do paciente
Patient Demographic	Patient's Sex	(0010,0040)	Sexo do paciente
Patient Medical	Medical Alerts	(0010,2000)	Condições que a equipe médica deve estar alerta

As “tags” em DICOM são compostas por dois números de 16bits cada, os números são chamados de “*group-member*” e “*element-member*” respectivamente. As “TAGS” são utilizadas para identificar os atributos de forma única. Na Parte 6 do padrão DICOM são listadas todas as “tags” existentes em um dado (CUNHA, 2010).

Os Módulos em DICOM são uma classificação dos atributos em grupos, de modo que um IOD é composto por diversos Módulos que contém atributos. O Quadro 4 mostra alguns Módulos que estão presentes no IOD Paciente. A coluna Referência é a ligação com outra tabela que possui todos os atributos do respectivo módulo.

Quadro 4 - Módulos presentes no IDO Paciente.

Módulo	Referência	Descrição
Patient Identification	C.2.2	Identifica o paciente
Patient Demographic	C.2.3	Descreve o paciente
Patient Medical	C.2.4	Informações médicas sobre o paciente

➤ IOD COMPOSTO

O IOD composto, como o próprio nome sugere, é composto de mais de uma Entidade do mundo real. A combinação dessas Entidades busca a construção de um objeto de maior complexidade de dados, como um exame ou procedimento. O Quadro 5 mostra alguns IOD compostos.

Quadro 5 - Exemplos de IODs Compostos

IOD	Descrição
CR	Radiografia Computadorizada
ES	Endoscopia
MR	Ressonância Magnética
SC	Filmes Escaneados
US	Ultrassom

Para ilustrar um IOD Composto, o Quadro 6 mostra as principais Entidades, Módulos e Atributos que especificam uma imagem que foi gerada por um equipamento de Ressonância Magnética (MR).

Quadro 6 Principais componentes do IOD MR (SILVA, Ronaldo, 2010).

Entidade	Módulo	Atributo	TAG	Descrição do Atributo
Patient	General Patient	Patient's Name	(0010,0010)	Nome completo do paciente
		Patient's ID	(0010,0020)	Código do paciente emitido pelo hospital
		Patient's Birth	(0010,0030)	Data de nascimento do paciente
Study	General Study	Study UID	(0020,000D)	Identificador único do estudo
		Study Date	(0008,0020)	Data do estudo
		Study Desc.	(0008,1030)	Descrição do estudo
Series	General Series	Modality	(0008,0006)	Modalidade (CT, MR, XR)
		Protocol name	(0018,1030)	Nome do protocolo utilizado
		Body Part	(0018,0015)	Descrição da parte do corpo examinada
Image	General Image	P. Orientation	(0020,0020)	Orientação do paciente
		Images Acqui.	(0010,1002)	Número de imagens na aquisição
	MR Image	Image Type	(0008,0008)	Tipo da imagem
		Echo Time	(0018,0081)	Parâmetro de aquisição de imagem

3.1.4.2.2 DICOM Parte 6 – Dicionário de Dados

É o registro de todos os elementos de dados disponíveis para representar informações e todos os identificadores únicos (UID). Para cada elemento são especificados os seguintes componentes:

- TAG única que contém o número do grupo e do elemento representado;
- Nome;

- Sua representação de valores – VR (String, inteiro, por exemplo); e
- Multiplicidade (quantos valores por atributo).

Para cada item unicamente identificado, especifica:

- Valor único, que é numérico e limitado a 64 caracteres;
- Nome;
- Tipo, Classe de objetos de informação; e
- Em que parte do padrão DICOM está definido.

➤ **IDENTIFICADORES ÚNICOS (UID)**

O UID é um número de identificação único utilizado nas mensagens DICOM, associado a uma organização e a uma função específica. Este número é composto por duas partes UID = <prefixo (org root) >. <suífixo>.

➤ **DICOM SOP CLASSES**

As capacidades de um conteúdo DICOM são expressas em pares de Classes e serviços. É uma forma para definir que uma determinada capacidade, como armazenamento de imagens de ultra-sonografia, está relacionada com o objeto (Imagem US) e com o serviço relacionado (armazenamento). O objetivo é que todas as classes DICOM SOP sejam identificadas de forma única (UID) e pelo nome.

3.1.4.2.3 DICOM Parte 10 – DICOM Files

Também chamada de Off-line Media, a Parte 10 do padrão DICOM descreve como armazenar informações de imagens médicas em uma mídia removível qualquer. No armazenamento na mídia é necessária a utilização de um índice, que fornece informações sobre todos os arquivos armazenados naquela mídia. Devido ao uso do índice não se faz necessário o uso de nomes de arquivos grandes, podendo apenas conter 8 caracteres. Os arquivos DICOM têm normalmente a extensão *.dcm*.

3.2 ESTUDO DO ARMAZENAMENTO DE IMAGENS EM UM SGBD

Por muitas décadas, o sistema de saúde operou utilizando papel e filmes. Os papéis utilizados para registro dos pacientes, e filmes utilizados como registro de exames. Na última década, a tecnologia da informação foi introduzida no sistema de saúde, e foi vastamente adotada pelos hospitais e centros médicos. O aumento do uso de equipamentos digitais nos hospitais e o aumento da qualidade das imagens geradas pelos equipamentos médicos geraram um problema de armazenamento (ANNAMALAI, 2010).

A solução inicial e mais simples é o armazenamento em sistemas de arquivos. Essa opção é bastante suscetível a falhas, pois não existe controle de acesso e dos processos que têm permissão para manipular os dados. A outra solução é o armazenamento de imagens em um SGBD – Sistema de Gerenciamento de Banco de Dados, que já tem seu uso consolidado no armazenamento de dados não complexos, os dados textuais.

O SGBD é o responsável pela manutenção e gerenciamento de uma base de dados. Ele é o responsável pela integridade, pelo controle de acesso e a manipulação dos dados. No capítulo anterior, ficou clara a existência de um padrão de imagens médicas no mercado, o padrão DICOM. O padrão que tem como objetivo aperfeiçoar a integração de sistemas heterogêneos, já está bastante consolidado nas aplicações médicas, mas, é pouco difundido nas soluções de banco de dados.

Atualmente, no mercado existem diversas soluções na área de gerenciamento de banco de dados, no entanto, o suporte a imagens médicas no formato DICOM, de forma nativa, é raro de ser encontrado, limitando-se apenas ao armazenamento no formato blob (Binary Large Objects). Neste capítulo será apresentada uma discussão sobre as vantagens e desvantagens do armazenamento de imagens médicas em um SGBD, e sua relevância na arquitetura de um ambiente virtual de diagnóstico a distância.

3.2.1 VANTAGENS

Historicamente, os SGBD eram conhecidos por terem problemas de desempenho na recuperação de imagens. No entanto, nos últimos cinco anos, as mudanças na tecnologia dos bancos de dados e a melhoria na capacidade e velocidade dos discos rígidos, tornaram seu uso bastante atrativo devido suas diversas características vantajosas no gerenciamento de dados (KRATOCHVIL, 2010). Algumas destas características serão discutidas nesta seção.

➤ Gerenciamento

Imagens armazenadas no banco podem ser ligadas diretamente com metadados², que podem ser modificados em uma única transação. Também em uma única transação uma imagem pode ser manipulada e um thumbnail³ desta imagem pode ser gerado. Informações relacionadas são mantidas em sincronia com as imagens no banco de dados, enquanto que se as imagens estiverem armazenadas em um sistema de arquivos comum, então será possível que um processo apague ou modifique esta imagem, perdendo o relacionamento com os dados armazenados no banco de dados (KRATOCHVIL, 2010).

➤ Segurança

O armazenamento em sistema de arquivos comum não permite um controle preciso das imagens. Isto significa que normalmente não é possível restringir o acesso de usuários específicos a imagens, se um usuário tem permissão para acessar um diretório, ele terá acesso a todas as imagens do diretório, não sendo possível a restrição de visualização de algumas delas.

O uso de um SGBD para armazenar as imagens torna possível o controle preciso de acesso, tornando possível o gerenciamento de acesso aos arquivos de forma individual, ou seja, cada usuário terá suas permissões de acesso estabelecidas pelo SGDB. Além dessa importante característica, o uso de um SGDB

² São dados sobre outros dados, ou seja, informações adicionais de características de dados.

³ São versões reduzidas de imagens, usadas para tornar mais fácil o processo de busca e reconhecimento.

traz como benefício à possibilidade de auditoria de quem acessou ou modificou as imagens e o estabelecimento de um tempo limite para cada operação de acesso às imagens, com o objetivo de não prender recursos do servidor em uma única operação por um longo período (KRATOCHVIL, 2010).

➤ **Backup e Restauração**

Quando as imagens são armazenadas no sistema de arquivos, e os dados textuais armazenados no SGBD, o processo de backup é feito em duas etapas e não garante a sincronia dos dados. Com o armazenamento das imagens no SGBD o processo de *backup* é simplificado, e a sincronia entre os dados é garantida, pois não há riscos de uma transação completada no SGBD não se refletir nas imagens. O processo de restauração em caso de falhas também é mais seguro: apenas uma restauração é necessária e o SGBD voltará ao funcionamento da última transação realizada antes do *backup*.

➤ **Replicação**

Atualmente, é comum o armazenamento de dados de forma replicada. O objetivo é garantir o acesso de pontos externos ao servidor de dados, sem a necessidade de uma conexão de internet permanente e a garantia extra de *backup* dos dados. Para isso, se faz necessário a replicação dos dados do servidor central em outros servidores externos. Esta replicação é feita de forma automatizada quando os dados estão armazenados em um SGBD, e se torna inviável quando se utiliza um sistema de arquivos para armazenamento desses dados.

3.2.2 DESVANTAGENS

Qualquer solução tem prós e contras, não poderia ser diferente com o SGBD. Segue abaixo algumas desvantagens no uso desta solução, e cabe ao arquiteto do projeto pesar e escolher de forma coerente qual a melhor solução para as suas necessidades.

➤ **Tamanho do Banco de Dados**

O armazenamento de imagens em um SGBD ocupa mais espaço que o seu armazenamento em um sistema de arquivos. Esse fato é devido ao formato de armazenamento utilizado em um SGBD, que contém outras informações de gerenciamento que não são utilizadas em um sistema de arquivos simples. Esta característica tem sido amenizada devido à redução dos custos de discos rígidos, e outras formas de armazenamento.

➤ **Complexidade**

Armazenar e recuperar imagens em um SGBD exige um trabalho a mais por parte dos desenvolvedores, este trabalho é devido à necessidade da construção de uma rotina de inserção e recuperação desses dados no banco. Tal dificuldade tem regredido devido à facilidade das novas tecnologias de banco de dados na realização dessas operações, e na documentação que pode ser encontrada de forma simplificada em livros e comunidades na internet.

3.2.3 CONSIDERAÇÕES

As características apresentadas em um ambiente virtual de diagnóstico a distância como a necessidade de autenticação, de restrição de acesso, de prevenção contra perda de informações e transmissão sem perda de dados, se faz necessário o uso de um SGBD na arquitetura desse sistema. As desvantagens apresentadas no uso de um SGBD não são relevantes comparadas às suas características essenciais ao funcionamento correto de um ambiente virtual.

3.3 ESTUDO DO ORACLE 11G MULTIMÍDIA DICOM

Os principais SGBD do mercado, como o PostGres (POSTGRES, 2010), MySQL (MYSQL, 2010), SQL-Server (SQL-SERVER, 2010) e ORACLE (ORACLE 11g, 2010), foram analisados, e apenas o ORACLE 11g apresentou suporte ao tipo de dado DICOM. Com o Oracle 11g, o formato DICOM tem total suporte. Aplicações podem utilizar PL/SQL (PL/SQL, 2010) e Multimídia Oracle Java API⁴ (ORDDICOM, 2010) para inserir, manipular e recuperar conteúdo DICOM. Devido a sua exclusividade, o ORACLE 11g será alvo de estudo neste capítulo, no qual serão abordadas suas características e funcionalidades no armazenamento de imagens no formato DICOM.

➤ **Flexibilidade**

Na utilização do Oracle para armazenar imagens DICOM, as imagens podem ser manipuladas como qualquer dado relacional comum. Conjuntos de imagens podem ser atualizados, apagados ou copiados utilizando comandos SQL⁵ (SQL, 2010). O conteúdo é armazenado em colunas nas tabelas, e também podem ser recuperados através de junções relacionais (ANNAMALAI, 2010).

