UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Thiago Coelho Prado

OTIMIZAÇÃO DA PERSISTÊNCIA DE DADOS EM PACS EMPREGANDO MODELOS DE DADOS HIERÁRQUICOS INDEXADOS

Dissertação submetido(a) ao Programa de Pós Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Ciência da Computação
Orientador: Prof. Dr. Aldo von Wangenheim

Florianópolis
2012

Catalogação na fonte pela Biblioteca Universitária
da
Universidade Federal de Santa Catarina

P896o Prado, Thiago Coelho
Otimização da persistência de dados em PACS empregando modelos de dados hierárquicos indexados [dissertação] / Thiago Coelho Prado ; orientador, Aldo von Wangenheim. – Florianópolis, SC, 2012.
123 p.: il., grafs., tabs., mapas

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Informática. 2. Ciência da computação. 3. DICOM (Protocolo de rede de computadores). 4. Telemedicina. I. Wangenheim, Aldo v. (Aldo von). II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.
CDU 681

Thiago Coelho Prado

OTIMIZAÇÃO DA PERSISTÊNCIA DE DADOS EM PACS EMPREGANDO MODELOS DE DADOS HIERÁRQUICOS INDEXADOS

Esta Dissertação foi julgado(a) adequado(a) para obtenção do Título de Mestre, e aprovada em sua forma final pelo Programa de Pós Graduação em Ciência da Computação

Florianópolis, 27 de Fevereiro de 2012.

_____
Prof. Dr. Ronaldo do Santos Mello.
Coordenador do Curso

Banca Examinadora:

_____
Prof. Dr. rer.nat. Aldo von Wangenheim,
Orientador
Universidade Federal de Santa Catarina

_____
Prof., Dr. Nelson Francisco Favilla Ebecken,
Universidade Federal do Rio de Janeiro

_____
Prof., Dr. Mario Antonio Ribeiro Dantas,
Universidade Federal de Santa Catarina

_____
Prof., Dr. Renato Fileto,
Universidade Federal de Santa Catarina

AGRADECIMENTOS

À minha família por todo apoio e incentivo.

À Ciça por toda paciência e por estar presente sempre nos bons e nos poucos maus momentos.

Ao Prof. Aldo von Wangenheim pela orientação e oportunidade de trabalho junto ao Grupo Cyclops.

Às amizades que cultivo desde o início da graduação e me acompanham até hoje.

Ao Prof. Mario Dantas, Douglas Macedo, Rafael Andrade, Alexandre Savaris pelas discussões produtivas, ao Cloves Langendorf Barcelos Jr, por me manter focado durante o período de pesquisa.

A todos os colegas do LABTELEMED, em especial Marcus Vinícius, Marcone Magnus, Felipe Duarte, Richard Martini, Well, Harley Wagner, Alexandre, Luiz Köhler, pelos momentos de descontração.

Bons amigos, bons livros e uma consciência sonolenta: essa é a vida ideal

(Mark Twain, 1898)

RESUMO

A variedade de modalidades produzidas pelo aumento de dispositivos que sejam capazes de gerar imagens médicas em formato DICOM tem crescido. Consequentemente, com a digitalização dos setores de radiologia, as bases de dados de exames podem ultrapassar as barreiras de *terabytes*. Fato este, combinado com a lei brasileira de obrigatoriamente manter exames radiológicos arquivados por no mínimo 20 anos, afeta diretamente a gestão de grandes volumes de dados. Este trabalho apresenta o desenvolvimento, avaliação empírica e discussão sobre o emprego do formato de dado hierárquicos HDF5 em conjunto com o núcleo de indexação e busca Apache Lucene. Para a validação da camada de persistência, foram realizados testes de desempenho entre a abordagem proposta contra o modelo em banco de dados relacional PostgreSQL usado na Rede Catarinense de Telemedicina. O desempenho do serviço de armazenamento, em conformidade com o padrão DICOM, foi analisado variando o volume de dados transmitidos entre cliente e servidor. As buscas sobre informações de um exame já realizado abrangeram cenários típicos como a consulta de exames de um paciente específico ou exames pertencentes a um intervalo temporal de dados. A recuperação avaliou três distintos casos: estudo, série ou imagem. Embora mostrando um desempenho superior em cenários de consulta com o uso do PostgreSQL, o uso combinado de HDF5 com Apache Lucene apresenta um meio eficiente de armazenar e recuperar imagens médicas DICOM, pois une o alto desempenho do HDF5 com a funcionalidade de consulta dos dados indexados. Neste ponto, o uso de uma camada para gerenciamento de exames médicos em DICOM, empregando o HDF5 juntamente com Apache Lucene, se apresentou como uma alternativa viável e eficiente. Além disso, trabalhos futuros podem aproveitar algumas características do formato HDF5, como obtenção direta do conteúdo da imagem médica e metadados por ferramentas para auxílio ao diagnóstico, tais como visão multimodal e processamento digital de imagens.

Palavras-chave: DICOM. HDF5. Lucene. PostgreSQL. PACS. Telemedicina.

ABSTRACT

The variety of modalities produced by the increasing number of devices that are capable of generating large sets of medical images using DICOM standard has been growing. Consequently, with the ongoing process of digitization of radiology sectors, the examinations database can exceed the terabytes barrier. This fact, combined with the Brazilian law of storing radiological examinations archived for at least 20 years, directly affects the management of large volumes of data. This paper presents the concept, development, assessment and discussion of the combined employment of the hierarchical data format HDF5 with the search engine library Apache Lucene. The validating process of the persistence layer, was carried out by performance tests over the proposed approach compared to the current model based on PostgreSQL and used in the Santa Catarina's Telemedicine Network. The storage service performance in compliance with the DICOM standard was analyzed by interchanging the volume of data transmitted between client and server. The search covered typical scenarios such as search examinations within a time interval or for specific patient. Retrieval operations evaluated three different cases: study, series or image. Although showing superior performance while using PostgreSQL in querying scenarios, the combined use of HDF5 with Apache Lucene provides an suitable way to store and retrieve medical images in DICOM, because it combines the high performance of HDF5 with the query functionality of the indexed data. At this point, the use of a management layer for medical examinations in DICOM, combined with HDF5 and Apache Lucene presented itself as an efficient and viable option. In addition, future studies can take advantage of some features from the HDF5 library, such as direct medical image content retrieval and metadata for computed aid diagnosis, such as multimodal vision and digital image processing.

Keywords: DICOM. HDF. Lucene. PostgreSQL. PACS. Telemedicine.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACID - Atomicity, Consistency, Isolation, Durability
ANSI - American National Standards Institute
API - Application Programming Interface
BLOB - Binary Large Object
CFM - Conselho Federal de Medicina
DICOM - Digital Imaging and Communications in Medicine
DIMSE - DICOM Message Service Element
GIST - Generalized Search Tree
GIN - Generalized Inverted Index
HIS - Hospital Information System
HL7 - Health Level Seven
IEC - International Electrotechnical Commission
IOD - Information Object Definition
ISO - International Organization for Standardization
JPEG - Joint Photographic Experts Group
MPI - Message Passing Interface
NETCDF - network Common Data Format
OID - Object Identifier
PACS - Picture Archiving and Communication Sistema
RCTM - Rede Catarinense de Telemedicina
RIS - Radiology Information System
RSL - Revisão Sistemática de Literatura
SCP - Service Class Provider
SCU - Service Class User
SGBDR - Sistema de Gerenciamento de Banco de Dados Relacional
STT/SC - Sistema Catarinense de Telemedicina e Telessáude
TCP - Transmission Control Protocol
TOAST - The Oversized-Attribute Storage Technique
UID - Unique Identifier
XML - Extensible Markup Language

SUMÁRIO

1. INTRODUÇÃO

Neste capítulo é apresentado o cenário que envolve sistemas computacionais para arquivamento e transferência de dados na área de medicina, bem como o problema de gerenciamento de dados que devem estar disponíveis por décadas. Além disso, também são mostrados os objetivos para criação de um mecanismo para resolução do problema exposto.

1.1 CONTEXTUALIZAÇÃO DO PROBLEMA

O período de tempo que um exame radiológico deve ser disponibilizado é diretamente dependente da legislação de cada país. Nos Estados Unidos varia de 5 a 10 anos, sendo que o tempo pode ser estendido se houver requerimento do governo (ACR, 2008). Em alguns países, há prazos específicos de acordo com a modalidade do exame, por exemplo, na Inglaterra é de 9 a 15 anos para mamografia (RCR, 2008) e na Alemanha é 30 anos para raios-x (BMJ, 1987). No Brasil, o Conselho Federal de Medicina (CFM) estabeleceu que a disponibilidade de acesso de um exame é de no mínimo 20 anos (CFM, 2007) independente da modalidade. Além disso, as imagens não podem possuir nenhum tipo de perda na qualidade da imagem, o que poderia induzir a um diagnóstico equivocado.

Estas necessidades de tempo de armazenamento e disponibilidade para recuperação implicam na gestão de grandes volumes de dados e trazem consequências diretas quanto à infraestrutura necessária para armazenar e recuperar esses dados de maneira eficiente utilizando os modelos tradicionais para armazenamento. Desde 2002 (CFM, 2002) o CFM definia que a linguagem padrão para manipulação dos dados digitais era o SQL (*Structured Query Language*), que é específica de sistemas de gerenciamento de banco de dados relacional, porém, tal limitação sobre qual tecnologia deve ser utilizada foi revogada em 2007(CFM, 2007), abrindo caminho para pesquisas de gerenciamento de dados através de outras tecnologias.

Os sistemas que oferecem suporte para o armazenamento, a visualização e a recuperação de imagens médicas surgiram no início da década de 80 (van de Wetering e Batenburg, 2009) e são conhecidos como Sistemas de Comunicação e Arquivamento de Imagens (*PACS - Picture Archiving and Communications Systems*) e têm a responsabilidade de aumentar a eficiência do *workflow* e reduzir os

custos operacionais nos setores radiológicos (Huang, 2010). Tais ambientes utilizam, para interoperabilidade entre alguns elementos do PACS, o formato de imagens médicas digitais e protocolo de comunicação DICOM (*Digital Image and Communications in Medicine*). O arquivamento e recuperação eficiente de exames médicos em formato digital tem se mostrado um grande desafio, pois o protocolo DICOM foi projetado para interoperabilidade e não eficiência. Há, por exemplo, abordagens desenvolvidas pela iniciativa privada para recuperação e visualização de imagens DICOM que foca no protocolo de transferência de dados DICOM, como a solução *open source* chamada MINT (*Medical Imaging Network Transport*)[1] desenvolvida nos Estados Unidos pela Universidade John Hopkins (Pianykh, 2012) e similar à uma funcionalidade já existente no padrão DICOM que emprega *webservices* (NEMA, 2011j).

O acesso a exames que fazem parte do histórico do paciente é fundamental para acompanhamento da evolução do seu estado e pode influenciar diretamente os pedidos de novos exames. Por exemplo, se um paciente já realizou exames que envolvem a irradiação para aquisição da imagem médica, ou seja, se foi exposto a níveis de radiação como raios-x, é importante que exames posteriores desse tipo sejam descartados a fim de evitar exposições (Ron, 2003).