➤ **Segurança**

Em um ambiente virtual de diagnóstico a distância, é essencial o controle de acesso aos dados médicos. Quando um arquivo DICOM é armazenado no sistema de banco de dados Oracle, este garante que acessos não autorizados aos dados não ser bloqueados. O Oracle possui funcionalidades que permitem aos administradores controlarem quais linhas e colunas das tabelas podem ser visualizadas por um usuário determinado (ANNAMALAI, 2010).

⁴ Conjunto de classes e interfaces para facilitar o desenvolvimento de aplicações.

⁵ Linguagem de Consulta Estruturada para pesquisas em bancos de dados.

➤ **Auditoria**

O Oracle permite que todas as operações sobre o conteúdo do banco de dados sejam gravadas. Essa informação pode ser acessada por administradores, que têm a possibilidade de efetuar a auditoria completa no banco de dados, com informações de data do evento, usuário que realizou e qual dado sofreu modificação.

➤ **Técnicas de redução de espaço**

No sistema de banco de dados Oracle existem algumas técnicas de redução de espaço ocupado. Essas técnicas podem melhorar o desempenho do sistema, bem como aumentar a velocidade do *backup* e restauração. As técnicas mais comuns são de compressão do conteúdo DICOM, a remoção de espaços não utilizados no banco de dados e redução dos arquivos de dados (ANNAMALAI, 2010).

➤ **Criptografia**

Para a proteção das imagens e dados, o Oracle utiliza poderosas chaves para criptografia. Essa característica garante privacidade e pode até proteger o conteúdo dos arquivos dos administradores do sistema. Os backups também podem ser criptografados, garantindo total privacidade às informações médicas dos pacientes (ANNAMALAI, 2010).

3.3.1 REPOSITÓRIO DE MODELOS DE DADOS DICOM

Todas as funcionalidades do Oracle Multimídia DICOM são controladas e gerenciadas por um conjunto de documentos XML (Extensible Markup Language) (XML, 2010), que estão armazenados em um repositório de modelo de dados DICOM (DICOM Data Model Repository). As funcionalidades padrão do repositório de modelo de dados são iniciadas automaticamente na instalação do banco de dados, não precisando de nenhuma configuração por parte dos usuários. Só sendo necessária essa intervenção apenas quando queiram alterar algum comportamento padrão. Alguns desses documentos XML são: O dicionário de dados de elementos

DICOM, documentos que controlam o mapeamento de atributos DICOM em XML, um documento que controla as preferências de funcionamento das funcionalidades do banco de dados (ANNAMALAI, 2010).

3.3.2 ARQUITETURA

A arquitetura do Oracle Multimídia DICOM é dividida em duas perspectivas: a camada do Banco de Dados e a camada cliente. A camada do Banco de dados contém diversos componentes que realizam o armazenamento, recuperação e processamento do conteúdo DICOM, e a camada cliente é formada pelas aplicações desenvolvidas na linguagem Java ou em outras linguagens de programação (PELSKI, 2010).

A camada de banco de dados armazena o conteúdo DICOM em tabelas, como ilustrado na Figura 6. O conteúdo DICOM é armazenado em uma coluna da tabela no formato ORDDicom. Na tabela, outra coluna armazena um thumbnail do arquivo DICOM em JPEG. Esta coluna é importante no processo de visualização e das imagens reduzidas nas aplicações clientes. Ainda na tabela, uma outra coluna armazena os documentos XML de metadados associados a cada imagem (PELSKI, 2010).

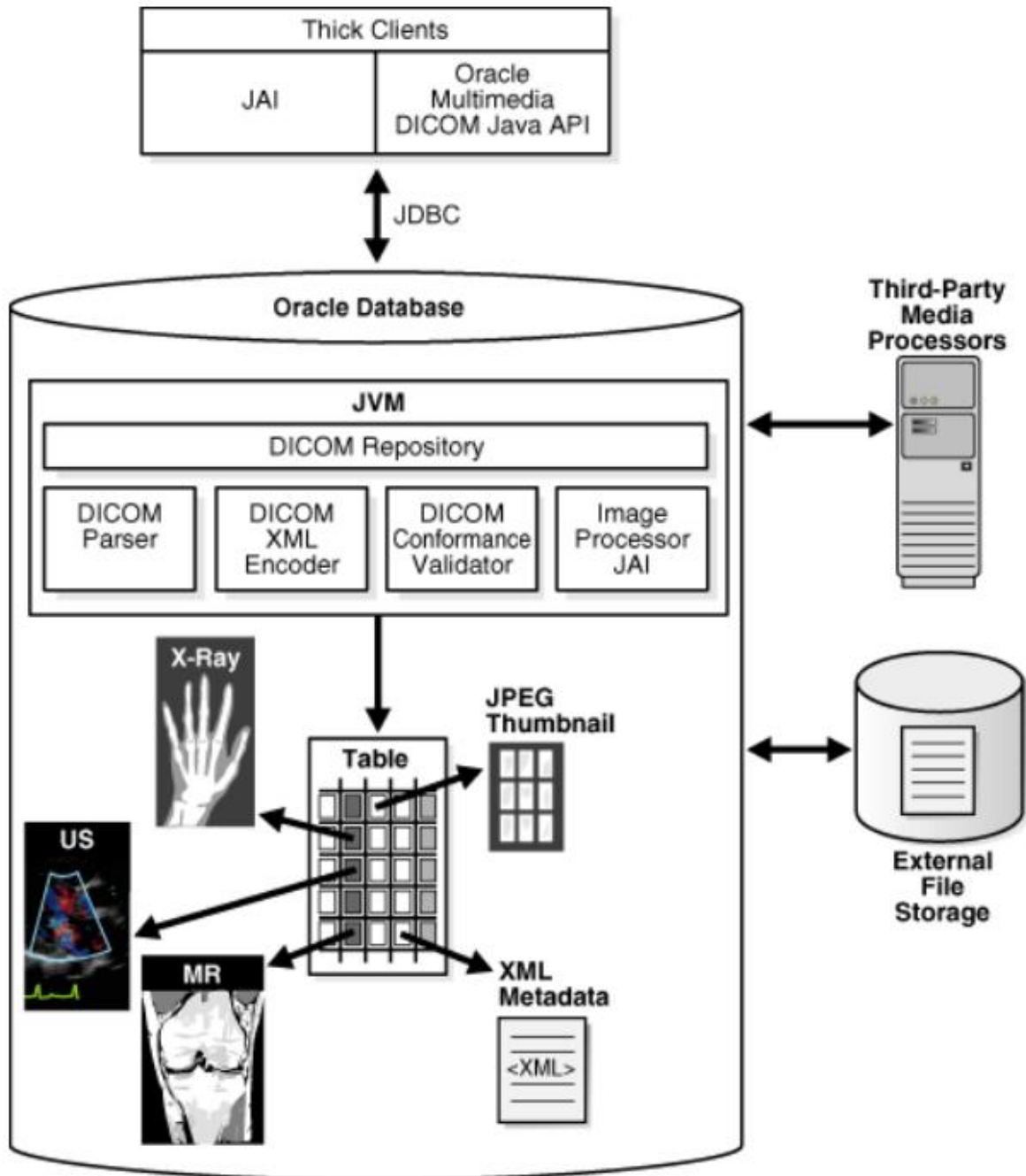


Figura 6 - Arquitetura do Oracle Multimídia DICOM (PELSKI, 2010).

Com a utilização de métodos do Oracle multimídia DICOM é possível a importação e exportação de conteúdo DICOM entre o banco de dados e um repositório de arquivos externo (External File Storage). Através de chamadas do JDBC⁶ (Java Database Connectivity) (JDBC, 2010), aplicações clientes podem acessar o conteúdo armazenado no banco de dados e manipular os objetos DICOM,

⁶ API escrita em JAVA para fazer o envio de instruções SQL ao banco de dados.

com a utilização da Oracle Multimídia DICOM Java API, que utiliza o tipo `OrdDicom` para recuperar os arquivos `ORDDicom` do banco de dados através do JDBC (PELSKI, 2010).

Em uma máquina virtual Java (JVM) (JVM, 2010) existe um repositório de modelo de dados, o analisador DICOM Parser, o DICOM XML Encoder, o DICOM Conformance Validator e o Image Processor. O DICOM parser tem a função de extrair os metadados do arquivo DICOM, de forma que esses dados fiquem disponíveis para futuras consultas. O XML Encoder mapeia os dados extraídos pelo DICOM Parser em um documento XML, de acordo com as regras de mapeamentos estabelecidas no repositório de modelo de dados. O DICOM Conformance Validator verifica se o conteúdo DICOM se enquadra nas regras estabelecidas no repositório de modelo de dados, quanto sua consistência semântica e sintática. O Image processor é o responsável por operações de processamento de imagem no banco de dados. Operações como a geração de *thumbnails* de tamanhos personalizados para imagens DICOM, e a conversão entre DICOM e outros formatos de imagens com suporte no sistema (PELSKI, 2010).

Outro ponto importante na arquitetura é que através da OCI (Oracle Call Interface) outros sistemas podem se comunicar com o sistema de banco de dados Oracle, abrindo portas para a utilização de outras aplicações (Third-Party Media Processors) em diferentes linguagens de programação (PELSKI, 2010).

3.3.3 ARMAZENANDO O CONTEÚDO DICOM

Para possibilitar o armazenamento de conteúdo DICOM, o Oracle 11g possui um novo tipo de objeto, o `ORDDicom`, que oferece suporte ao conteúdo DICOM de forma nativa. Esse objeto armazena o conteúdo DICOM, extrai seus metadados, e disponibiliza os métodos necessários para manipular seu conteúdo. O tipo de dado `ORDDicom` pode ser utilizado normalmente em uma coluna de uma tabela, similarmente a outros dados não complexos.

A Figura 7 contém uma representação do conteúdo de um objeto `ORDDicom`. A parte 1 contém um conjunto de atributos e métodos que podem ser utilizados a partir de uma instância do objeto. A parte 2 contém os atributos DICOM extraídos e armazenados em um documento XML de metadados. A parte 3 representa o arquivo

DICOM original, que fica armazenado como um BLOB no banco de dados. A parte 4 contém atributos que são usualmente acessados, como os atributos da SOP Class UID. Finalmente, a parte 5 representa atributos internos do banco de dados (PELSKI, 2010).

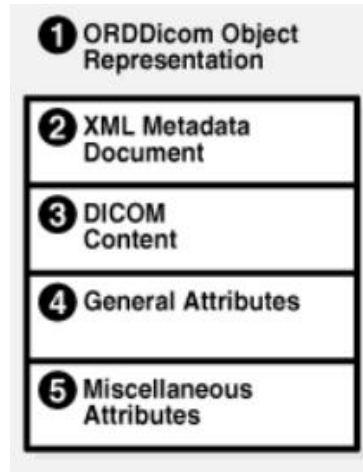


Figura 7 - Objeto ORDDicom (PELSKI, 2010).

A Figura 8 mostra o ambiente de um banco de dados, que contém uma tabela com conteúdo DICOM. O item 1 na figura representa o banco de dados Oracle, o item 2 representa uma tabela do banco, o item 3 representa a coluna que identifica uma imagem na tabela, o item 4 representa o conteúdo DICOM na tabela. O item número 5 representa que a coluna que armazena o conteúdo DICOM é do tipo de dado ORDDicom.

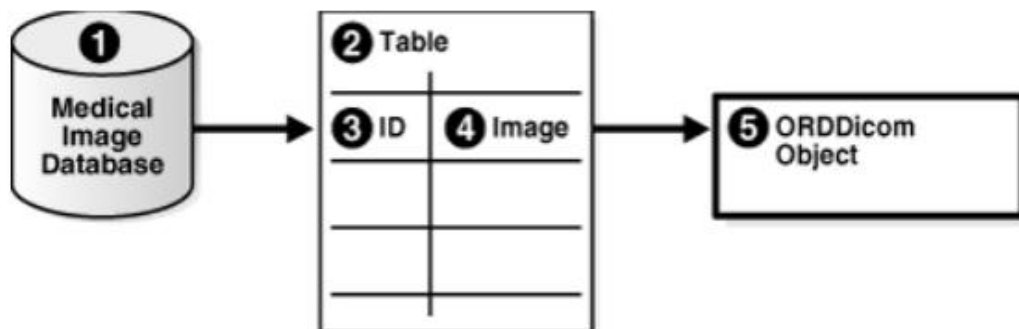


Figura 8 - Imagem DICOM armazenada no banco de dados (PELSKI, 2010).