Além disso, o uso de um formato de dados que represente naturalmente um conjunto de imagens obtidas no processo de realização de um exame pode facilitar o acesso às informações médicas. Sendo assim há potencial para emprego na área de processamento de imagens, visualização multimodal ou outra qualquer aplicação que visa prover semântica associada aos dados de imagem.

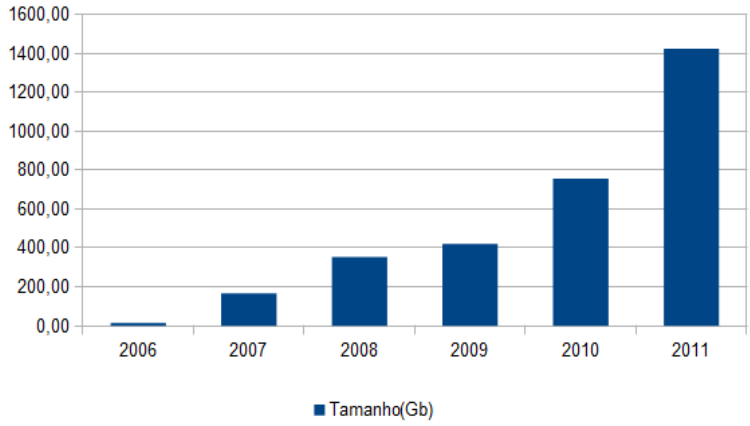
A necessidade da criação de uma estratégia de armazenamento altamente escalável, com possibilidade da extensão para diferentes formatos e o acesso para quaisquer tipos de dados originados nos meios científicos e na engenharia, assim como a preservação de dados por longo prazo, para que os dados estejam disponíveis por décadas ou até mesmo séculos motivaram a criação do padrão HDF (Hierarchical Data Format) (HDFGROUP, 2011a). O HDF é um formato de dados amplamente utilizado em projetos que lidam com coleta de dados atmosféricos, dados para estudo no campo aeroespacial, área financeira ou epidemiologia, dentre outros (HDFGROUP, 2011b) e sua versão mais atual é chamada de HDF5.

[1] http://code.google.com/p/medical-imaging-network-transport/

Como fruto da parceria entre a Universidade Federal de Santa Catarina (UFSC) e a Secretaria de Saúde do Estado de Santa Catarina surgiu o STT/SC (Sistema Catarinense de Telemedicina e Telessáude) [2] que pertence à RCTM (Rede Catarinense de Telemedicina) (Maia *et al.*, 2006) (Wallauer *et al.*, 2008) e oferece uma ampla paleta de serviços para o suporte à atenção à saúde, entre eles telediagnóstico por imagem em eletrocardiografia, dermatologia e muitas áreas da radiologia, segunda opinião formativa e formação continuada. A meta do STT/SC é digitalizar e integrar todos os serviços de diagnóstico por imagem da rede pública de Santa Catarina. Os hospitais pertencentes ao STT/SC possuem equipamentos que têm capacidade de produzir exames no formato digital DICOM, reduzindo ou ainda eliminando os custos relacionados à produção de filmes radiológicos e a necessidade de espaços físicos para armazená-los.

O aumento do volume de dados médicos anuais coletados nas bases de dados pertencentes ao STT/SC, que pode ser visto na Figura 1, gerou a necessidade de se desenvolver novas metodologias de armazenamento. Tal aumento decorre do crescimento do número de hospitais que utilizam o STT/SC e do aumento de implantação de dispositivos capazes de produzir imagens digitais.

Figura 1 - Evolução do tamanho (em *gigabytes*) da base de imagens médicas no padrão DICOM no STT/SC



[2] http://telessaude.sc.gov.br

O comportamento do gráfico de crescimento da base de dados de exames no formato DICOM e as características do HDF5 levaram a partir de 2008 (Macedo *et al.*, 2008; Amaral, 2009; de Macedo *et al.*, 2011) à realização de uma série de experimentos, com resultados promissores. A avaliação deste trabalho estava dentro do escopo de armazenamento de imagens médicas no padrão DICOM em ambiente distribuído comparando-se o formato HDF5 com a utilização do PostgreSQL, que atualmente é a tecnologia de persistência empregada no STT/SC. Além disso, algumas características do HDF5, como o suporte a acesso randômico aos dados ou leitura e escrita dos dados de forma paralela através do emprego de MPI, abrem caminho para novas pesquisas.

Neste contexto, o objetivo do presente trabalho é o estudo e avaliação empírica de uma metodologia eficiente para armazenamento, busca e recuperação de imagens médicas mantendo a interoperabilidade oferecida pelo padrão DICOM. Os métodos utilizados são originados em pesquisas anteriores relacionados à utilização de um formato de dados de alto desempenho em ambiente distribuído. O objetivo é prover para as interfaces de serviços de armazenamento, busca e recuperação oferecidos pelo ambiente PACS uma camada para utilização do formato de dados hierárquicos HDF5.

Entretanto, diferente do modelo relacional que tem integrado a linguagem SQL (*Structured Query Language*) para gerenciamento dos dados o HDF5 não disponibiliza meios de manipulação dos dados armazenados. Dessa forma, aproveitando a característica do modelo de busca e recuperação do padrão DICOM, a estrutura de dados arquivo invertido foi escolhida para manipulação dos metadados de um exame em formato digital DICOM.

1.2 OBJETIVOS

Neste subcapítulo é mostrado o objetivo geral, bem como são enumerados os objetivos específicos para conclusão do trabalho.

1.2.1 Objetivo Geral

Prover uma camada de acesso a imagens médicas transmitidas e armazenadas em conformidade com o padrão DICOM. A camada proposta deve ser escalável e realizar o armazenamento, a busca e a recuperação de exames com eficiência superior ao modelo atual.

1.2.2 Objetivos Específicos

Obj.1. Escolha e integração do mecanismo de indexação escolhido ao meio de persistência de imagens médicas no padrão DICOM;

Obj.2. Implementação de uma camada que disponibilize interfaces para armazenamento, busca e recuperação. Estas interfaces devem ser utilizadas pelas classes de serviço descritas do padrão DICOM;

Obj.3. Definição de cenários em que serviços que se comuniquem pelo protocolo definido no padrão DICOM são utilizados;

Obj.4. Avaliar métricas de desempenho obtidas no cenário de execução em um ambiente PACS a fim de validar a viabilidade do emprego em ambiente real das tecnologias empregadas.

1.3 CONTEXTUALIZAÇÃO DO PROBLEMA

Esta dissertação está organizado em seis capítulos. No capítulo dois, é apresentada uma sumarização do estado da arte e o processo de condução de pesquisa e levantamento de trabalhos correlatos. No terceiro capítulo está presente o embasamento teórico dos padrões e tecnologias relevantes para este trabalho. No quarto capítulo são mostrados detalhes dos métodos empregados para implementação das camadas de persistência. No quinto capítulo são vistos os resultados experimentais entre o modelo atual de arquivamento de exames utilizados no STT/SC e o modelo proposto. Por fim, no capítulo seis são apresentadas as conclusões, discussão e trabalhos futuros.

2. ESTADO DA ARTE

Tradicionalmente o modelo relacional é empregado para armazenamento na informática médica e telemedicina assim como em diversas áreas. Dessa forma, são encontrados alguns projetos na área de arquivamento e gerenciamento de imagens médicas que lidam com tal modelo, como o *middleware* TRENCADIS (Blanquer *et al.*, 2006) desenvolvido em Valência na Espanha, que é um serviço de *grid* para compartilhamento de exames DICOM que incluí ontologias; o projeto nacional eDiaMoND de base de dados nacional de mamografias no Reino Unido (Power *et al.*, 2004); o serviço de *grid* de imagens médicas desenvolvido no CERN (Montagnat *et al.*, 2008) e também na Colômbia o projeto mantisGRID para o mesmo fim (Garcia Ruiz *et al.*, 2011). Nos Estados Unidos o projeto Globus MEDICUS (Erberich *et al.*, 2007) oferece uma rede de compartilhamento de exames entre mais de cinquenta hospitais.

Exceto por trabalhos do mesmo grupo (Macedo *et al.*, 2008; Amaral, 2009; de Macedo *et al.*, 2011), não foram encontrados na literatura pesquisas sobre a aplicação do formato HDF5 como meio de persistir exames em DICOM, uma vez que normalmente as soluções PACS existentes no mercado tendem a utilizar sistema de arquivos ou SGBDR. Inclusive vale citar que o SGBDR Oracle oferece solução integrada para manipulação do conteúdo (vídeos, laudo estruturado, formas de onda) e aos mais de 2000 metadados dos exames em formato DICOM (ORACLE, 2011).

Sendo assim, foram pesquisados como outros trabalhos gerenciam dados que estão armazenados de maneira hierárquica e também pesquisas que tenham como tema o modelo de busca e recuperação DICOM. Tal pesquisa foi executada através de um processo chamado Revisão Sistemática da Literatura (RSL).

2.1 REVISÃO SISTEMÁTICA DA LITERATURA

De acordo com (Kitchenham, 2004) a RSL estabelece critérios rigorosos de pesquisa a fim de sintetizar toda literatura relevante de forma imparcial, que siga um conjunto de passos para tornar a pesquisa reprodutível. Portando, aqui são descritos como foi executada cada etapa da RSL.

2.1.1 Contextualização da Pesquisa

O formato HDF5 apresentou resultados favoráveis para uso em ambiente distribuídos para armazenamento em imagens médicas. Porém, o HDF5 não oferece mecanismos de consultas para recuperação posterior e, portando, há uma lacuna no escopo do gerenciamento dos dados contidos no formato hierárquico. A integração do HDF5 com um mecanismo de consulta do conteúdo dos metadados de exames médicos em DICOM é o tema da pesquisa apresentada neste trabalho.

2.1.2 Questões Específicas da Pesquisa

Que mecanismos podem ser integrados ao HDF5 para indexação dos metadados dos objetos DICOM semelhantes à interface de gerenciamento de dados SQL?

Como prover uma maneira eficiente de armazenamento, busca e recuperação de dados de imagens médicas através dos serviços que seguem conformidade com o padrão DICOM e que estes dados estejam persistidos por uma camada que acesse o formato HDF5?

2.1.3 Definições de Bases de Dados e Termos de Pesquisa

As fontes escolhidas nas quais serão realizadas as pesquisas dos termos relevantes são:

- ACM Digital Library
- IEEExplore
- Springer Link
- PubMed
- Science Direct

As pesquisas realizadas abrangeram os trabalhos publicados desde 1993 até o primeiro semestre de 2011. A escolha da data de origem dá-se no lançamento da versão inicial do padrão DICOM. Os termos que serão incluídos deverão responder as questões relacionadas ao modelo de persistência de dados dos exames DICOM (*query*, *retrieve*, DICOM, *storage*) e a busca e recuperação dos dados no formato HDF5 (*hdf*, *hierarchical data format*, *query*, *indexing*, *index*). Para um primeiro refinamento, foi retirado o termo *web* pois eram

encontradas trabalhos de outros escopos. Além disso, a pesquisa foi limitada somente ao título, *abstract* e palavras-chave.

2.1.4 Procedimento e Critério de Seleção Dos Estudos

Todos os artigos selecionados foram publicados na língua inglesa no período de janeiro de 1993 até junho de 2011. Foram levados em consideração artigos de conferências ou publicados em journals. Os estudos considerados como relevantes tem como tema de pesquisa a demonstração de alternativas ao armazenamento e recuperação de imagens DICOM em relação ao modelo tradicional, de preferência em ambientes em larga escala. Também são avaliadas técnicas de busca e recuperação de maneira semântica de dados armazenados no formato HDF5.
A filtragem inicial sobre os resultados de pesquisa nas bases de busca seguiram o processo de leitura do título e *abstract* do artigo. Isso resultou em um total de 27 artigos. Posteriormente a leitura completa de cada um dos trabalhos selecionados foi feita para embasar a pesquisa deste trabalho.

2.1.5 Lista de Requerimento e Critério de Qualidade Dos Estudos

Somente foram levados em consideração estudos que continham experimentos realizados. Estudos que apresentaram pesquisas relacionadas aos escopos específicos, como a construção de bases de imagens para recuperação baseada em contexto ou similares foram descartados, pois o foco da pesquisa não é a concepção do modelo HDF5 para uso de alguma base em particular e sim uma camada para uso de qualquer aplicação de domínios específicos dentro da área médica.

2.1.6 Estratégia para Extração dos Dados

Para cada estudo selecionado como relevante foram extraídas as informações pertinentes e classificadas em cada um dos tópicos abaixo:

- Identificação do Estudo (autores e data);
- Escopo do Estudo;
- Descrição dos modelos de persistência presentes;
- Descrição dos modelos de gerenciamento dos dados;
- Experimentos realizados;

- Discussão dos resultados;

2.1.7 Estratégia para Sintetização dos Dados

Os dados foram dispostos na forma de tabela de forma sumarizada para análise dos estudos selecionados durante a condução da revisão sistemática.

2.1.8 Identificação dos Estudos Relevantes

A Tabela 1 contém as bases, termos pesquisados e a quantidade de resultados encontrados.

Tabela 1 - Número de Estudos Encontrados durante RSL

Base	***Termos***	***Número de resultados***
ACM Digital Library	((Abstract:DICOM and (Abstract:query or Abstract:retrieval or Abstract:store or Abstract:storage)) and (not Abstract:web)) or ((Abstract:hdf) AND (Abstract:query or Abstract:retrieval or Abstract:index or Abstract:indexing))	4
IEEExplore	(("Abstract":DICOM) AND ("Abstract":query OR "Abstract":retrieval OR "Abstract":store OR "Abstract":storage) AND NOT "Abstract":web) OR (("Abstract":hdf OR "Abstract":hierarchical data format) AND ("Abstract":indexing OR "Abstract":index OR "Abstract":query OR "Abstract":retrieval))	113
Springer Link	'ab:(((dicom and (query or retrieval or storage)) and not web) or (hdf and (index or query or retrieval)))' published between '1 Jan 1993' and '28 Sep 2011'	78
PubMed	((DICOM[Title/Abstract] AND (query[Title/Abstract] OR retrieve[Title/Abstract] OR store[Title/Abstract] OR storage[Title/Abstract])) NOT	191

Base	*Termos*	*Número de resultados*
	web[Title/Abstract]) OR ((hdf[Title/Abstract] or hierarchical data format[Title/Abstract]) AND (retrieval[Title/Abstract] OR query[Title/Abstract] OR index[Title/Abstract] OR indexing[Title/Abstract]))	
Science Direct	((Title-Abstr-Key(DICOM) AND (Title-Abstr-Key(query) OR Title-Abstr-Key(retrieval) OR Title-Abstr-Key(store) OR Title-Abstr-Key(storage))) AND NOT Title- Abstr-Key(web)) OR ((Title-Abstr-Key(hdf) OR Title-Abstr-Key(hierarchical data format)) AND (Title-Abstr-Key(query) OR Title-Abstr-Key(retrieval) OR Title-Abstr-Key(indexing) OR Title-Abstr-Key(index)))	57
	Total	**443**

2.1.9 SUMÁRIO DOS ESTUDOS RELEVANTES SELECIONADOS

A Tabela 2 resume as principais informações extraídas dos estudos considerados relevantes.

Tabela 2 - Sumário sobre os campos e seus valores sobre os estudos relevantes

Campo	*Conteúdo Extraído*
Autor(es)	Gosink, L. Shalf, J. Stockinger, K. Wu, K. Bethel, W.
Ano	2006
Título	HDF5-FastQuery: Accelerating Complex Queries on HDF Datasets using Fast Bitmap Indices
Domínio	Gerenciamento de dados científicos oriundos de experimentos de física.

Modelo de persistência	Hirarchical Data Format 5 (HDF5)
Modelo de gerenciamento dos dados	Índice Bitmap desenvolvido na pesquisa, chamado HDF5-FastQuery, utilizando uma implementação de bitmap *open source* chamada FastBit; Árvore- R*, *open source* e empregada em pesquisa similar anterior com HDF5 em (Nam e Sussman, 2003); Interface de leitura nativa do HDF5 em linguagem C
Experimentos realizados	Comparação da abordagem HDF5-FastQuery com solução em árvore-R* Comparação da abordagem HDF5-FastQuery com leitura nativa oferecida pela interface do HDF5
Discussão dos resultados	HDF5-FastQuery obteve acesso aos dados até duas vezes mais rápidas à dados multidimensionais em comparação à solução que usa interface nativa do HDF5. O modelo em árvore-R não suportou o aumento progressivo do volume de dados, apresentando falhas em um conjunto de informação consideradas pequenas pelo estudo;
Autor(es)	Costa, Carlos; Freitas, Filipe; Pereira, Marco; Silva, Augusto; Oliveira, José
Ano	2009
Título	Indexing and retrieving DICOM data in disperse and unstructured archives
Domínio	Busca e Recuperação de objetos DICOM
Modelo de persistência	Sistema de Arquivos
Modelo de gerenciamento dos dados	Índice Invertido, através da implementação do Apache Lucene original em Java
Experimentos realizados	Análise sobre o tempo de indexação de um volume de 24 mil imagens e variando o modelo de busca e recuperação, através da indexação dos campos obrigatórios descritos no padrão DICOM, indexação de todo documento ou indexação dos campos de busca e recuperação em conjunto com *thumbnail* do exame.
Discussão dos	A análise sobre os resultados de cada complexidade

resultados	adicionada ao modelo de busca e recuperação e a conclusão de que a nova forma de armazenamento e recuperação pode substituir ou estender o modelo tradicional de banco de dados relacional.
Autor(es)	Abduljwad, F.; Ning, W. ;De, X.
Ano	2010
Título	SMX/R: Efficient way of storing and managing XML documents using RDBMSs based on paths
Domínio	Aborda o problema de armazenamento, busca e recuperação de documentos no formato hierárquico XML
Modelo de persistência	Documentos no formato XML
Modelo de gerenciamento dos dados	Banco de dados relacional. Duas tabelas são usadas, a primeira para expressar a relação entre os nodos através de sete colunas e a segunda contem o caminho de dados interno do XML através de três colunas.
Experimentos realizados	Comparação com um modelo prévio no qual o trabalho foi baseado chamado de XRel. As análises comparativas feitas em relação ao número de operações de junções que devem ser utilizadas pelos dois modelos.
Discussão dos resultados	O modelo proposto superou o modelo pré existente nas análises, pois durante operações de busca apresentou um número menor e fixo de junções de tabelas, o que foi apresentado como uma vantagem sobre o outro modelo avaliado.
Autor(es)	Cohen, Shirley ; Hurley, Patrick ; Schulz, Karl W. ; Barth, William L. ; Benton, Brad
Ano	2006
Título	Scientific formats for object-relational database systems: a study of suitability and performance
Domínio	Pesquisa de extensão do modelo tradicional de armazenamento de dados comparando com modelo objeto relacional ou outros modelos para dados científicos, como o HDF5 e netCDF

Modelo de persistência	Banco de Dados Objeto Relacional, netCDF, HDF4 e HDF5
Modelo de gerenciamento dos dados	Interno do Bando de Dados Objeto Relacional e interfaces nativas de acesso ao netCDF, HDF4 e HDF5
Experimentos realizados	Acesso aos elementos de dados para realizar operações de agregação de dados como soma, média, obter valor máximo e mínimo.
Discussão dos resultados	Resultados favoráveis aos formatos científicos sobre o banco de dados. O formato que apresentou melhor resultado foi o HDF5 com desempenho superior em 200 % sobre o modelo mais inferior. Além disso, foi constatada que a dificuldade de administração do banco de dados era maior que no formato de dados científico, exigindo um administrador de sistemas de banco de dados.

Autor(es)	Folino, G.; Shah, A. A.; Kransnogor, N.
Ano	2009
Título	On the storage, management and analysis of (multi) similarity for large scale protein structure datasets in the grid.
Domínio	Comparação das similaridades entre proteínas e pós processamento dos resultados para criação de uma base de conhecimento.
Modelo de persistência	Banco de Dados Relacional e HDF5
Modelo de gerenciamento dos dados	SQL e interface nativa de acesso do HDF5
Experimentos realizados	Séries de algoritmos diferentes de comparação de similaridade de proteínas foram empregadas para acesso dos dados armazenados.
Discussão dos resultados	Desempenho significantemente superior na utilização do formato HDF5 sobre o banco de dados relacional, no caso o Oracle

Autor(es)	Sahoo, Swarup; Agrawal, Gagan
Ano	2005
Título	Supporting XML Based High-Level Abstractions on HDF5 Datasets: A Case Study in Automatic Data

	Virtualization
Domínio	Gerenciamento de dados em HDF5 através da disponibilização em alto nível por uma abstração da estrutura dos dados hierárquicos armazenados. Esses dados podem ser processados sem que se tenha conhecimento de como estão representados internamente através do fornecimento de um arquivo descritor.
Modelo de persistência	HDF5
Modelo de gerenciamento dos dados	XML Schema em conjunto com o XQuery, que posteriormente é traduzida por um compilador para funções de acesso de entrada e saída da interface nativa do HDF5
Experimentos realizados	Operações de recuperação variando-se o tamanho do conjunto ou o tamanho dos dados.
Discussão dos resultados	Resultados promissores em relação ao desempenho do código gerado pelo compilador para processar os dados armazenados que são acessados através da interface de programação do HDF5.

2.2 TRABALHOS CORRELATOS

Apesar da utilização do HDF5 ter se mostrado bastante promissora em experimentos anteriores que visavam avaliar o desempenho para operações de armazenamento e recuperação, detectou-se um problema durante a necessidade de busca dos dados. Diferente do banco de dado relacional, o HDF5 não oferece uma interface de gerenciamento de dados similar ao SQL, dificultando assim a busca e filtragem de dados de acordo com certas restrições requeridas.