3.3.4 MÉTODOS DO OBJETO ORDDicom

Neste tópico serão abordados os principais métodos utilizados pelo Oracle Multimídia. Métodos são procedimentos ou funções que são utilizados para manipular o conteúdo DICOM armazenado no banco de dados. São eles:

- *Import()* – Método que importa um arquivo DICOM, que está armazenado em um diretório do sistema de arquivos, para uma tabela do banco de dados;
- *Export()* – Método que produz uma cópia do conteúdo DICOM, que está no banco e dados, e gera um arquivo DICOM no sistema de arquivos;
- *setProperties()* – O DICOM Parser extrai os metadados do conteúdo DICOM e torna-os disponíveis em um documento XML;
- *extractMetadata()* – Esse método extrai um subconjunto dos metadados do conteúdo DICOM. Este subconjunto é selecionado pelo usuário, que terá os metadados disponíveis em um documento XML;
- *processCopy()* – Esse método transforma o conteúdo DICOM em uma imagem JPEG ou outros formatos de imagens, com suporte no banco de dados. Ele possibilita a geração de *thumbnails* que são de extrema importância na construção de aplicações;
- *makeAnonymous()* – Método que cria um novo objeto ORDDicom baseado em um já existente, mas com as informações que identificam o paciente modificadas ou removidas; e
- *extractValue()* – Método que extrai um atributo, especificado pelo usuário, dos metadados do conteúdo DICOM selecionado.

3.3.5 RECUPERAÇÃO DE IMAGEM POR CONTEÚDO

A Recuperação de imagem por conteúdo é uma funcionalidade do Oracle Multimídia, não sendo específica ao módulo Multimídia Dicom. No entanto, esta capacidade é de bastante utilidade nas pesquisas por exames similares. Como apresentado na arquitetura do Oracle multimídia DICOM, existe uma coluna na tabela de armazenamento do conteúdo DICOM, responsável por armazenar o *thumbnail* gerado a partir do conteúdo do arquivo DICOM. Através da extração de características do *thumbnail*, é possível se recuperar imagens por similaridade com alguma outra.

3.3.5.1 A classe *OrdImageSignature*

No Oracle Multimídia, existe o tipo de dado *OrdImageSignature*, que é uma classe que armazena a assinatura de uma imagem do tipo *OrdImage*. Através dos métodos disponíveis nessa classe é possível se fazer comparações entre as assinaturas de outras imagens, tornando possível a comparação de conteúdo entre elas. Os principais métodos são:

- *generateSignature()* - Gera uma assinatura para a imagem especificada (ORACLE, 2010); e
- *isSimilar()* - Método que recebe como parâmetros diversos atributos de comparação, e um limite de aceitação. Este método retorna se a comparação entre duas assinaturas de imagens está dentro do limite estabelecido, de acordo com os parâmetros passados (ORACLE, 2010).

3.3.6 ORACLE MULTIMÍDIA DICOM JAVA API

A API Java DICOM do Oracle Multimídia possibilita o desenvolvimento de aplicações com a manipulação direta do conteúdo DICOM. A API conta apenas com uma classe, a *OrdDicom*, que contém diversos métodos de manipulação do conteúdo DICOM. Alguns dos principais métodos seguem abaixo:

- *setPropreties()* - Método utilizado para extrair os metadados do conteúdo DICOM em um arquivo XML, e carregar os atributos do objeto *OrdDicom* com os dados extraídos;
- *getAttributeByTag()* - Retorna um atributo DICOM de um objeto *OrdDicom*, passando como parâmetro a “*tag*” do atributo desejado. O método *setPropreties()* deve ser utilizado antes deste; e
- *getAttributeByName()* - Retorna um atributo DICOM de um objeto *OrdDicom*, passando como parâmetro o nome do atributo desejado. O método *setPropreties()* deve ser utilizado antes deste.

3.4 ESTUDO DA TECNOLOGIA WEB

A internet tornou-se um meio essencial na troca de informações em todas as áreas de conhecimento. Em uma aplicação de ambiente virtual de diagnóstico a distância não poderia ser diferente, a internet é uma ferramenta essencial para sua viabilidade. No Capítulo 4, foi estabelecido o uso do Oracle 11g Multimedia Dicom como melhor opção de banco de dados em um ambiente virtual de diagnóstico a distância. O Oracle 11g possui uma integração bastante simplificada com a linguagem Java, através de JDBC. Outro ponto importante é a existência de uma API para manipulação de objetos ORDDIcom, a Oracle Multimídia DICOM Java API, que também é escrita em Java. Devido aos fatores apresentados, a tecnologia Web que melhor se adéqua ao desenvolvimento de um ambiente virtual de diagnóstico a distância, com utilização do Oracle 11g, é a JSP (Java Server Pages) (JSP, 2010) em conjunto com Servlets (SERVLETS, 2010).

Nesta seção serão abordados aspectos da tecnologia JSP e Servlets, como suas características e arquitetura.

3.4.1 SERVLETS

Servlets são classes Java, desenvolvidas com uma estrutura bem definida, e que rodam em um Servlet Container. Um Servlet Container funciona como um servidor web, com a função de receber requisições, processá-las e retornar uma resposta (BARALE, 2007) (TEMPLE, 2004).

3.4.1.1 Arquitetura

Servlets recebem requisições de usuários, através de browsers, e retornam repostas após efetuar um processamento. Devido a essa característica, Servlets são conhecidas por terem um modelo de *request/response*. Quando um usuário envia uma requisição para uma determinada servlet, um objeto da classe correspondente a Servlet é instanciado no Servlet Container, que gera uma resposta e a envia ao usuário através do browser. Um Servlet é instanciado apenas uma vez no servidor, não sendo possível a presença de dois ou mais objetos de uma mesma classe, ou a

presença de mais de um processo por servlet para se gerenciar (CAELUM, 2010) (BARALE, 2007). A arquitetura de servlets é mostrada na Figura 9.

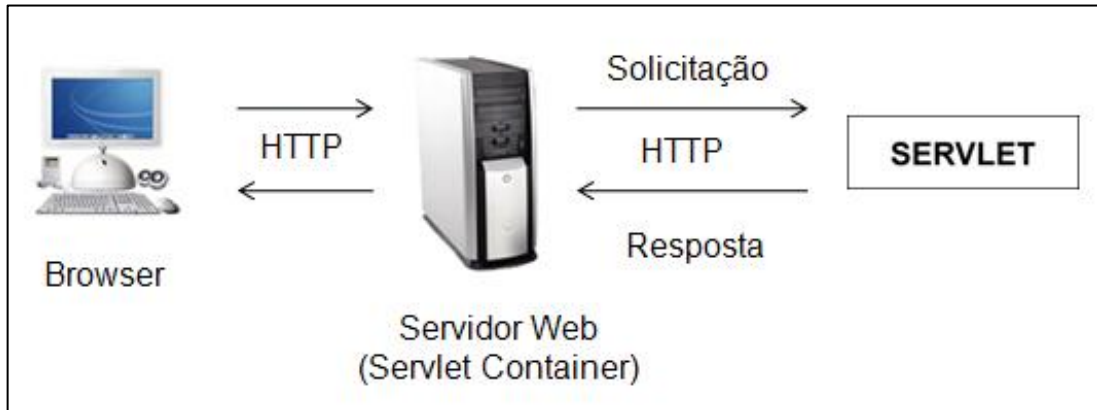


Figura 9 - Arquitetura Servlet

3.4.1.2 Protocolo HTTP

O protocolo HTTP é amplamente utilizado na internet, quando uma página Web é acessada diversas requisições HTTP são enviadas para o servidor, que hospeda aquela página, à medida que o servidor responde essas requisições o browser do usuário monta o conteúdo da página e o exibe na tela.

Existem alguns métodos HTTP que podem ser especificados no envio das requisições. Os principais deles são: o método *get*, utilizado para requisitar alguma informação ao servidor; e o método *post*, utilizado para enviar informações ao servidor.

3.4.1.3 Classe HTTP Servlet

A biblioteca Servlet contém basicamente duas classes, a `GenericServlet` e a `HttpServlet`. A Classe `HttpServlet`, alvo de estudo nesse tópico, estende a classe `GenericServlet` e contém mais dois outros objetos, a `HttpServletRequest` e `HttpServletResponse`. A classe `GenericServlet` atende a quaisquer requisições, enquanto que a `HttpServlet` apenas a requisições HTTP (BARALE, 2007).

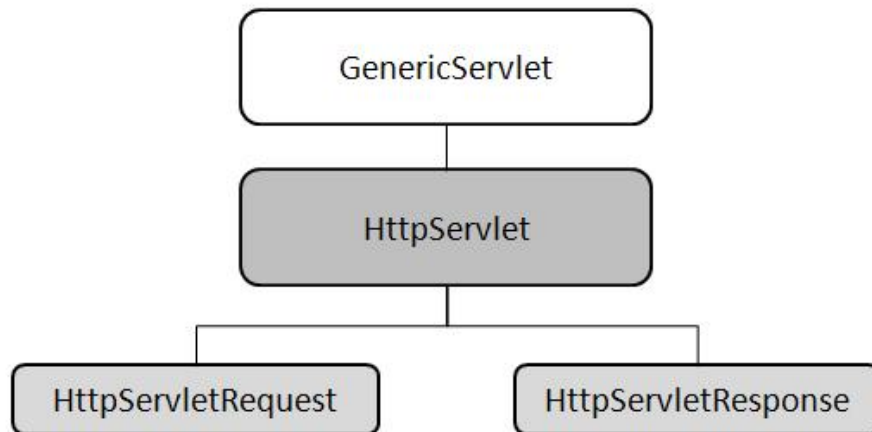


Figura 10 - Hierarquia Servlet.

A classe HttpServlet contém seis métodos, que são chamados conforme as requisições do HTTP. A chamada dos métodos é feita automaticamente, de acordo com o tipo de requisição passado no formulário HTML do browser. Os métodos principais são:

- *DoPost()*: envia dados ao servidor, uma única vez.
- *DoGet()*: envia dados ao servidor, repetidas vezes.
- *DoPut()*: envia um arquivo ao servidor.
- *DoDelete()*: remove um documento ou uma página do servidor.
- *Service()*: chamado quando não é especificado nenhum outro método.

A figura 7 mostra um exemplo de um formulário HTML que utiliza o método post. Este formulário irá enviar uma requisição HTTP para a Servlet AdicionaContato, que automaticamente chama o método doPost().

```

<form action="adicionaContato" method="post">
  Nome: <input type="text" name="nome" /><br />
  E-mail: <input type="text" name="email" /><br />
  Endereço: <input type="text" name="endereco" /><br />
  DataNascimento: <input type="text" name="dataNascimento" /><br />
  <input type="submit" value="Gravar" />
</form>
  
```

Figura 11 - Exemplo método post.

Caso na requisição não seja informado nenhum método, a servlet irá automaticamente executar o método `service()`, que é chamado na execução da servlet.

➤ HTTP Servlet Request

O objeto `HttpServletRequest` é o responsável por recuperar as informações passadas nas requisições enviadas para o servidor. Com ele é possível se recuperar os parâmetros da requisição, o cabeçalho e outras informações. Os principais métodos desse objeto são:

- `getHeaderNames()`: recupera os nomes do cabeçalho.
- `getHeader()`: recupera os valores do cabeçalho.
- `getParameter ()`: recupera o valor de um parâmetro.

➤ HTTP Servlet Response

O objeto `HttpServletResponse` é o responsável por enviar as respostas ao cliente, através do método `setContentType()` é informado o tipo de retorno, e através do método `getReader()` o conteúdo é enviado.

```
1 protected void service(HttpServletRequest request,
2     HttpServletResponse response) throws ServletException, IOException {
3     PrintWriter out = response.getWriter();
4
5     // escreve o texto
6     out.println("<html>");
7     out.println("<body>");
8     out.println("Oi mundo!");
9     out.println("</body>");
10    out.println("</html>");
11 }
```

Figura 12 - Exemplo do método `getReader()`.

3.4.1.4 Ciclo de uma Servlet

O ciclo de vida de toda Servlet é composto por três fases: inicialização, atendimento a requisições (request/response) e finalização. Os métodos utilizados na execução de cada fase são: *init()*, *service()* e *destroy()* respectivamente.