Os trabalhos pesquisados foram investigados a fim de solucionar o problema de criar um sistema de gerenciamento de imagens DICOM contidos no HDF5. Assim, a procura dos trabalhos correlatos cobriu vários campos diferentes como, por exemplo: a indexação dos metadados dos objetos DICOM, a busca e recuperação de dados contidos em arquivos hierárquicos, por exemplo, o formato HDF5 ou XML, e a comparação da utilização de banco de dados relacional com outras formas de persistência de dados, como HDF5 e netCDF (*net*

Common Data Format, outro formato de arquivo para dados científicos).

A linha de pesquisa em (Gosink *et al.*, 2006) tem como opção de indexação de dados científicos em conjunto com o HDF5 através do uso de índices de *bitmaps* adicionando suporte à *queries* de intervalo dos dados. Os experimentos foram realizados sobre dados multidimensionais e a implementação chamada de *HDF5-FastQuery* obteve bons resultados de desempenho em relação ao acesso de dados nativos ao HDF5. Foram realizadas tentativas de comparação com uma implementação que utilizava *R*-Tree*, mas essa se mostrou instável com o aumento do volume dos dados. A pesquisa aproveita a propriedade de dados que não sofre atualizações, isto é, a característica de um conteúdo estar sempre disponível para leitura, sem modificações é uma das recomendações ao se empregar o uso do índice *bitmap*. Além disso, a informação contida no modelo hierárquico era composta de valores numéricos com intervalos já conhecidos, por exemplo, as temperaturas mínimas e máximas obtidas em experimentos científicos. Tal índice para os metadados contidos em exames DICOM não se enquadra somente nesta semântica de intervalo.

Como proposta de resolução de um problema semelhante ao encontrado aqui, o trabalho (Abduljwad *et al.*, 2010) trata de oferecer uma maneira eficiente para gerenciamento dos dados em XML através da avaliação do uso de tabelas de banco de dados relacional. O método consiste em primeiramente em processar o documento em XML para obter a árvore de relação entre os nodos, após essa etapa os dados são armazenados em duas tabelas, uma para representação da hierarquia entre os nodos e outras contendo os pares de caminho e valor.

Em (Sahoo e Agrawal, 2005) é abordado o problema de criar automaticamente serviços eficientes de manipulação de dados que ofereça uma abstração em alto nível de informações armazenadas em baixo nível. Neste caso, há um mapeamento da estrutura em HDF5 para XML-Schema com o objetivo de oferecer um meio de lidar com os dados em baixo nível através de funções em XQuery. Tais funções posteriormente são transformadas em funções da API nativas do HDF5. Dessa maneira, o requisitante do serviço obtém o resultado do processamento sem precisar estar ciente do *layout* do dado armazenado.

A obtenção do grau de similaridade entre um conjunto de proteínas gera um grande volume de dados que variam na sua representação e no seu tipo e necessitam de um meio ótimo de armazenamento e recuperação para que posteriormente sejam processados e avaliados a fim de gerar uma base de conhecimento para

entendimento das estruturas das proteínas. A escolha desse meio é avaliada através dos experimentos realizados em (Folino *et al.*, 2009). Tais experimentos visam mensurar o desempenho entre o modelo do SGBDR Oracle e o HDF5 e tem como conclusão que o uso do HDF5 é mais eficiente e ocupa menos espaço em disco diante de um ambiente com aumento de volume de dados.

Em (Cohen *et al.*, 2006), tendo como cenário dados obtidos de pesquisas no escopo meteorológico, é comparado a eficiência no emprego do tipo de dado em vetor chamado *VArray* e tabelas aninhadas disponibilizados pelo SGBDR Oracle com os containers de dados oferecidos pelos formatos de dados científicos netCDF, HDF4 e HDF5. O HDF5 se mostrou mais eficiente em relação a todos os outros meios em operações executadas em grandes volumes de dados.

A justificativa do estudo (Costa *et al.*, 2009) é a substituição ou extensão dos modelos tradicionais de armazenamento de dados através do emprego do mecanismo de indexação de documentos Lucene para prover uma interface de busca, com uma interface *web*, e recuperação de arquivos DICOM. O ambiente que foi validado tinha como função disponibilizar metadados de arquivos DICOM em *part 10* armazenados em estrutura de diretórios como descrito pelo padrão DICOM em (NEMA, 2011i). Além dos metadados contidos no modelo de busca e recuperação do DICOM (NEMA, 2011d) foram indexados todos os metadados contidos nos documentos e também *thumbnails* para facilitar pré visualizações dos estudos a serem recuperados. Apesar da proposta inicial, não foi realizado nenhum experimento comparativo com modelos de persistência de dados tradicionais, apenas foram avaliados configurações diferentes, como a exclusão da possibilidade de se indexar todo conteúdo ou os *thumbnails*.

De acordo com as características, que serão vistas mais adiante, do modelo de busca e recuperação em conformidade com o padrão DICOM e com o conteúdo obtido durante a fase de RSL optou-se pela utilização de indexação de termos de documentos. Ou seja, extraí-se a informação relevante de um exame em formato digital e a disponibiliza para consulta de forma que deva estar associada ao exame.

Uma vez que há disponível uma ampla variedade *open source* de motores de busca, torna-se desnecessária a criação de uma solução própria. Entre as ferramentas relevantes encontradas estão a indexação para busca textual oferecida por SGBDR, como o PostgreSQL e MySQL. O PostgreSQL oferece dois meios para recuperação de documentos, um que disponibiliza uma generalização de árvores balanceadas e outro que implementa índice invertido. Já o MySQL

oferece um índice do tipo *FULLTEXT*, que ao ser associado à uma coluna possibilita a consulta de termos de um documento.

Há também ferramentas projetadas especificamente para área de recuperação de informação. Dentre as principais, há o Apache Lucene (Lucene, 2011a), que disponibiliza diversos meios de personalização para melhor adequação ao ambiente em que irá ser utilizado e tem como um de seus usuários a enciclopédia colaborativa Wikipédia. A ferramenta Sphinx (Sphinx, 2011), empregada amplamente em diversas áreas, disponibiliza consulta pela linguagem SQL de documentos que podem estar armazenados em banco de dados relacional, bases NoSQL ou ainda documentos em sistemas de arquivo. O Xapian (Xapian, 2011) tem como principal funcionalidade o emprego de técnicas probabilísticas para recuperar documentos classificados de acordo com sua relevância. Estas bibliotecas disponibilizam resultados referentes à escalabilidade e desempenho que devem ser esperados [3].

[3] Lucene: http://lucene.apache.org/core/features.html
Sphinx: http://sphinxsearch.com/about/sphinx/
Xapian: http://xapian.org/docs/scalability.html

3. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são descritos todos os elementos utilizados neste trabalho, sendo eles o ambiente PACS, o padrão de interoperabilidade médica DICOM, o formato HDF5 para armazenamento dos exames em DICOM e a implementação do índice invertido.

Dentre as soluções existentes de índice invertido, a implementação Lucene oferece uma solução *open source* continuamente atualizada, com interface simples para programação e apresenta alto desempenho e escalabilidade.

Desta forma, foi combinado o uso do HDF5 em conjunto à implementação do Apache Lucene em linguagem "C++", chamada de CLucene (CLucene, 2011) para realização de busca aos dados armazenados e auxílio à recuperação de objetos DICOM a partir do HDF5.

3.1 DICOM

Como este trabalho propõe uma abordagem que modifica profundamente a filosofia tradicional de implementação de PACS, vamos apresentar brevemente o padrão DICOM contextualizando-o à luz de nosso enfoque.

O surgimento e propagação na década de 70 de equipamentos médicos, inicialmente tomografia computadorizada, que produziam imagens médicas em formato digital fez com que o Colégio Americano de Radiologia (*American College of Radiology* - ACR) e a Associação Nacional dos Fabricantes de Produtos Elétricos (*National Electrical Manufacturers Association* - NEMA) se unissem para criar uma padronização na transferência de imagens e troca de informação entre diferentes fabricantes. A esta padronização deu-se o nome de DICOM (*Digital Imaging and Communication in Medicine*) e um comitê composto de diversos grupos de trabalho, divididos por área, tem como responsabilidade a manutenção dos documentos relacionados ao padrão (NEMA, 2011a). A base da terceira e também atual versão do padrão é de 1993 e optou-se por ser a última, sendo que não há mais lançamento de novas versões e sim atualizações aplicadas regularmente.

Os documentos do padrão são divididos em partes com diferentes objetivos, dentre os quais têm grande importância para esse trabalho a padronização referente ao formato digital que as imagens devem ser armazenadas binariamente (NEMA, 2011i) em sistema de arquivos, o mapeamento dos objetos do mundo real para a estrutura de dados do

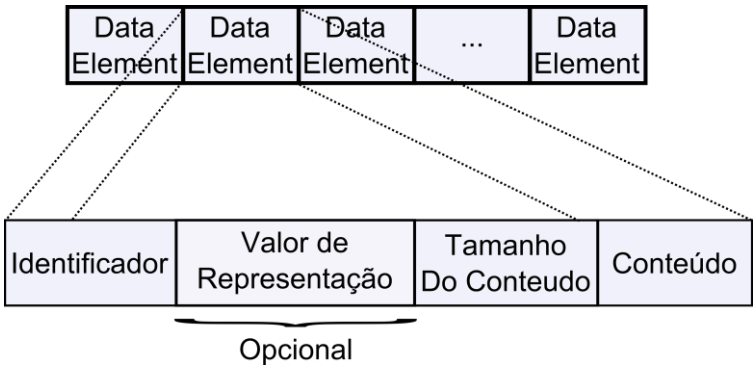
DICOM (NEMA, 2011c) e quais os serviços que um cliente/servidor DICOM pode prover (NEMA, 2011d).

Para assegurar a conformidade com o padrão os fabricantes de dispositivos e/ou software produzem documentos chamados *Conformance Statements* de acordo com um guia (NEMA, 2011b). O resultado são documentos que contém exatamente todas as informações necessárias como quais tipos de modalidades são suportados, sintaxes de transferência de dados, serviços que podem requisitar ou serem requisitados e quais possíveis extensões estão disponíveis. Há a necessidade da criação de tal conjunto de informações, pois apenas um subconjunto do padrão é implementado.

A digitalização dos objetos do mundo real é realizada através do emprego de modelos conhecidos como entidade-relação, sendo que todos os mapeamentos dessas relações são formalizados no documento de definição de objetos DICOM (NEMA, 2011c) e a abstração que contém as informações necessárias para descrever como foi encapsulado chama-se IOD (*Information Object Definition*).

Cada um dos IODs é constituído de elementos de dados (*data elements*) obrigatórios e opcionais que são identificados unicamente internamente no padrão em um dicionário de dados (NEMA, 2011f) e agrupados de acordo com similaridade. Dessa maneira, o identificar, chamado de *tag*, é composto pelo grupo (G) ao qual pertence e o elemento dentro desse grupo (E), resultando assim em um identificador seguindo o formato hexadecimal (GGGG, EEEE), por exemplo, o identificador (0010, 0010) é mapeado para o nome do paciente.

Figura 2 - Representação da estrutura *data element*



Fonte: (NEMA, 2011e)

A ordem da estrutura do *data element* em um IOD é visto na Figura 2 e, sem levar em consideração o conteúdo, dependendo do sintaxe de transferência e do tipo de dado varia de 8 a 12 *bytes*. Os dados presentes são o identificador único e o conteúdo, que pode ser textual ou binário. O valor de representação, ou seja, o tipo de dado (nome de pessoa, data, hora, código, idade, dentre outros) também pode estar presente, mas é condicionada em relação à sintaxe de transferência. As demais informações, que são obrigatoriedade, obrigatoriedade condicional e valor de multiplicidade de dados são obtidas no documento de dicionário de dados (NEMA, 2011f) e no documento de definição do objeto de informação (NEMA, 2011c).

Os IODs podem ser objetos de dois tipos: normalizado ou composto. Os objetos normalizados são elementos específicos do mundo real, isto é, pode existir um elemento que agrupe todas as informações referentes ao paciente, como nome, idade, sexo, altura peso, dentre outros dados e, de maneira similar, um elemento composto pelos atributos do estudo, como dados temporais sobre quando o estudo foi realizado, identificador do estudo. A combinação dos tipos de dados normalizados resulta nos IODs do tipo composto, que por sua vez são constituídos de todos os dados necessários para contextualização da informação de um exame. Cada IOD contêm um identificador único global de até 64 caracteres chamado de UID (*Unique Identifier*) em que o prefixo deste identificador deve estar registrado em alguma organização reguladora, como a ANSI nos Estados Unidos ou DIN na Alemanha. O UID é formalizado no documento que define as codificações e estruturas de dados (NEMA, 2011e).

Para ilustrar um IOD do tipo composto, pode-se pensar em um corte de uma tomografia computadorizada, que contém centenas de cortes. Toda informação pertencente ao corte é mapeado para um IOD. Assim, individualmente um IOD composto sempre é completo de forma que contenha informações redundantes de todos os dados sobre um exame. Por exemplo, dados do paciente, dado do estudo, dados da série e da instância em si. O conteúdo da instância é especifica da natureza do equipamento onde foi produzida e pode ser uma imagem, dados em formas de onda para eletrocardiogramas ou laudo ditado, pdf (Portable Data Format) ou ainda vídeos.

A transformação dos IODs em formato de arquivo é descrito no documento de mídias de armazenamento e formato de arquivos para compartilhamento de mídias (NEMA, 2011i). Além disso, os arquivos em conformidade com o padrão podem conter um cabeçalho com a

descrição de como interpretar a informação contida. Esse cabeçalho sempre está em *little endian explicity*.

Na computação *little endian ou big endian* é a forma de representação de dados. O modificador de dados *explicity* condiciona se o valor de representação dos *data elements* está presente ou não. O conteúdo do arquivo pode estar codificado de outras maneiras, como *little endian implicity* ou ainda *big endian explicity*, sendo que se for uma imagem, por exemplo, pode ser o formato original sem compressão, JPEG *Baseline* ou mesmo JPEG 2000, dentre outros formatos de imagens especificados pelo padrão DICOM.

O padrão DICOM, além de definir um formato de imagens médicas, também é formaliza um protoloco de comunicação que utiliza os padrões já existentes baseado no ISO-OSI (*International Standards Organization Open Systems Interconnection*). Especificamente no documento de troca de mensagens (NEMA, 2011g) é indicado como as classes de serviço devem ser definidas através de um protocolo chamado DIMSE (*DICOM Message Service Element*). Tal protocolo se situa na camada de aplicação que por sua vez encontra-se sobre a camada de transporte. O protocolo de transporte escolhido pelo comitê foi o TCP (*Transfer Control Protocol*), pois por sua definição habilita um canal de comunição confiável que assegura que os dados transferidos cheguem ao destino e também aproveita a interoperabilidade já existente entre dispositivos que se comunique em uma rede.

Os elementos que se comunicam através do protocolo DICOM são chamados de entidades de aplicação e no documento de descrição de classes de serviços (NEMA, 2011d) contém as guias de como devem ser as interações entre diferentes entidades. A entidade que provê os serviços tem como nomenclatura a sigla SCP (*Service Class Provider*) e a que requisita o serviço é chamado de SCU (*Service Class User*). A relação de fluxo normal entre duas entidades ocorre em três etapas, conforme ilustrado na Figura 3.

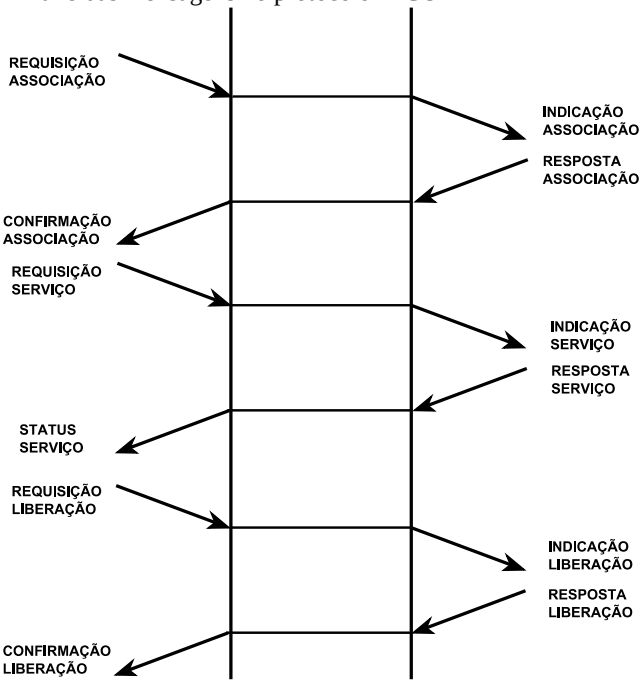
Todos os serviços primeiramente devem realizar uma associação a fim de verificar se as duas entidades estão registradas entre si. Nessa etapa também ocorre a negociação de quais operações podem ser executadas e quais extensões de serviço estão disponíveis, tal como busca de nomes de pessoas utilizando lógica nebulosa. Posteriores ao sucesso dessa fase são trocadas as mensagens referentes ao serviço a ser realizado e ao final é requerida a finalização de forma explícita da associação estabelecida previamente. Na máquina de estados do protocolo DICOM e obrigatoriamente em qualquer implementação estão

previstos os fluxos alternativos de interrupção e cancelamento em cada etapa.

Há dois tipos de serviços que podem ser realizados, serviços do tipo C e serviços do tipo N. Somente o primeiro tipo é abordado aqui, pois a realização de armazenamento, busca e recuperação é sobre os IODs do tipo composto. Dentre os serviços que podem ser oferecidos, três serão analisados neste trabalho em relação aos mecanismos utilizados na camada de persistência de dados:

- Armazenamento: **C-Store** (*Composite Store*)
- Busca: **C-Find** (*Composite Find*)
- Recuperação: **C-Get e C-Move** (*Composite Get/Move*)

Figura 3 - Fluxo das mensagens no protocolo DICOM



Fonte: (NEMA, 2011h)

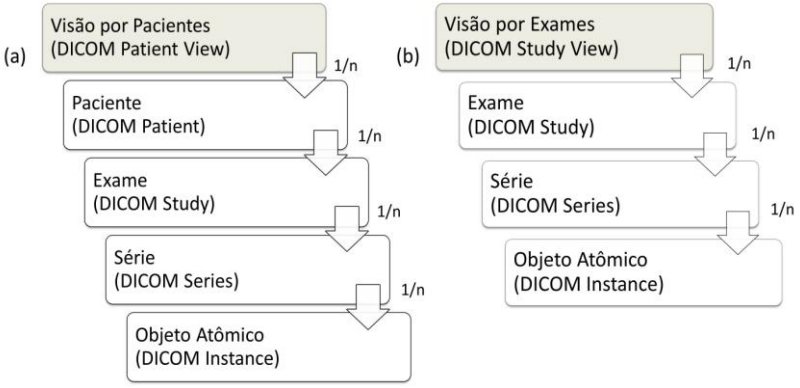
3.1.1 Classes de Serviço

Como o foco do padrão DICOM é a interoperabilidade, alguns tópicos não são formalizados, como a escolha da tecnologia em que os

exames devem ser armazenados, apenas é descrito o procedimento que o SCP e o SCU devem realizar. O fabricante do equipamento médico ou do PACS seleciona o modelo de persistência de dados que parecer ser a melhor solução.

As informações contidas em um objeto DICOM têm relação entre si de maneira hierárquica, um paciente tem um ou mais estudos, que por sua vez é composto por uma ou mais séries, que é o nível que contém informações da modalidades de um exame, e em cada série há as instâncias do exame realizado, como, por exemplo, a imagem em valores de densidade gerada pelo equipamento de ressonância magnética. Tendo isso como base é fundamentada no modelo de busca e recuperação DICOM (NEMA, 2011d) como as informações estão disponíveis: estudo ou paciente como raiz da busca, sendo que os dados referentes ao paciente migram para o nível do estudo quando este é a raiz. A raiz é o ponto de partida durante a operação de consulta padrão, tanto para os metadados DICOM, quanto para busca do UIDs para realizar a recuperação. A Figura 4a ilustra o primeiro modelo e em Figura 4b o segundo, e cada nível apresenta uma série de atributos que são compostos por uma chave única, um conjunto mínimo de chaves adicionais obrigatórias e também chaves opcionais.

Figura 4 - Os dois modelos de busca e recuperação de informação no padrão DICOM: Visão por Pacientes e Visão por Exames



Fonte: (NEMA, 2011d)

No modelo de busca e recuperação (NEMA, 2011d) é indicado que, para a manipulação da busca, o requisitante do serviço pode utilizar os seguintes tipos de operadores a fim de refinamento de resultados:

- **Valor único:** o valor retornado deve ser exatamente igual à chave passada, é utilizado em conjunto com outras chaves para restringir resultados;
- **Universal:** busca através de chave vazia para trazer todos os valores existentes que estejam associados a esta chave;
- ***Wildcard*:** busca coringa através de caracteres do tipo "*" e "?", ou seja, para todos os pacientes que começam com a letra M, a busca seria feita com M* e se apenas um caractere varia a passa-se como valor da chave de busca Mari?, sendo que "?" é substituído por qualquer caractere. É o único tipo de busca que o padrão impõe em ser sensível a caracteres minúsculos ou maiúsculos;
- **Lista de UID:** lista de chaves referentes às chaves únicas, por exemplo, trazer um conjunto de exames que atendam a um UID, pode ser utilizado em relação a trazer todos os exames realizados por algum paciente específico, já buscada através de uma operação com busca com chave única;
- **Busca por Intervalo:** para dados temporais, como intervalos de datas, horas ou datas-horas. Há abertura para estimar o quão preciso é a consulta, por exemplo, consulta simples da realização de um estudo em intervalo de hora, minuto ou mais refinada com um intervalo de hora, minuto, segundo e milissegundo com até seis casas decimais para representação da informação;
- **Busca por Item de Sequencia:** essa consulta é realizada sobre um tipo de *data element* específico que tem a propriedade de agrupar outros *data elements*, isto é, é uma estrutura que pode conter zero ou mais grupos de *data elements.* Por exemplo, no agrupamento passado como parâmetro da busca pode haver dados referentes a buscas temporais juntamente com valores únicos, tornando assim uma combinação de duas buscas diferentes. Não será abordado aqui.