O processo de inicialização ocorre quando o ServletContainer carrega o Servlet. Existem duas formas de uma servlet ser inicializada, a primeira é através de uma configuração que automaticamente inicializa a servlet no momento em que o próprio servidor é iniciado, a segunda ocorre quando chega uma primeira requisição mapeada para a servlet.

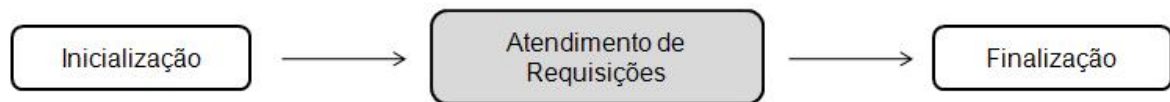


Figura 13 - Ciclo de uma Servlet.

No momento em que a servlet está carregada, ela já está apta a receber e responder requisições, e enquanto o servidor estiver carregado a servlet continuará nesta fase. Uma vantagem nessa característica é que, como a Servlet permanece carregada no ServletContainer, os seus dados de variáveis e atributos persistem ao longo de diversas requisições recebidas. Desta forma, é possível se manter uma única conexão ativa com o Banco de dados, sem a necessidade de abrir uma nova conexão a cada nova requisição enviada a servlet, dentre outras possibilidades.

O Servlet é finalizado quando a aplicação que o executa é inativada pelo ServletContainer, ou quando o próprio servidor é desativado.

3.4.1.5 Gerenciamento de Sessão

Atualmente, as aplicações utilizam um sistema de identificação do usuário como o sistema de *login*. O protocolo HTTP é um protocolo *stateless*, isto significa que nenhuma informação em relação a requisições é persistida no servidor. Para possibilitar o armazenamento de informações, a tecnologia Servlet usa o conceito de sessão.

Uma sessão é criada de forma única para um usuário, e sempre que esse usuário fizer uma requisição, ela será feita dentro daquela sessão. Desta forma se torna possível o controle de restrições para cada usuário, bem como o controle de páginas disponíveis para visualização do mesmo.

Na criação de uma sessão o objeto *Session* é instanciado. Quando criado o objeto envia ao browser do cliente e ao servidor um identificador da Sessão iniciada, que só é desativado quando a sessão é finalizada pelo browser ou aplicação. O objeto *Session* utiliza alguns métodos para guardar informações do usuário. Os principais métodos são: *setAttribute()*, utilizado para armazenar um dado na sessão, para isso ele recebe como entrada o nome do atributo e o seu valor; e o *getAttribute()*, utilizado para recuperar uma informação da sessão, para isso ele recebe como entrada o nome do atributo.

3.4.1.6 Filtros

A API de Servlets provê um mecanismo de filtros bastante útil. Filtros são classes que interceptam uma requisição, e permitem que código seja executado antes do processamento da requisição. Os filtros implementam a interface *javax.servlet.Filter*, e precisam conter três métodos: *init()*, *destroy()* e *doFilter()*. O método mais importante é o *doFilter()*, que fará todo o processamento. Abaixo segue um exemplo do método *doFilter()*, que verifica se o usuário já está autenticado no sistema, caso não esteja ele é redirecionado para a página de login. A figura 14 apresenta um exemplo de Filtro.

```
public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain)
    throws IOException, ServletException {

    HttpSession session = ((HttpServletRequest)req).getSession();
    String login = (String)session.getAttribute("LOGIN");

    if(login == null || login == "null"){

        session.setAttribute("MSGLOGIN", "Você não está logado no sistema!");
        ((HttpServletRequest)res).sendRedirect("../loginForm.jsp");

    }else{
        chain.doFilter(req, res);
    }
}
```

Figura 14 - Exemplo do método *doFilter()*.

3.4.2 JAVA SERVER PAGES (JSP)

Uma JSP é uma página HTML que pode conter código Java. Quando isso acontece, o código Java é chamado de *scriptlet*. A JSP foi criada com a intenção de reduzir a dificuldade na apresentação de conteúdo por parte das servlets, pois o processo de envio de resposta é bastante limitado quanto ao layout da página. Com a utilização da JSP é possível trabalhar em conjunto com um designer e ter a separação entre as servlets e as páginas de exibição ao usuário (TEMPLE, 2004) (BARALE, 2007).

As JSP contêm elementos Java, que podem realizar um processamento por si, como podem também recuperar dados de uma servlet previamente executada e apresentar esse conteúdo ao cliente. Uma característica importante das JSP é o fato do código HTML ser gerado dinamicamente no momento da execução, desta forma alterações no layout da página são possíveis sem a interrupção da aplicação.

A JSP funciona da seguinte forma: quando um usuário faz uma requisição de uma página JSP ao servidor, o mesmo verifica se a JSP está compilada, caso já esteja, a resposta HTML é gerada para o browser, do contrário, a JSP é transformada em uma Servlet, que é compilada no servidor. Devido ao processo de compilação, há uma lentidão no primeiro acesso a uma página JSP. Após a compilação, os acessos seguintes a uma página JSP são muito mais rápidos (BARALE, 2007).

3.4.3 VANTAGENS NA UTILIZAÇÃO DE JSP E SERVLETS

O uso da tecnologia de Servlets e de páginas JSP oferece vantagens em relação ao uso de outras tecnologias existentes no mercado (como PHP e ASP). Essas vantagens são herdadas da própria linguagem Java. Abaixo estão listadas algumas das principais vantagens no uso de Servlets e páginas JSP.

- Facilidade no desenvolvimento de aplicações: como a tecnologia é baseada na linguagem Java, toda a programação é orientada a objetos, facilitando o desenvolvimento de sistemas complexos. Além das características inerentes a linguagem Java, como o gerenciamento automático de memória, que oferece uma grande ajuda aos programadores, que não tem que se

preocupar com a reserva e liberação de memória durante o processo de execução (TEMPLE, 2004).

- Documentação Acessível: a linguagem Java é bastante difundida entre os desenvolvedores, contando com uma enorme comunidade e ampla documentação. Além de diversas bibliotecas e códigos prontos, que aceleram o processo de desenvolvimento de aplicações (TEMPLE, 2004).
- Portabilidade: Como qualquer aplicação Java, as Servlets e páginas JSP são multiplataforma, ou seja, podem ser implantadas em diversas plataformas, como Windows, Unix, e Mac, sem a necessidade de qualquer modificação na codificação da aplicação (TEMPLE, 2004).

3.5 METODOLOGIA EMPREGADA

O estudo das características e arquiteturas das tecnologias de Imagens Digitais, sistemas de banco de dados e páginas WEB, possibilitou a análise dos pontos fortes e fracos de cada tecnologia. Após a análise destes pontos, levando em consideração as necessidades de um sistema de armazenamento de imagens médicas e a capacidade de integração entre as tecnologias, foi desenvolvida a proposta da arquitetura de armazenamento e manipulação de imagens médicas para ambientes virtuais de diagnóstico a distância.

4. ARQUITETURA PROPOSTA E PROTÓTIPO

Neste capítulo serão apresentados os resultados obtidos com a utilização das tecnologias estudadas ao longo deste trabalho.

4.1 ARQUITETURA PROPOSTA

Neste tópico serão abordadas as tecnologias propostas para a construção da arquitetura, bem como o funcionamento de cada uma no contexto da arquitetura proposta.

4.1.1 TECNOLOGIAS PROPOSTAS

Diante do estudo realizado ao longo deste trabalho, as tecnologias propostas para o desenvolvimento de um ambiente virtual de diagnóstico a distância são:

- Solução de banco de dados - Oracle 11g;
- Tecnologia Web - JSP e Servlets; e
- Tecnologia de Servlet Container - Tomcat 6.0 (TOMCAT, 2010).

A razão da escolha do Tomcat 6.0 foi devido à sua fácil integração com o Eclipse⁷ (ECLIPSE, 2010) e por ser amplamente utilizado por desenvolvedores de aplicações web. Entretanto, a utilização do Tomcat 6.0 não é fator determinante para o sucesso da arquitetura proposta, diferentemente das outras tecnologias sugeridas. Seu uso é recomendado por não oferecer custos.

4.1.2 FUNCIONAMENTO DA ARQUITETURA

A arquitetura proposta, mostrada na Figura 15, apresenta como será feita a integração entre as tecnologias escolhidas e como será seu funcionamento. A arquitetura é composta da solução de banco de dados, do servidor Web, do filtro de

⁷ Ambiente de desenvolvimento Java.

segurança, e das páginas JSP e Servlets. Cada componente tem uma função específica e bem definida, como detalhado abaixo:

- **Servidor Web** - O apache 6.0 faz o papel do Servidor Web (ServletContainer) da arquitetura proposta. A função do servidor web é a recepção, o processamento e resposta das requisições do browser cliente. A comunicação entre o browser e o Servidor Web é realizada através do protocolo HTTP (HTTP, 2010).
- **Filtro de Segurança** - O filtro tem o papel de interceptar as requisições que são enviadas as Servlets, com o objetivo de verificar se o usuário requisitante tem permissão para executar Servlets ou acessar a página JSP requisitada;
- **Servlet / JSP** - Servlets e as páginas JSP concentram todo a lógica da aplicação. Neste componente as requisições são processadas e os resultados enviados ao browser cliente. A comunicação com o sistema de banco de Dados (Oracle 11g) é realizada através de JDBC, de forma segura e eficiente; e
- **Oracle 11g** - SGBD responsável pelo armazenamento de todos os dados da aplicação, ou seja tanto o conteúdo DICOM quanto o textual são armazenados no mesmo banco de dados, facilitando o processo de recuperação e manipulação da informação. No SGBD também é realizado processamento de dados, como consultas de recuperação de metadados DICOM e recuperação de imagens por conteúdo.

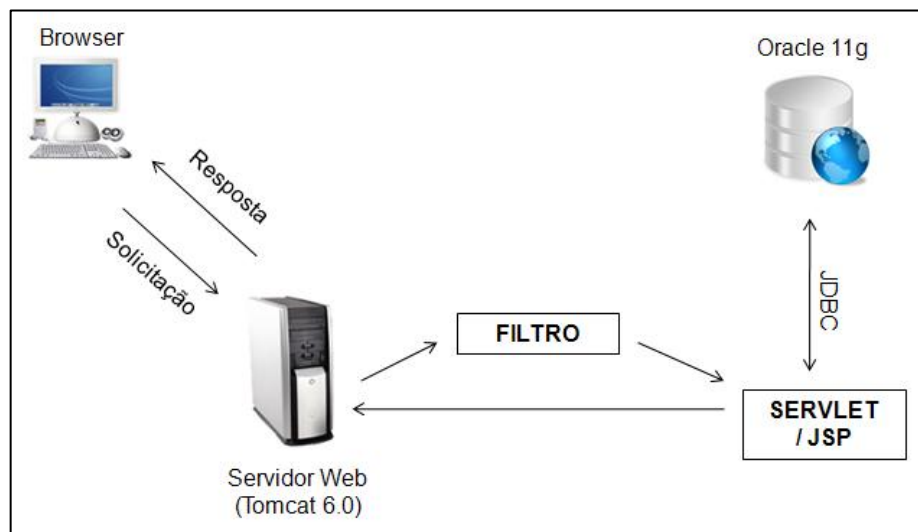


Figura 15 - Arquitetura proposta.

4.2 DESENVOLVIMENTO DO PROTÓTIPO

O protótipo desenvolvido agrega algumas funcionalidades críticas referentes ao armazenamento e manipulação de imagens num ambiente virtual de diagnóstico a distância. Estas funcionalidades são:

- Cadastro de Médicos - Os usuários do protótipo são os médicos, e nesta funcionalidade é possibilitado o cadastramento de novos médicos no banco de dados;
- Gerenciamento de Sessão - Através do mecanismo de *login*/senha, as páginas e funcionalidades da aplicação só poderão ser acessadas mediante a uma autenticação prévia do médico;
- Cadastro de Exames - Esta funcionalidade é a responsável por cadastrar os exames no banco de dados. Os referidos exames são enviados ao servidor no formato DICOM;
- Recuperação de Exames - O usuário tem acesso a uma página de pesquisa de Exames, os quais podem ser visualizados e baixados em arquivos DICOM. A busca dos exames é realizada através da consulta aos metadados DICOM, que estão armazenados no banco de dados; e
- Recuperação de Exames por Conteúdo - O usuário pode requisitar a recuperação de exames similares ao exame selecionado. A busca dos exames é realizada através da comparação das características das imagens. Depois de realizado o processamento, os exames são exibidos e disponibilizados para *download* no *browser* do usuário.