A recuperação de objetos DICOM dá-se através de identificadores únicos (UID) do nível de busca requisitado. As operações sobre esse tipo de dado somente suporta operadores de busca

de valor único, universal e lista de UID. É natural que em um ambiente PACS se utilize primeiramente da operação C-Find para filtrar dados e obter os UIDs e assim executar a operação C-Move/C-Get.

No padrão, o comportamento da entidade que provê o serviço de busca/recuperação é descrito para cada uma das classes de serviços. Para cada um dos serviços há dois tipos de fluxos, um voltado para interoperabilidade entre os fabricantes e um estendido que suporta busca relacional. Vale ressaltar que a busca relacional descrita no padrão não obriga o uso de busca relacional formalizada através da álgebra relacional que é utilizada em implementações de banco de dados relacional. Trata-se apenas de uma busca que remove a restrição de se ter cada uma das chaves únicas dos níveis anteriores presente no nível que se deseja buscar.

3.1.1.1 C-Store

A SCP é responsável por executar o armazenamento do objeto DICOM de maneira transparente retornando para o requisitante a indicação do sucesso ou falha da operação. O objeto por sua vez deve ser disponibilizado para acesso em algum tipo de mídia por uma janela de tempo.

O serviço de armazenamento é realizado de forma assíncrona do ponto de vista do exame, isto é, não existe nenhum tipo de informação indicando que todos os cortes de uma tomografia já foram recebidos pelo SCP, cada IOD é transferido do SCU para o SCP individualmente.

3.1.1.2 C-Find

A SCP é responsável pela busca de dados relacionados à um exame. É possível realizar a busca a partir de qualquer um dos níveis de busca para obtenção da informação desejada. O requisitante do serviço deve fornecer a raiz da busca, o nível a ser pesquisado e um conjunto de chaves de busca e então o provedor é responsável por recuperar as informações. Um exemplo de busca é a requisição dos UID dos estudos de um determinado paciente em um intervalo de data, sendo utilizados os operadores de busca universal (UID dos estudos), busca por valor único (nome do paciente) e intervalo de data.

Em um modelo tendo o paciente como raiz da, por exemplo, se é desejada alguma chave do nível de série é necessário que durante a troca de mensagens do serviço C-Find seja fornecida a chave única de cada nível acima do nível requerido, ou seja, as chaves únicas do nível de

estudo e paciente. Esse é o comportamento padrão para interoperabilidade do serviço e é chamado de busca hierárquica, porém a fim de evitar múltiplas requisições de busca hierárquica, também existe uma extensão que é negociada durante a associação das entidades de aplicação e é chamado de modelo de busca relacional, que tem como característica a não obrigatoriedade de se ter a chave única dos níveis acima do nível pesquisado.

3.1.1.3 C-Move/C-Get

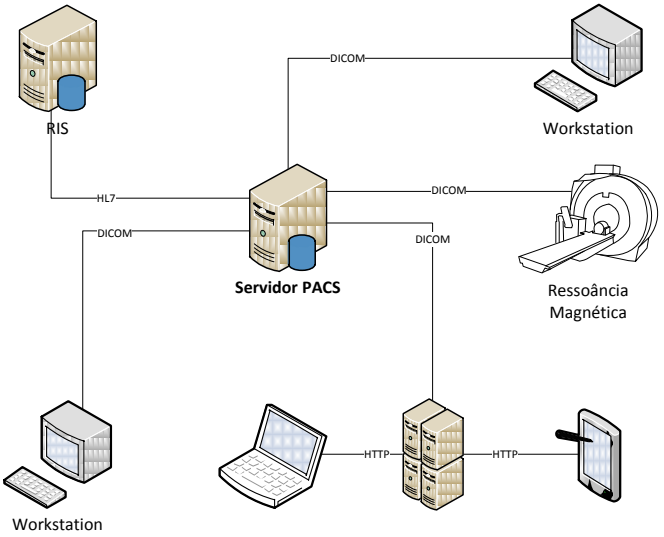
A SCP pode prover os dois serviços que são responsáveis pela recuperação de um objeto DICOM. A utilização dele dá-se no fornecimento UIDs do nível que se deseja recuperar e então todas as instâncias que se encontram abaixo desse nível são retornadas ao SCU.

A diferença entre a execução do comando C-Move e o C-Get é que o primeiro é requisitado para copiar um objeto de uma entidade de aplicação para uma terceira parte, requerendo duas conexões e associações diferentes para transmissão e o segundo comando realiza a transferência do objeto na mesma conexão da requisição. É justificado que o primeiro comando é mais seguro, pois a terceira parte para onde o objeto DICOM será transferido deve estar registrado no SCP que realizará o serviço, porém pode haver problemas de restrição de comunicação relacionados ao *firewall*. Qualquer tipo de falha que possa comprometer a segurança e privacidade das informações do paciente, durante a transmissão dos dados, pode ser mitigado através do uso de criptografia na camada de transporte, como aproveitamento da infraestrutura de chaves públicas e privadas no emprego do TLS (*Transport Layer Security*), tal conceito de segurança de informação foge do escopo deste trabalho.

3.2 SISTEMA DE COMUNICAÇÃO E ARQUIVAMENTO DE IMAGENS (PACS)

De acordo com (Huang, 2010) os sistemas de comunicação e arquivamento de imagens *PACS* são compostos por entidades que vão desde dispositivos para aquisição de dados e imagens médicas, até dispositivos de armazenamento e subsistemas integrados via rede para prover meios de auxílio para interpretação primária aos radiologistas, sendo que alguns dos elementos fundamentais pertencente ao PACS são ilustrados na Figura 5.

Figura 5 – Elementos presentes em um PACS e os meios empregados para interoperabilidade dos sistemas



O objetivo de se adotar uma política de integração dos sistemas digitais em hospitais ou clínicas é tornar o fluxo de trabalho em ambiente radiológico mais eficiente, uma vez que os exames estão disponíveis para acesso alguns segundos após serem enviados aos PACS e também é possível diminuir o custo operacional.

Os dados dos pacientes são providos para o PACS por outro sistema, geralmente através do padrão de interoperabilidade em tecnologia de informação em saúde HL7 (*Health Level 7)*, chamado de sistema de informação radiológica (*RIS – Radiology Information System*) que tem como objetivo oferecer suporte a agendamento de exames, rastreamento de filmes digitais e criação e armazenamento de laudos.

O RIS, também através do HL7, é integrado ao sistema de informação hospitalar (*HIS – Hospital Information System*) que é responsável pela administração hospitalar e gerencia recursos como folha de pagamento, controle de pacientes, avaliação e planejamento de custos. Além do HL7, para interoperabilidade do sistema é utilizado o protocolo DICOM para transferência de imagens entre os componentes do PACS, assim como são apoiados o uso tecnologias e padrões livres.

O fluxo da realização de um exame médico consiste no registro do paciente, no agendamento associado a uma ou mais modalidades. Em seguida, as informações sobre o procedimento a ser feito fica disponível no servidor do PACS e então diretamente no dispositivo que irá gerar as imagens médicas o executor do exame obtém os dados do agendamento através de um serviço do protocolo DICOM chamado *worklist*. Após a realização do exame, ele é disponibilizado para acesso por *workstations* ou dispositivos que sejam capazes de visualização e realização de diagnóstico. O laudo então é produzido, podendo ser através do formato de laudo estruturado DICOM SR (*Structured Report*) que naturalmente e referenciam os estudos, séries e instâncias que foram necessários para o diagnóstico e após sua conclusão é enviado ao PACS, novamente através do protocolo DICOM, para acesso e utilização futura.

O STT/SC adotou um modelo baseado em *web* como arquitetura do PACS, assim as estações de trabalho cliente têm plataformas independentes, podendo ser acessadas de quaisquer dispositivos com acesso à rede, inclusive os dispositivos móveis.

3.3 PERSISTÊNCIA DE DADOS

Uma vez que o padrão DICOM não estabelece regras sobre a tecnologia a ser usada para persistência de dados, nesta seção são apresentadas algumas tecnologias utilizadas para esse fim.

3.3.1 Sistema de Arquivos

Em (NEMA, 2011i) é formalizado o formato de arquivo binário que representa o objeto DICOM, assim como ele deve ser armazenado em alguns tipos de mídias. Por estarem no décimo documento do padrão, os arquivos em formato DICOM são chamados de DICOM *part 10*. Para mídias removíveis como CDs ou DVDs o padrão descreve um tipo de índice rudimentar hierárquico estático chamado de DICOMDIR, que é responsável por sumarizar a estrutura dos arquivos em *part 10*, que geralmente segue o modelo de hierarquia do próprio exame, isto é, organiza os diretórios em estudo, série e instância, e no diretório "folha" fica o arquivo DICOM *part 10*.

3.3.2 Sistema de Gerenciamento de Banco De Dados Relacional

Os Sistemas de Gerenciamento de Banco De Dados Relacional (SGBDR) originaram-se na década de 60 (Codd, 1970) e é a forma mais

comum para armazenamento de dados. É empregado amplamente há décadas nas mais diversas áreas e dispõe de múltiplas soluções, algumas *open source* como o PostgreSQL ou MySQL e outras corporativas como o Oracle, IBM DB2 ou Microsoft SQL Server.

A linguagem SQL é definida como o meio tradicional de manipulação de dados de um SGBDR. Usualmente, além do modelo de tabelas, é oferecida também a possibilidade do uso de *large objects*, que representa uma maneira genérica de se armazenar qualquer dado, sendo texto ou binário sem limitações ao tamanho.

3.3.2.1 SQL

O SQL é uma linguagem de programação que tem suas origens na álgebra e cálculo relacional, com o objetivo de gerenciar os dados de um SGBDR, padronizada pelas organizações ANSI (*American National Standards Institute*) e ISO e continuamente atualizada por um comitê técnico pertencente a ISO e IEC (*International Electrotechnical Commission*). Os requerimentos mínimos para conformidade, estruturas de dados e operações estão descritas formalmente nos documentos ISO/IEC 9075 (ISO/IEC, 2008c; b; a).

A linguagem SQL tem como estrutura básica uma sintaxe de declarações que permitem manipular e definir dados através de comandos de consulta e modificação de tabelas (*INSERT INTO*, *SELECT*, *UPDATE*, *DELETE*), de controle de transações (*COMMIT*, *ROLLBACK*) e manutenção de tabelas e banco de dados (*CREATE TABLE*, *ALTER TABLE*, *DROP TABLE*, *CREATE DATABASE*), visões (*CREATE VIEW*, *ALTER VIEW*, *DROP VIEW*) e permissões (*GRANT*, *REVOKE*). Além disso há possibilidades de filtragem com a clausula WHERE que pode ser combinada com outras clausulas com operadores de lógica booleana. Há extensão para agregação, ordenação, junção de tabelas, entre outras funcionalidades, como ações (*CASCADE*, *RESTRICT*, *NO ACTION*, *SET NULL*, *SET DEFAULT*) que alteram estados entre tabelas que estão relacionadas entre si.

3.3.2.2 PostgreSQL

A solução de SGDBR adotada para armazenamento no STT/SC é o PostgreSQL[4] e foi escolhido por ser um sistema maduro e robusto, com mais de 15 anos desde sua versão inicial, é *open source* e

[4] www.postgresql.org/

implementa a maior parte do padrão SQL (PostgreSQL, 2009a). É empregado em áreas de comércio eletrônico, educação, financeira, governamental, telecomunicação e outros (PostgreSQL, 2011). Possui uma interface de programação para diferentes linguagens e disponibiliza uma gama de extensões desenvolvidas por terceiros que possibilita abranger mais cenários de uso de acordo com a necessidade, como manipulação de arquivos em formato XML armazenados, funções de criptografia, adição de dicionário de sinônimos para recuperação de documentos, dentre outros.

O PostgreSQL está em conformidade com as propriedades de confiabilidade ACID (atomicidade, consistência, isolamento, durabilidade) que foi descrita em um trabalho sobre recuperação orientada à transação (Haerder e Reuter, 1983):

- Atomicidade: operações podem ser agrupadas em uma transação e caso uma delas falhe nenhuma modificação é feita no banco, mantendo-se assim no último estado de consistência, dessa maneira previne que atualizações no banco sejam parciais;
- Consistência: propriedade que garante que uma transação só pode ir para um estado válido de acordo com regras definidas;
- Isolamento: a ocorrência de uma transação não pode ser afetada por outras transações ocorrendo simultaneamente;
- Durabilidade: uma vez que a transação é completada, qualquer falha subsequente no banco de dados não pode afetar as informações que foram incluídas ou modificadas anteriormente.

3.3.2.2.1 Organização dos Dados

O PostgreSQL pode gerenciar um ou mais banco de dados criados, que por sua vez tem seus dados em tabelas. As tabelas são estruturas que agregam dados através de colunas, que identificam o tipo de dados e linhas que representam uma entrada na tabela (Douglas, 2005). O conjunto de entradas em uma tabela é agrupado em blocos de dados chamados páginas, que originalmente tem a capacidade de armazenar 8,192 *bytes* e tal propriedade é passível de modificação durante o processo de compilação do PostgreSQL.

As entradas no banco tem tamanhos variados e portanto não é possível prever quantas entradas estão contidas em uma única página. A ordem das entradas em uma página é de acordo com a ordem de

inserção. Desta forma, durante um processo de consulta é realizada a leitura das entradas sequencialmente.

Cada entrada de uma tabela é identificada por uma chave única e para acelerar o seu acesso são armazenadas na estrutura dados de árvore B. Tal estrutura garante a consulta, escrita e remoção de um nodo em tempo logarítmico, inclusive no pior caso. Além disso, as chaves únicas podem ser associadas a outras tabelas para criação de relação. A referência de uma chave única externa à tabela é chamada de chave estrangeira.

Os índices também são armazenados da mesma forma que as entradas das tabelas e por nomenclatura as páginas de um índice são chamadas de páginas *index* e das entradas de páginas *heap*. Caso o conteúdo de uma entrada ou índice seja maior que o tamanho máximo de uma página, a parte excedente será escrita em uma tabela TOAST (*The Oversized-Attribute Storage Technique*), que é uma extensão para a tabela original da entrada (PostgreSQL, 2009b).

Os *large object*, são segmentados usando a mesma estrutura de páginas, e são armazenados em uma tabela, pertencente ao grupo interno do PostgreSQL, chamada pg_largeobject, que tem como esquema as seguintes colunas:

- *loid*: identificador único do tipo *oid* (*object identifier*) associado ao *large object*;
- *pageno*: como as páginas tem tamanhos limitados, o dado binário ou textual é truncado em páginas numeradas sequencialmente a partir do número zero, essa coluna do tipo int4 (inteiro de 4 bytes) tem como objetivo indicar quais páginas pertencem a um *oid* específico;
- *data*: coluna do tipo *bytea* que contém o bloco do dado binário ou textual. A quantidade de informação armazenada neste tipo de dado é limitada ao tamanho da página dividido por 4, isto é, por padrão 2 *kilobytes*, uma vez que o tamanho da página é 8,192 *bytes*.

Diferente das tabelas, que por sua vez são acessadas e modificadas através da linguagem SQL, os *large objects* são criados, modificados e removidos por uma interface de programação em linguagem C. Também é necessário que seja usado em conjunto com as declarações transacionais SQL. Nas versões do PostgreSQL posteriores a 9.0 a tabela *pg_largeobject* não tem mais permissão pública para

leitura, as permissões de acesso são controladas por uma nova tabela chamada *pg_largeobject_metadata*.

Para acelerar o acesso às páginas, estas ficam em memória em uma região chamada de *buffer* compartilhado, que é uma área comum que contém páginas de um ou mais bancos de dados. Para ilustrar, o sistema de paginação é análogo aos blocos de dados de um processo que está executando em um sistema operacional, ou seja, são carregadas na memória de acesso rápido e quando não mais utilizadas são atualizados no sistema de arquivos.

3.3.2.2.2 Indexação

Normalmente, a indexação das chaves únicas é feita através da estrutura de árvore B, porém o PostgreSQL oferece a escolha de outros meios como o *bitmap* (Chan e Ioannidis, 1998), GiST (*Generalized Search Tree*, disponibiliza uma interface de programação que é a base para extensão de árvores balanceadas customizadas como árvore B+ ou R) (Kornacker *et al.*, 1997) ou gin (*Generalized Inverted Index*, um índice invertido que tem como estrutura o par chave e lista de identificadores das entradas no banco).

3.3.2.2.3 Gerenciamento e Manutenção

Além disso, o PostgreSQL oferece uma estrutura que utiliza as facilidades da linguagem SQL para gerenciamento de usuários, políticas de acesso para garantir segurança, facilidades de *backup* e restauração, e meios de realizar manutenção e obter dados métricas de performance de operações realizas em tabelas e índices a fim de identificar pontos de otimização nas *queries*.

3.3.3 Hierarchical Data Format

O HDF (*Hierarchical Data Format*) foi criado pelo NCSA (*National Center for Supercomputing Applications*) em conjunto com a NASA (*National Aeronautics and Space Administration*) e é aplicado em diversos campos como simulação computacional em *grids* (Shasharina *et al.*, 2007), computação paralela em ambientes de múltiplos computadores (Yu *et al.*, 2006), gerenciamento de dados geográficos (Lee e Spence, 2002), dentre outros.

O objetivo do HDF é prover armazenamento eficiente de grandes volumes de dados científicos expressos de maneira hierárquica para

acesso por longo período de tempo (Folk e Pourmal, 2010). As operações de criação, modificação dos dados por meio de uma API nativa em linguagem Fortran ou "C". Há duas versões, conceitualmente diferentes, que recebem atualizações regularmente, o HDF4, para compatibilidade de sistemas legados, e o HDF5, atual versão. O uso do HDF5 é encorajado para novos projetos uma vez que soluciona uma série de limitações do HDF4, principalmente o tamanho máximo de 2GB de um arquivo no formato HDF4 e aperfeiçoamento do código para ser mais eficiente, dentre outras características[5].

3.3.3.1.1 Estrutura de um arquivo em formato HDF5

O HDF5 possui uma estrutura simples, que é dividida em grupos, que contém um cabeçalho (constituído do nome do grupo e uma lista de atributos) e uma tabela de símbolos com todos os objetos pertencentes ao grupo, e *dataset*, que é estruturado em um cabeçalho com quatro elementos e um conjunto de dados, que pode ter representação multidimensional. A composição de um cabeçalho do *dataset* é dividida em:

- Nome: caracteres alfanuméricos para identificar o *dataset* ao acessa-lo;
- Tipo de dado: tipo atômico, em que fazem parte os tipos primitivos de dados (caracteres, inteiros, ponto flutuante), ou tipo composto, que encapsula tipos nativos;
- Espaço de dados: descreve a dimensionalidade, em tamanho fixo ou ilimitado, do *dataset.* Também expõe trechos do conjunto de dados que podem ser acessados parcialmente, a fim de tornar a leitura mais eficiente;
- *Layout* dos dados: contínuo, que é o armazenamento padrão dos dados, segmentado em blocos de tamanho fixo ou compacto (sendo que aqui o dado não é comprimido em algum tipo de codificação específica e sim armazenado diretamente no cabeçalho do *dataset* quando o tamanho dos dados for considerado pequeno).

Além dos dois elementos que formam o HDF5, há também atributos que podem estar associados e/ou compartilhados entre os

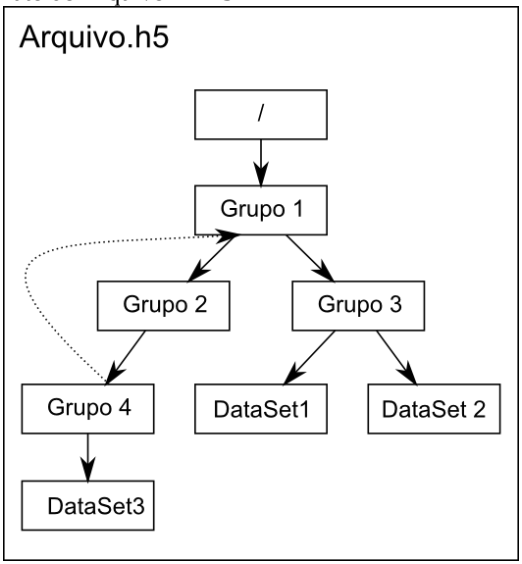
[5] http://www.hdfgroup.org/h5h4-diff.html

grupos ou *datasets*. Tais atributos tornam os dados armazenados auto descritivos e podem facilitar o acesso por longo período de tempo.

Adicionalmente, algumas características de criação e acesso dos grupos e *datasets* podem ser definidas através da manipulação de listas de propriedades, que são coleções de nome/valor que possibilitam alterar o tamanho alocado para os metadados, modificar a capacidade reservada para *buffers* durante operações de leitura, acessar aos dados através de uma camada virtual (acesso direto por rede, armazenamento somente na memória, acesso paralelo aos dados através de MPI (*Message Passing Interface*) ou ainda *drivers* customizados), modificação da codificação dos caracteres.

Para melhor entendimento de como é o formato HDF5, exemplificado na Figura 6, pode ser feito uma analogia com um sistema de arquivo do UNIX. Sempre há um diretório raiz que é identificado pelo caractere "/". Assim, um diretório é análogo a um grupo e o *dataset* é similar a um arquivo dentro de um grupo, ou seja, é o elemento que contém os dados. Um grupo pode conter outros grupos internamente, assim como um diretório pode conter outros diretórios.

Figura 6 - Formato de Arquivo HDF5



Fonte: http://www.hdfgroup.org/HDF5/doc/H5.intro.html

Assim como em um sistema de arquivos também é possível criar *symbolic links* (ligações simbólicas que se removidas não afetam os

dados, porém se os dados originais são removidos a ligação perde referência ao dado), representada como a linha tracejada na Figura 6 ou *hard links* (ligações diretas ao elemento, assim quando se remove um elemento é removida a ligação, se o total de referências ao objeto for nulo, então se perde a capacidade de acessá-lo; naturalmente quando um objeto é criado automaticamente é criado um *hard link*), representada pela linha direcional solida na Figura 6. Além disso, da mesma maneira que para acessar um arquivo dentro do sistema de arquivos é obrigatório fornecer sua localização também ocorre o mesmo comportamento no HDF5. Isto é, para criar, modificar ou acessar um grupo ou um *dataset* é necessário fornecer à API o caminho similar aos diretórios, delimitando os grupos com o caractere "/".