4.2.1 RECURSOS DE HARDWARE E SOFTWARE

O ambiente utilizado no desenvolvimento do servidor Web foi o Eclipse, enquanto que o SQL Developer (SQL-DEVELOPER, 2010) foi utilizado na configuração e desenvolvimento dos scripts do banco de dados. A configuração da máquina utilizada como servidor segue abaixo:

- Sistema operacional - Windows XP Professional 32 bits (WINDOWS, 2010);
- Memória RAM - 2GB 667 MHz;
- Disco Rígido - 120GB Sata II;
- Processador: - Core 2 duo 2.2 GHz;

No servidor foram instalados os seguintes componentes:

- Oracle 11g;
- Tomcat 6.0;

4.2.2 ESTRUTURA E FUNCIONAMENTO DAS CLASSES

O protótipo foi desenvolvido em camadas, e contém os seguintes pacotes de classes Java: jdbc, modelo, dao, filtro e servlet.

O pacote jdbc (Figura 16) contém a classe ConnectionFactory, que é a fábrica de conexões da aplicação. Toda conexão com o banco de dados é feita através desta classe.

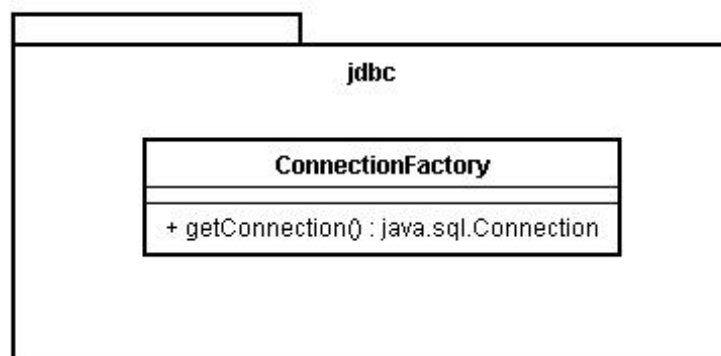


Figura 16 – Classes do pacote jdbc.

O método *getConnection()* da classe *ConnectionFactory* é utilizado para abrir a conexão com o banco de dados através da API JDBC.

O pacote modelo (Figura 17) contém as classes: *AtributosXML*, *Exame* e *Medico*, que representam a modelagem dos dados da aplicação, os objetos que são recuperados e inseridos no banco de dados. Todas as classes do pacote modelo têm métodos públicos para manipulação dos seus atributos.

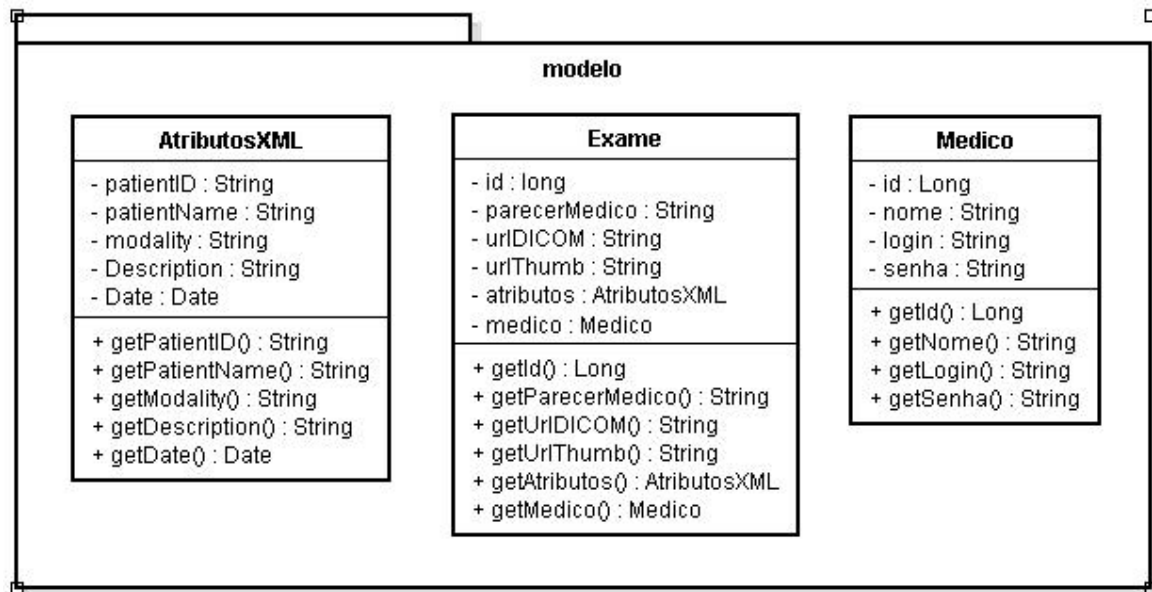


Figura 17 – Classes do pacote modelo.

O pacote dao (Figura 18) contém duas classes: *MedicoDAO* e *ExameDAO*, que são utilizadas para chamar os procedimentos específicos e recuperar dados do banco de dados.

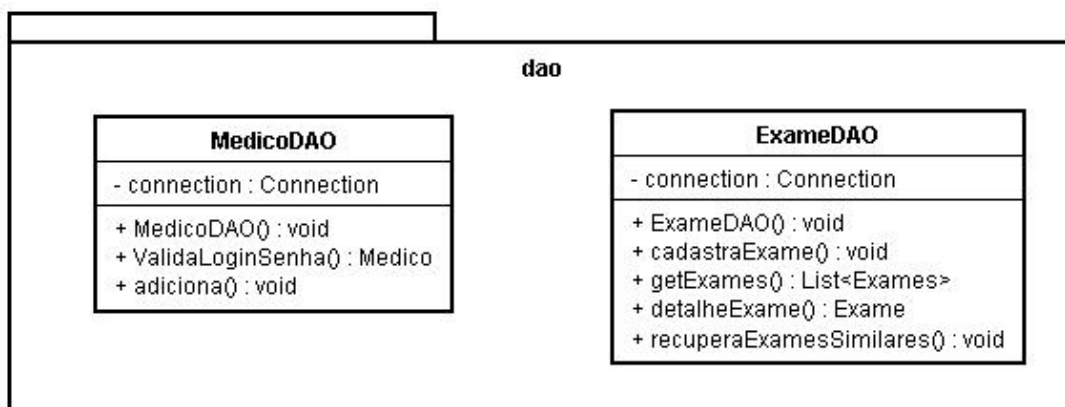


Figura 18 - Classes do pacote dao.

A classe *MedicoDAO* contém os seguintes métodos:

- *MedicoDAO()* - Construtor da classe *MedicoDAO*, e tem a função de abrir a conexão com o banco de dados no momento da criação do objeto;
- *ValidaLoginSenha()* - Método que tem a função de validar com o banco de dados se a autenticação do usuário está correta; e
- *Adiciona()* - Método utilizado para o cadastro de novos Médicos no banco de dados.

A classe *ExameDAO()* possui os seguintes métodos:

- *ExameDAO()* - Construtor da classe *ExameDAO*, e tem a função de abrir a conexão com o banco de dados no momento da criação do objeto;
- *cadastraExame()* - Método utilizado para cadastrar um novo exame no banco de dados;
- *getExames()* - Método utilizado para recuperação dos exames cadastrados no banco de dados;
- *detalheExame()* - Método utilizado para executar os procedimentos de geração de *thumbnail* e exportação do arquivo DICOM. Esses arquivos vão ser visualizados e disponibilizados no *browser* do usuário; e
- *recuperaExamesSimilares()* - Método utilizado para executar o procedimento de recuperação de exames similares a imagem selecionada. Os exames similares encontrados são apresentados pelo *browser* ao usuário.

O pacote filtro (Figura 19) contém apenas uma classe, a *FiltroSegurança*, que é a responsável pela segurança do sistema.



Figura 19 - Classes do pacote Filtro.

A classe *FiltroSeguranca* contém o método *doFilter()*, responsável por verificar se o usuário tem permissão para acessar a página JSP ou Servlet requisitada.

Por fim, o pacote servlet (Figura 20) contém servlets responsáveis por receber as requisições do *browser*, executar os métodos correspondentes do pacote dao e retornar a resposta ao usuário. A função de cada servlet segue abaixo:

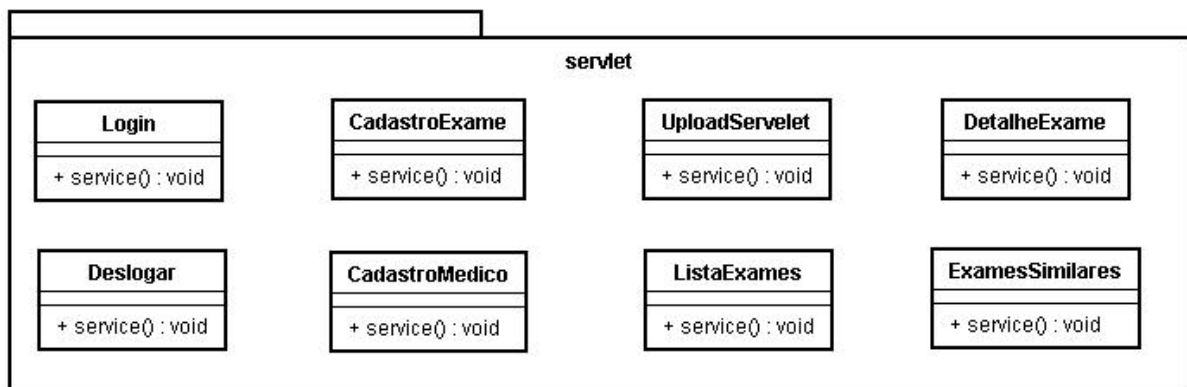


Figura 20 - Classes do pacote servlet

- *Login* - responsável por efetuar a autenticação do usuário, e caso seja válida redirecionar o usuário para a página inicial do sistema;
- *Deslogar* - utilizada para interromper a sessão do usuário no sistema;
- *CadastroExame* - cadastra um exame no banco de dados. O método *cadastraExame()* da classe *ExameDAO* é executado nesse procedimento;

- *CadastroMedico* - cadastra um médico no banco de dados. O método *adiciona()* da classe *MedicoDAO* é executado no procedimento;
- *UploadServlet* - responsável por fazer o upload do arquivo DICOM do exame, passado pelo usuário, para o sistema de arquivos do servidor. O upload é uma etapa anterior ao cadastramento do exame no banco de dados;
- *ListaExames* - utilizada para recuperar os exames cadastrados no banco de dados. O método *getExames()* da classe *ExameDAO* é executado no procedimento;
- *DetalheExames* - utilizada para recuperar informações específicas de um exame, como: parecer médico, *thumbnail* e médico que cadastrou o exame. O método *detalhaExame()* da classe *ExameDAO* é executado no procedimento;
e
- *ExamesSimilares* - utilizada para recuperar os exames similares ao exame selecionado. O método *recuperaExamesSimilares()* da classe *ExameDAO* é executado e os exames recuperados apresentados ao usuário via *browser*.

4.2.3 TABELAS DO BANCO DE DADOS

O banco de dados consiste de duas tabelas: Médicos e Exames. A tabela de Médicos possui os campos: id, nome, login e senha. A tabela de Exames possui os campos: id, id_medico, parecerMedico, dicom, imageThumb, imageSignature. O modelo lógico é mostrado na Figura 21.

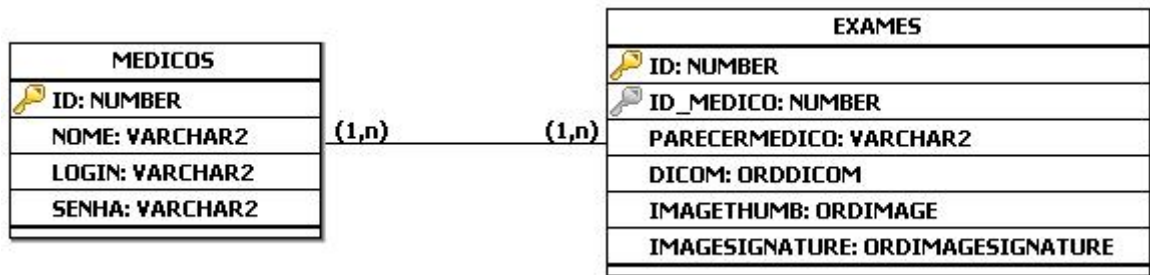


Figura 21 - Modelo lógico do banco de dados.

Na tabela Medicos, os campos possuem as seguintes funções:

- **Id** - Chave primária da tabela;
- **Nome** - Armazena o nome do Médico;
- **Login** - Armazena o login do Médico; e
- **Senha** - Armazena a senha do Médico.

Na tabela de Exames, os campos possuem as seguintes funções:

- **Id** - Chave primária da tabela;
- **Id_Medico** - Chave estrangeira, que estabelece o relacionamento com a tabela de Medicos;
- **ParecerMedico** - Armazena o parecer dado pelo Médico;
- **Dicom** - Armazena o conteúdo DICOM;
- **ImageThumb** - Armazena o thumbnail gerado para o conteúdo DICOM; e
- **ImageSignature** - Armazena a assinatura do campo ImageThumb do Exame.

4.2.4 PROCEDIMENTOS DO BANCO DE DADOS

Neste tópico serão apresentados os scripts das *procedures* utilizadas no armazenamento, processamento e recuperação dos dados no Oracle 11g. Abaixo segue a lista de todas as *procedures*, com seus respectivos códigos e funcionamento:

- PROC_GERA_THUMB - este procedimento gera o *thumbnail* do conteúdo DICOM e a assinatura da imagem do *thumbnail* gerado. Neste procedimento são utilizados os métodos: *processCopy()*, que é nativo ao objeto *OrdDicom* e *generateSignature()*, que é nativo ao objeto *OrdImagemSignature*. Código mostrado na Figura 22.

```

create or replace
procedure PROC_GERA_THUMB(source_id number, verb varchar2)
is
  dcmSrc    ordsys.orddicom;
  imgDst    ordsys.ordimage;
  v_imagem_assinatura ORDSYS.ORDImageSignature;
begin
  select dicom, imageThumb
  into dcmSrc, imgDst
  from EXAMES
  where id = source_id for update;
  dcmSrc.processCopy(verb, imgDst);

  update EXAMES
  set imageThumb = imgDst
  where id = source_id;

  select imagesignature into v_imagem_assinatura from exames where id = source_id for update;
  v_imagem_assinatura.generateSignature(imgDst);
  update exames set imagesignature = v_imagem_assinatura where id = source_id;
  commit;
end;

```

Figura 22 - Código do procedimento PRO_GERA_THUMB

- PROC_INSERE_EXAME - este procedimento insere um exame no banco de dados, importa o conteúdo DICOM e chama o procedimento PROC_GERA_THUMB. Código mostrado na Figura 23.

```

create or replace procedure PROC_INSERE_EXAME(filename varchar2, idmedico number, parecermedico varchar2) is
  dcm ordsys.orddicom;
  idExame number;
begin
  idExame := SQ_ID_EXAME.NEXTVAL;
  insert into EXAMES (id, dicom, imageThumb, anonDicom, idmedico, parecermedico, imageSignature)
  values (idExame, ordsys.orddicom('file', 'IMAGEDIR', filename, 0),
         ordsys.ordimage.init(), ordsys.orddicom(), idmedico, parecermedico, ORDSYS.ORDImageSignature.init())
  returning dicom into dcm;
  dcm.import(1);

  update EXAMES set dicom=dcm where id=idExame;

  commit;

  PROC_GERA_THUMB(idExame, 'fileformat=jfif fixedscale=350 350');
end;

```

Figura 23 - Código do procedimento PROC_INSERE_EXAME.

- PROC_EXPORTA_DICOM - procedimento que exporta o conteúdo DICOM para um arquivo no formato .dcm. Neste procedimento é utilizado o método *export()*, nativo ao objeto OrdDicom. Código mostrado na Figura 24.

```

create or replace procedure PROC_EXPORTA_DICOM
(source_id number, filename varchar2)
as
  dcmSrc ordsys.orddicom;
begin
  select dicom into dcmSrc from EXAMES where id = source_id;
  dcmSrc.export('FILE', 'SITEDIR', filename);
end;

```

Figura 24 - Código do procedimento PROC_EXPORTA_DICOM.

- PROC_EXPORTA_THUMB - procedimento que exporta o *thumbnail* para um arquivo JPEG. Neste procedimento é utilizado o método *export()*, nativo ao objeto OrdImage. Código mostrado na Figura 25.

```

create or replace procedure PROC_EXPORTA_THUMB (source_id number, filename varchar2) as
  imgSrc ordsys.ordimage;
  ctx raw(64) := null;
begin
  select t.imageThumb into imgSrc from EXAMES t where t.ID = source_id;
  imgSrc.export(ctx, 'FILE', 'SITEDIR', filename);
end;

```

Figura 25 - Código do procedimento PROC_EXPORTA_THUMB.

- PROC_RETORNA_EXAMES - procedimento que retorna todos os exames cadastrados na tabela EXAMES. Neste procedimento é utilizado o método *extractValue()*, que extrai um metadado específico do conteúdo DICOM. Os seguintes metadados são recuperados na consulta: nome do paciente, modalidade do exame, data do exame, e a descrição do exame. A consulta também possui um filtro por nome do paciente. Código mostrado na Figura 26.

```

create or replace
PROCEDURE PROC_RETORNA_EXAMES
( nomePaciente IN varchar2,
  o_cursor OUT sys_refcursor)
IS
BEGIN
  if nomePaciente is not null then
    OPEN o_cursor FOR
      select id, IDMEDICO,
        extractValue(t.dicom.metadata,
          '/DICOM_OBJECT/*[@name="Patient''s Name"]/VALUE',
          'xmlns=http://xmlns.oracle.com/ord/dicom/metadata_1_0') as "PATIENT_NAME",
        extractValue(t.dicom.metadata,
          '/DICOM_OBJECT/*[@name="Patient ID"]',
          'xmlns=http://xmlns.oracle.com/ord/dicom/metadata_1_0') as "PATIENT_ID",
        extractValue(t.dicom.metadata,
          '/DICOM_OBJECT/*[@name="Modality"]',
          'xmlns=http://xmlns.oracle.com/ord/dicom/metadata_1_0') as "MODALITY",
        extractValue(t.dicom.metadata,
          '/DICOM_OBJECT/*[@name="Study Date"]',
          'xmlns=http://xmlns.oracle.com/ord/dicom/metadata_1_0') as "STUDY DATE",
        extractValue(t.dicom.metadata,
          '/DICOM_OBJECT/*[@name="Image Type"]',
          'xmlns=http://xmlns.oracle.com/ord/dicom/metadata_1_0') as "STUDY DESCRIPTION"
      from EXAMES t
      where extractValue(t.dicom.metadata,
        '/DICOM_OBJECT/*[@name="Patient''s Name"]/VALUE',
        'xmlns=http://xmlns.oracle.com/ord/dicom/metadata_1_0') LIKE concat(concat('%',nomePaciente),'%')
      order by id asc;
  end if;
END;

```

Figura 26 - Código do procedimento PROC_RETORNA_EXAMES.

- PROC_RETORNA_EXAMES_SIMILARES - procedimento que retorna os exames similares a um exame especificado. Os exames encontrados tem seu conteúdo DICOM exportado para arquivos .dcm, além de ter seus *thumbnails* exportados para arquivos JPEG. Neste procedimento é utilizado o método *ImgSimilar()*, que compara duas assinaturas de imagens, verificando se são similares. Código mostrado na Figura 27.

```

create or replace
PROCEDURE PROC_RETORNA_EXAMES_SIMILARES
(IDEXAME INTEGER, color VARCHAR2, texture VARCHAR2, shape VARCHAR2, location VARCHAR2, limite INTEGER) AS

v_imagem_assinatura ORDSYS.ORDImageSignature;

BEGIN
SELECT imageSignature INTO v_imagem_assinatura FROM EXAMES WHERE id = IDEXAME;

FOR v_exames IN
(SELECT id
 FROM EXAMES
 WHERE id <> IDEXAME
  AND ORDSYS.IMGSimilar (imageSignature, v_imagem_assinatura,
    'color = ''' || color || ''', '''
    'texture = ''' || texture || ''', '''
    'shape = ''' || shape || ''', '''
    'location = ''' || location || ''', limite) = 1)
LOOP
PROC_EXPORTA_THUMB(v_exames.id,concat(v_exames.id,'.jpg'));
PROC_EXPORTA_DICOM(v_exames.id,concat(v_exames.id,'.dcm'));
END LOOP;

```

Figura 27 - Código do procedimento PROC_RETORNA_EXAMES_SIMILARES.

4.2.5 DESEMPENHO DA API JAVA DICOM

A API Java DICOM permite uma integração de forma simplificada entre a aplicativo e o banco de dados, através da utilização do tipo de dados OrdDicom da API. Utilizando a API é possível recuperar o conteúdo DICOM do banco de dados diretamente no objeto criado na aplicação. Após essa recuperação realizada, para se obter metadados do conteúdo DICOM o método *setPropreties()* deverá ser executado. Tal método recupera todos os metadados do conteúdo DICOM, e os armazena no objeto OrdDicom do aplicativo, após esse procedimento é possível a recuperação de cada atributo específico com a utilização de um outro método, o *getAttributeByName()*.

Apesar de bastante simples, essa abordagem não possui um desempenho satisfatório quando comparada com a recuperação dos atributos através de *Procedures* do banco de dados. Um estudo comparativo entre a utilização da API Java DICOM e o procedimento PROC_RETORNA_EXAMES foi realizado. A figura 28 apresenta um gráfico comparativo entre as duas abordagens.

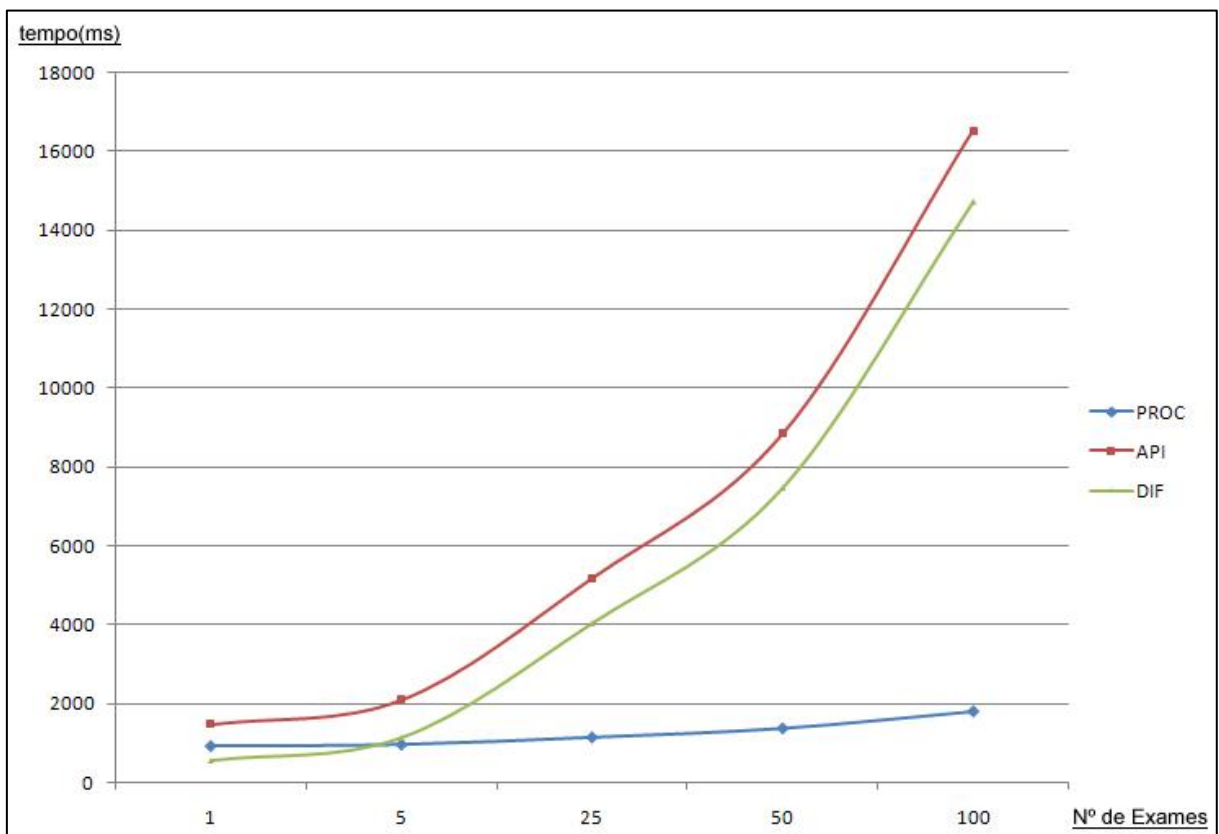


Figura 28 - Gráfico comparativo entre o uso do PROC_RETORNA_EXAMES e API Java DICOM.

O resultado mostra que para o mesmo volume de dados, o tempo de recuperação da informação é bem diferente nas duas abordagens, aumentando de forma significativa na utilização da API quando o volume dos dados recuperados é maior. Devido aos resultados apresentados nos testes, a API Java DICOM não foi utilizada na recuperação do conteúdo DICOM neste projeto.

4.2.6 FUNCIONALIDADES IMPLEMENTADAS

Neste tópico serão apresentadas as telas finais do processo de desenvolvimento do protótipo.

➤ Gerenciamento de Sessão

A Figura 29 apresenta a tela de autenticação do usuário, que é exibida na abertura do sistema ou quando o usuário tenta acessar uma área restrita a usuários autenticados,

A imagem mostra uma caixa de diálogo de autenticação com um fundo cinza. No topo, há o rótulo "Login:" seguido de um campo de entrada de texto contendo o texto "Insira o login". Abaixo disso, há o rótulo "Senha:" seguido de um campo de entrada de texto com caracteres ocultos por pontos. Na base da caixa, há um botão "Ok" com um efeito de sombra.

Figura 29 - Tela de autenticação.

Depois de autenticado, o usuário estará apto a acessar as funcionalidades do sistema. A Figura 30 mostra a tela inicial após a autenticação bem sucedida. Ela contém além do menu da aplicação, informações sobre a sessão estabelecida pelo usuário.

	Márcio Dias Costa	PROTÓTIPO - Arquitetura de armazenamento e manipulação de imagens médicas para ambientes virtuais de diagnóstico à distância.
<p><u>MENU</u></p> <p>Cadastro Médicos</p> <p>Cadastro Exames</p> <p>Pesquisa Exames</p> <hr/> <p>Logado como: <i>mdckoury</i> Desde: 21/06/2010 18:41 Deslogar</p>	<p>BEM VINDO!</p> <p>MÁRCIO DIAS COSTA</p> <p>Usuário logado como: <i>mdckoury</i> Sessão estabelecida em: 21/06/2010 18:41</p>	

Figura 30 - Tela de navegação do sistema.

➤ Cadastro de Médicos

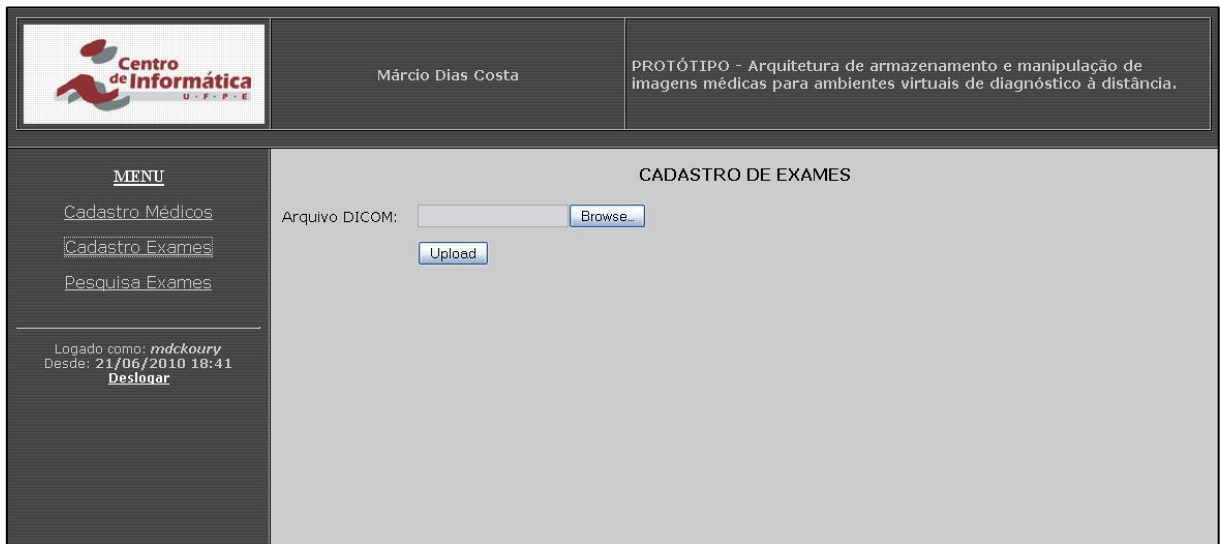
A Figura 31 apresenta a tela de cadastro dos médicos, que é utilizada para coletar as informações passadas pelo usuário e executar o procedimento de cadastro.

	Márcio Dias Costa	PROTÓTIPO - Arquitetura de armazenamento e manipulação de imagens médicas para ambientes virtuais de diagnóstico à distância.
<p><u>MENU</u></p> <p>Cadastro Médicos</p> <p>Cadastro Exames</p> <p>Pesquisa Exames</p> <hr/> <p>Logado como: <i>mdckoury</i> Desde: 21/06/2010 18:41 Deslogar</p>	<p>CADASTRO DE MÉDICOS</p> <p>Nome: <input type="text"/></p> <p>Login: <input type="text"/></p> <p>Senha: <input type="password"/></p> <p><input type="button" value="Gravar"/></p>	

Figura 31 - Tela de cadastro de médicos.

➤ Cadastro de Exames

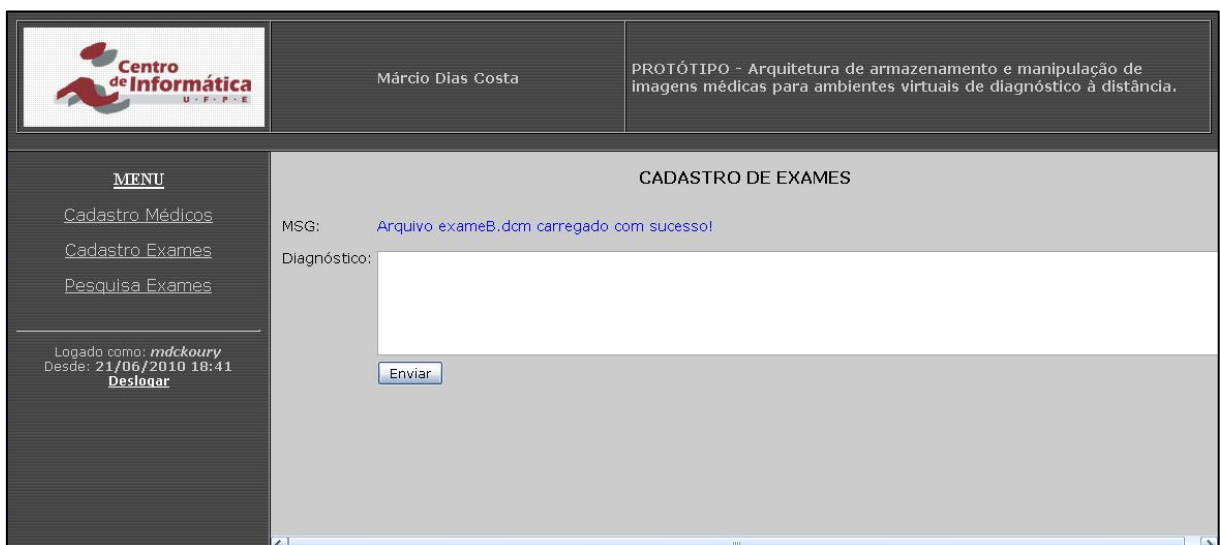
O cadastro de exames é realizado em duas etapas. A primeira é o *upload* do arquivo, que é selecionado na tela apresentada na Figura 32.



The screenshot shows the 'Cadastro de Exames' web application interface. At the top, there is a header with the logo of 'Centro de Informática U.F.P.R.E.' on the left, the user name 'Márcio Dias Costa' in the center, and the title 'PROTÓTIPO - Arquitetura de armazenamento e manipulação de imagens médicas para ambientes virtuais de diagnóstico à distância.' on the right. Below the header, there is a sidebar menu on the left with the following items: 'MENU', 'Cadastro Médicos', 'Cadastro Exames', and 'Pesquisa Exames'. The main content area is titled 'CADASTRO DE EXAMES' and contains a form with the following elements: 'Arquivo DICOM:' followed by a text input field, a 'Browse...' button, and an 'Upload' button. At the bottom of the sidebar, there is a login status: 'Logado como: mdckoury', 'Desde: 21/06/2010 18:41', and a 'Deslogar' button.

Figura 32 – Primeira tela do cadastro de exames.

A segunda etapa é a informação do diagnóstico realizado pelo médico e mostrado na tela apresentada na Figura 33.



The screenshot shows the 'Cadastro de Exames' web application interface after a successful upload. The header and sidebar are identical to Figure 32. The main content area is titled 'CADASTRO DE EXAMES' and displays a success message: 'MSG: Arquivo exameB.dcm carregado com sucesso!'. Below the message, there is a 'Diagnóstico:' label followed by a large text input field and an 'Enviar' button.

Figura 33 – Segunda tela do cadastro de exames.

➤ Recuperação de Exames

A recuperação dos exames é feita através da tela apresentada na Figura 34. Os usuários podem fazer buscas por nome do paciente e visualizar detalhes de um exame específico. Os metadados são exibidos em uma tabela, que possui um botão “visualizar”, utilizado para exibir os detalhes de um exame específico.

Id	Nome Paciente	Modalidade	Descrição	Data	Login Médico	
111	PATIENT G	MR	MAGNETIC RESONAN	2006-12-01	mdckoury	Visualizar
112	PATIENT H	MR	MAGNETIC RESONAN	2006-12-01	mdckoury	Visualizar
113	PATIENT D	CR	COMPUTED TOMOGRA	2008-12-01	mdckoury	Visualizar
114	PATIENT I	MR	MAGNETIC RESONAN	2006-12-01	mdckoury	Visualizar

Figura 34 - Tela de pesquisa de exames.

A tela de detalhamento do exame é apresentada na Figura 35.

Médico:	Márcio Dias Costa
Parecer:	Parecer teste.
	Imagens Similares

Figura 35 - Tela de detalhamento dos exames.

➤ Recuperação de Exames Similares

Através da tela de detalhamento dos exames, é possível chamar um procedimento que recupera exames similares ao selecionado, que estão cadastrados no banco de dados. A tela de recuperação de exames similares é apresentada na Figura 36.

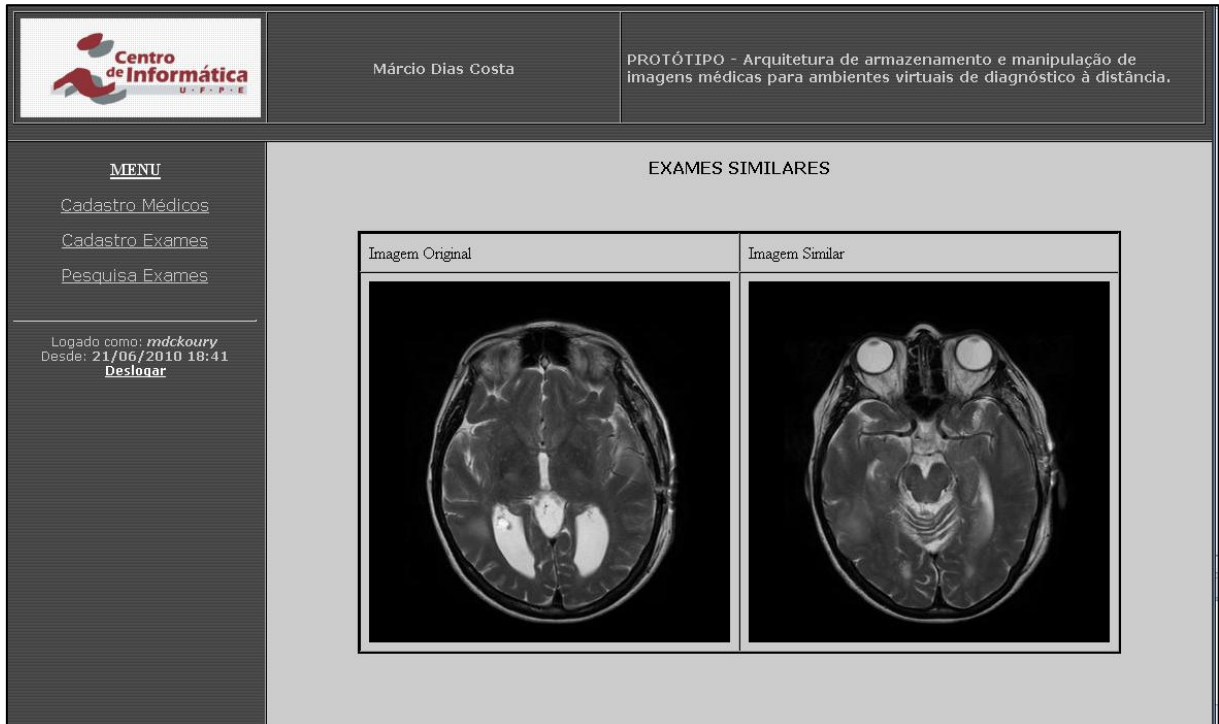


Figura 36 - Tela de exames similares.

5. CONCLUSÃO

Os estudos realizados neste trabalho mostraram a deficiência das atuais arquiteturas de armazenamento de imagens médicas. Deficiência devido ao não suporte do formato DICOM no SGBD utilizado, que é o formato padrão no compartilhamento e armazenamento de imagens médicas.

A arquitetura proposta neste trabalho supriu tal deficiência. Com o seu uso, é possível o armazenamento do padrão DICOM no SGBD de forma eficiente, bem como a recuperação das imagens por metadados e/ou por suas características. Através desta arquitetura, o processo de armazenamento e distribuição das imagens médicas é facilitado. A partir do momento que o padrão DICOM é utilizado, todas suas vantagens são incorporadas à solução.

Os estudos das tecnologias utilizadas na construção da arquitetura mostraram que o uso da API Java DICOM não é eficiente. Desta forma, seu uso não é aconselhado, podendo tornar a aplicação lenta para recuperação de um grande volume de dados.

5.1 TRABALHOS FUTUROS

- A fragilidade da API Java Dicom, no aspecto do desempenho, deixa uma lacuna para futuros estudos no desenvolvimento de formas mais eficientes na recuperação de metadados pela da API; e
- O volume de dados e a organização XML dos metadados do formato DICOM sugerem estudos futuros de Business Intelligence nesta área, que visariam organizar os metadados armazenados, auxiliando o processo de tomada de decisão em diagnósticos médicos;

REFERÊNCIAS

ACR. **American College of Radiology**. Disponível em: <<http://www.acr.org/>>. Acesso em: 25 jun. 2010.

ANNAMALAI, Melliyal. **Oracle Database 11g DICOM Medical Image**. Disponível em:<http://www.oracle.com/technology/products/multimedia/pdf/11gr2_collateral/dicom11gr2_wp_medimgsupport.pdf>. Acesso em: 01 jun. 2010.

AVI. **Audio Video Interleave**. Disponível em: <<http://pt.wikipedia.org/wiki/AVI>>. Acesso em: 25 jun. 2010.

BARALE, Rafael Ferreira. **DESENVOLVIMENTO DE UM SISTEMA DE VENDAS NA WEB UTILIZANDO JSP**. 2007. 77 f. Dissertação (Tcc) - Curso de Bacharelado Em Sistemas De Informação, Uniminas, Uberlândia, 2007.

BMP. **BITMAP**. Disponível em: <<http://en.wikipedia.org/wiki/Bitmap>>. Acesso em: 25 jun. 2010.

CAELUM. **FJ-21: Java para Desenvolvimento Web**. Disponível em: <<http://downloads.caelum.com.br/apostila/caelum-java-web-fj21.pdf>>. Acesso em: 03 abr. 2010.

CARITÁ, E. C. et al. Implantação de PACS com Suporte à Recuperação de Imagens Baseada em Conteúdo em Hospital Universitário. In: X Congresso Brasileiro de Informática em Saúde, Florianópolis – SC, 2006.

CUNHA, Roberto de Oliveira. **Dicom e XML**. Departamento de Engenharia de Telecomunicações – Universidade Federal Fluminense (UFF). Disponível em: <<https://docs.google.com/viewer?url=http://www.midiacom.uff.br/~debora/fsmm/trab-2006-1/dicomxml.pdf>>. Acesso em: 05 abr. 2010.

DICOM. **DICOM Standard**. Disponível em: <<ftp://medical.nema.org/medical/dicom/2009/>>. Acesso em: 13 abr. 2010.

ECLIPSE. **Eclipse**. Disponível em: <<http://www.eclipse.org>>. Acesso em: 13 abr. 2010.

FRANCESCHI, Wagnes Borges. **PROCEDIMENTOS E PRÁTICAS PARA DIGITALIZAÇÃO DE IMAGENS MÉDICAS**. 2006. 124 f. Dissertação (Mestrado) - Curso de Programa De Pós-graduação Em Engenharia Elétrica E Informática Industrial - Cpgei, Universidade Tecnológica Federal Do Paraná, Curitiba, 2006.

FREITAS, Cilas De. **UMA ARQUITETURA BASEADA EM PADRÕES ABERTOS PARA VISUALIZAÇÃO CIENTÍFICA VIA INTERNET APLICADA À MEDICINA**. 2002. 74 f. Dissertação (Mestrado) - Curso de Pós-graduação em Informática, Departamento de Setor de Ciências Exatas, Universidade Federal do Paraná, Curitiba, 2002.

GIF. Graphics Interchange Format. Disponível em: <http://pt.wikipedia.org/wiki/Graphics_Interchange_Format>. Acesso em: 25 jun. 2010.

HASEGAWA, Fabio Massao; AIRES, João Paulo. Proposta de um Padrão de Metadados Para Imagens Médicas. In: ERI-PR, 14., 2007, Ponta Grossa - PR. **Anais...** . Ponta Grossa - PR: Escritório de Relações Internacional, 2007. p. 48 - 56. Disponível em: <www.sbc.org.br/bibliotecadigital/download.php?paper=688>. Acesso em: 03 abr. 2010.

HENRIQUE NETO, Geraldo; OLIVEIRA, Wdson De; VALERI, Fabio Valiengo. **Armazenamento de Imagens Médicas com InterBase.** Disponível em: <<http://www.dcc.ufla.br/infocomp/artigos/v3.1/art03.pdf>>. Acesso em: 06 abr. 2010.

HTTP. Hypertext Transfer Protocol. Disponível em: <http://pt.wikipedia.org/wiki/Hypertext_Transfer_Protocol>. Acesso em: 25 jun. 2010.

INTERBASE. INTERBASE OPEN SOURCE. Disponível em: <<http://info.borland.com/devsupport/interbase/opensource/>>. Acesso em: 25 jun. 2010.

JDBC. JDBC™ Database Access. Disponível em: <<http://java.sun.com/docs/books/tutorial/jdbc>>. Acesso em: 25 jun. 2010.

JPEG. JPEG STANDARD. Disponível em: <<http://www.jpeg.org/jpeg/index.html>>. Acesso em: 25 jun. 2010.

JSP. Java Server Pages Technology. Disponível em: <<http://java.sun.com/products/jsp/>>. Acesso em: 25 jun. 2010.

JVM. Java Virtual Machine Specification Disponível em: <<http://java.sun.com/docs/books/jvms/>>. Acesso em: 25 jun. 2010.

KRATOCHVIL, Marcel. **The move to store images in the database.** Disponível em: <http://www.oracle.com/technology/products/multimedia/pdf/general/why_images_in_database.pdf>. Acesso em: 13 jun. 2010.

MPEG. MPEG Reference. Disponível em: <<http://www.mpeg.org>>. Acesso em: 25 jun. 2010.

MYSQL. MYSQL Open Source Database Disponível em: <<http://www.mysql.com/>>. Acesso em: 25 jun. 2010.

NEMA. National Electrical Manufacturers Association. Disponível em: <<http://www.nema.org>>. Acesso em: 25 jun. 2010.

ORACLE (Comp.). **API Reference: OrdImageSignature.** Disponível em: <http://download-west.oracle.com/docs/cd/B15904_01/web.1012/b14025/oracle/ord/im/OrdImageSignature.html>. Acesso em: 10 jun. 2010.

ORACLE 8g. **Oracle Database Release 8.0.6 Documentation**. Disponível em: <<http://www.oracle.com/technology/documentation/oracle8.html>>. Acesso em: 25 jun. 2010.

ORACLE 11g. **Oracle Database 11g Release 2**. Disponível em: <<http://www.oracle.com/technology/products/database/oracle11g/index.html>>. Acesso em: 25 jun. 2010.

ORDDICOM. **Oracle Multimedia DICOM Java API Reference**. Disponível em: <http://download.oracle.com/docs/cd/B28359_01/appdev.111/b28417/oracle/ord/dicom/OrdDicom.html>. Acesso em: 25 jun. 2010.

PELSKI, Sue. **Oracle Multimedia: DICOM Developer's Guide**. Disponível em: <http://download.oracle.com/docs/cd/E11882_01/appdev.112/e10778/ch_cncpt.htm>. Acesso em: 01 jun. 2010.

PL/SQL. **Oracle PL/SQL Technical Information**. Disponível em: <http://www.oracle.com/technology/tech/pl_sql/index.html>. Acesso em: 25 jun. 2010.

PNG. **Portable Network Graphics**. Disponível em: <<http://www.libpng.org/pub/png/>>. Acesso em: 25 jun. 2010.

POSTGRES. **PostgreSQL Network Graphics**. Disponível em: <<http://www.postgresql.org/>>. Acesso em: 25 jun. 2010.

PSD. **ADOBE PHOTOSHOP DOCUMENT**. Disponível em: <<http://www.adobe.com/br/products/photoshop/>>. Acesso em: 25 jun. 2010.

QUICKTIME. **Apple Quicktime**. Disponível em: <<http://www.apple.com/quicktime/>>. Acesso em: 25 jun. 2010.

RAW. **OpenRAW**. Disponível em: <<http://www.openraw.org/>>. Acesso em: 25 jun. 2010.

ROSA, Natália Abdala et al. **Recuperação de Imagens Médicas por Similaridade em um Hospital Universitário**. Disponível em: <www.inf.ufpr.br/tps06/IntegracaoPACS/Reclma_HCRP.pdf>. Acesso em: 25 out. 2010.

SERVLET. **Java Servlet Technology**. Disponível em: <<http://java.sun.com/products/servlet/>>. Acesso em: 25 jun. 2010.

SILVA, Everton Rodrigues. **Desenvolvimento de ambiente virtual cooperativo de diagnóstico à distância, integrado a ferramentas de manipulação de imagens**. 2010. Dissertação em desenvolvimento. (Mestrado) - Curso De Pós-graduação Em Banco de dados – Centro de Informática, Universidade Federal de Pernambuco.

SILVA, Ronaldo Gomes da. **PADRÃO DE COMUNICAÇÃO DE IMAGENS MÉDICAS**. Disponível em: <<http://www.scribd.com/doc/9692832/Padrao-DICOM>>. Acesso em: 07 maio 2010.

SQL. **Structured Query Language**. Disponível em: <<http://www.sql.org>>. Acesso em: 25 jun. 2010.

SQL-DEVELOPER. **Oracle SQL Developer**. Disponível em: <http://www.oracle.com/technology/products/database/sql_developer/index.html>. Acesso em: 25 jun. 2010.

SQL-SERVER. **Microsoft SQL-Server**. Disponível em: <<http://www.microsoft.com/everybodysbusiness/pt/br/products/sql-server2008.aspx>>. Acesso em: 25 jun. 2010.

TEMPLE, A; DE MELLO, R. F; CALEGARI, D. T; SCHIEZARO, M. **JSP, Servlets e J2EE**. 2004. Disponível em <<http://143.107.183.115/mello/outros/livro-v03-figuras.pdf>> Acessado em 20 jun. 2010.

TIFF. **Tagged Image File Format**. Disponível em: <http://pt.wikipedia.org/wiki/Tagged_Image_File_Format/>. Acesso em: 25 jun. 2010.

TOMCAT. **Apache Tomcat**. Disponível em: <<http://tomcat.apache.org>>. Acesso em: 25 jun. 2010.

WINDOWS. **Microsoft Windows XP**. Disponível em: <<http://www.microsoft.com/windows/windows-xp/default.aspx>>. Acesso em: 25 jun. 2010.

XML. **Extensible Markup Language**. Disponível em: <<http://www.w3.org/XML>>. Acesso em: 25 jun. 2010.