Algumas características interessantes para o armazenamento de objetos DICOM em HDF5, além da própria representação hierárquica natural de um exame, são:

- Acesso randômico - recuperação de dados específicos, como o pixel data e seus metadados para processamento de imagem, ou os campos de um laudo estruturado para construção de índices para busca semântica;
- Portabilidade e multiplataforma - arquivos HDF5 podem ser utilizados numa gama ampla de plataformas computacionais;
- Capacidade de lidar com grandes quantidades de dados - ideal para o volume atual de dados gerados pelos dispositivos integrados ao STT/SC;
- Disponibilidade dos dados para acesso por décadas, fundamental para recuperação do histórico de exames de um paciente;

3.3.3.1.2 Gerenciamento dos Dados

Internamente o formato de dados HDF5 utiliza uma implementação de árvore B para acelerar a recuperação dos dados, mas apesar disso não há meio de acesso externo para essa facilidade, sendo considerada uma desvantagem para os usuários que necessitam de algum meio de manipulação similar aos SGBDR. Como consequência é encontrada pesquisas para superar o problema de consultas e gerenciamento dos dados armazenado em formato HDF5 em (Nam e Sussman, 2003; Gosink *et al.*, 2006; PyTables, 2011), bem como tema deste presente trabalho.

Como já anunciado no Capítulo de TRABALHOS CORRELATOS para criar mecanismos de *queries* sobre os dados armazenados em HDF5 foi escolhido o uso de índices invertidos. A estratégia utilizando índices invertidos foi selecionada por permitir a indexação homogênea e extensível de todo tipo de metadado associado a objetos DICOM em qualquer nível da hierarquia.

3.3.3.1.3 Características de Consistência

Em relação às propriedades similares as ACID descritas anteriormente na seção sobre SGBDR, o HDF5 não possuí um modelo de transação para modificações de dados que garanta atomicidade das operações. O acesso por múltiplas *threads* de um processo não pode modificar os dados com garantia de consistência, inclusive acesso a grupos e *datasets* diferentes, uma vez que há estruturas globais que são alteradas durante o processo de atualização da informação. Tal conceito na computação é conhecido como *thread safety* e cabe ao utilizador da biblioteca HDF5 criar mecanismos para garantir tal propriedade. Ainda assim é possível habilitar uma primitiva de sincronização, que usa a abstração de semáforos na interface de programação, para permitir que múltiplos processos possam fazer leituras enquanto um único processo estiver modificando o arquivo em HDF5, porém tal funcionalidade deve ser selecionada durante o processo de construção da biblioteca HDF5 e é incompatível com a funcionalidade de paralelismo através de MPI.

Há projetos para versões futuras de inclusão de um sistema de registro de mudanças realizadas para recuperação somente dos metadados a fim de evitar corrupção do arquivo HDF5 em caso de ocorrência de falha inesperada, esse tipo de sistema é conhecido como *journaling* e está presente nos atuais sistemas de arquivos de diversos sistemas operacionais.

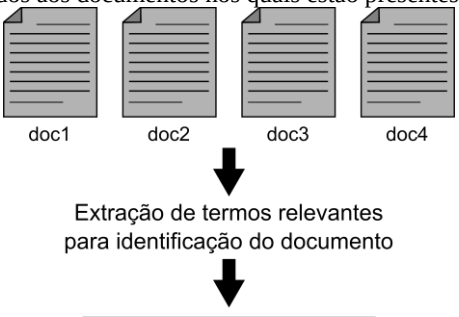
3.3.4 Apache Lucene

O Lucene é um projeto da Fundação Apache[6] que foi projetado para oferecer um núcleo de indexação de metadados e busca de documentos, através de uma API *open source* em Java, para a área de recuperação de informação. Por ser de propósito geral é empregado em plataformas de registro médico eletrônico, ambientes de desenvolvimento integrado (IDE), redes sociais e ferramentas de busca

[6] http://lucene.apache.org/java/docs/index.html

web, dentre outros (Nam e Sussman, 2003; Erik *et al.*, 2005; Lucene, 2011b). Internamente o Lucene utiliza índices invertidos em que os termos apontam para os documentos que estão relacionados como é ilustrado na Figura 7.

Figura 7 - Representação da indexação com a estrutura de índice invertido. Os termos ficam associados aos documentos nos quais estão presentes



ID	Termo	Documento
1	termo1	doc1, doc4
2	termo2	doc2,doc3
3	termo3	doc1, doc2, doc3
4	termo4	doc1, doc2
5	termo5	doc3,doc4
6	termo6	doc3
7	termo7	doc2

Usualmente, o processo de indexação na área de recuperação de informação segue alguns passos, sendo que primeiramente ocorre a aquisição dos termos necessários através de um coletor, como ocorre com *web crawlers* ou *web spiders* que desempenham esse papel para indexação a ser utilizada por motores de busca de páginas *web*. A partir dos termos coletados é necessária a construção de uma abstração do conteúdo a ser indexado chamada documento, que é constituído por unidades denominadas campos, que são elementos nominais associados aos termos. Dessa maneira, os documentos serão retornados de acordo com o pareamento dos termos requisitados durante a busca com os termos contidos no índice. Antes da indexação propriamente dita pode ocorrer a etapa de análise do documento que tem como função o aprimoramento da fase de busca através da separação dos termos em uma série de elementos atômicos chamados *tokens*. Por fim o documento é adicionado ao índice seguindo um formato próprio do Lucene.

Concebido para aproveitar a capacidade dos *hardwares* atuais, que suportam aplicações *multithread* para satisfazer requesitos de

desempenho, o Lucene foi projetado para assegurar a condição de *thread safety*. Dessa forma, na interface de programação, as classes responsáveis pela escrita e leitura do índice podem ser compartilhadas entre *threads* diferentes. Além disso, é *thread friendly*, pois oferece mecanismos para evitar condição de corrida, que é uma situação de disputa por recursos durante a execução de operações pela classe de escrita do índice. É importante ressaltar que similarmente ao PostgreSQL, o Lucene igualmente garante as propriedades transacionais ACID.

Nas próximas seções são vistos com mais detalhes a importância de cada um dos principais elementos utilizados durante o processo de indexação e busca.

3.3.4.1 Campos

O campo é o elemento que é ligado a cada valor a ser indexado, por exemplo, em um sistema de busca de livros podem ser extraídos os termos associados aos campos de título, autor, ano ou gênero. Durante a criação de um campo pode ser necessário ajustar certas propriedades que irão refletir no processo de busca:

- Analisado - quebra o conteúdo do campo em *tokens* de acordo com o analisador escolhido;
- Não analisado - o conteúdo do campo é tratado como um único *token*;
- Analisado sem normas - campo analisado que não armazena as normas, valor em ponto flutuante que representa a relevância de um documento, no índice;
- Não analisado sem normas - semelhante à propriedade acima em que o valor do campo não é analisado.

Entretanto, o valor do campo pode ser sinalizado para não ser passível de busca, isto é, outros termos são usados na operação de consulta e então o termo não indexado é extraído do documento recuperado.

A não análise do termo, ou seja, o armazenamento do conteúdo da informação na sua forma original como um único *token* e sujeito a recuperação somente é recomendada para termos considerados pequenos, todavia é possível habilitar uma opção nativa para compressão dos dados. A

Tabela 3 ilustra algumas combinações de opções de indexação e armazenamento de termos.

Tabela 3 - Exemplo de combinações de propriedades de um campo

Indexado	Armazenado	Exemplo de Uso
Não Analisado	Sim	Identificadores, Chaves Primárias, Números de Documento
Não Analisado	Não	Palavras Chaves Ocultas
Analisado	Sim	Títulos de Documentos, Sumarização de Documento.
Analisado	Não	Conteúdo de Documento
Não Indexado	Sim	Possíveis informações sobre o documento, como data de criação.

Fonte: (Erik *et al.*, 2005)

3.3.4.2 Analisadores

A função dos analisadores é processar o texto a ser indexado a fim de melhorar o resultado levando em consideração o ambiente onde operações de buscas serão realizadas. Para isso os analisadores percorrem os textos e os separam em unidades chamadas *token*. Nessa etapa, de acordo com a necessidade do usuário do Lucene, os *tokens* podem ser corrigidos caso haja erros ortográficos, injeção de sinônimos ao campo, a partir de dicionários associados ao Lucene, para aumentar a cobertura das buscas, formatação do conteúdo para caixa baixa, redução das palavras para raiz, ou seja, o elemento que expressa o significado da palavra, separação de *tokens* por espaço em branco ou ainda, baseado em uma gramática, reconhecer endereços de email, acrônimos e análises de palavras de língua inglesa para mapeamento para *tokens*. O Lucene oferece uma interface de programação extensível para criação de analisadores próprios quando nenhum dos nativos apresenta recuperação de resultados satisfatórios.

3.3.4.3 Buscas

Há duas formas de realizar buscas no Lucene, a primeira é com uma classe que analisa uma expressão geral de consulta e procura

transforma-la em uma busca na notação própria do Lucene. A segunda alternativa é explicitamente executar a consulta através de classes específicas:

- Termo: qualquer documento que contenha o termo busca estará na lista de resultados;
- Combinação de termos através de operadores de lógica booleana: obrigatoriedade de junções de termos com o operador AND ou da não obrigatoriedade com o operador OR ou exclusão com NOT, operadores de adição e subtração podem ser utilizados para termos mandatórios ou não;
- Intervalo de dados: os termos são armazenados no índice em ordem lexográfica. Os termos de início e fim podem estar incluídos ou excluídos no resultado respectivamente com colchetes ou chaves. O Lucene dispõe de facilidades para normalização dos tipos de dados temporais para que estejam em ordem e facilitar a operação de busca;
- Prefixo: retorna documentos que contém o início de acordo com o valor busca;
- Caracteres curinga: consulta mais geral semelhante à busca por prefixo que através do caractere "*" denota qualquer resultado para zero ou mais caracteres e o caractere "?" para zero ou um caractere. Pode ocorrer degradação no tempo do resultado se for utilizado poucos caracteres antes do caractere curinga. Não é recomendado o uso do caractere curinga como primeiro caractere da busca;
- Frase: combinado com o uso do analisador de espaço em branco, frases podem ser consultadas e os documentos retornados dependem da distância entre os *tokens* da consulta com os *tokens* resultantes durante a operação de análise no processo de indexação;
- Consulta nebulosa: retorna documentos que contém termos que possuem alguma similaridade com o termo buscado. A análise de similaridade é feita através da distância de Levenshtein, formalizada em (Levenshtein, 1966).

Para refinamento da consulta é possível também especificar em qual campo se encontra o valor dos documentos a serem recuperados. O resultado da busca é um vetor que contém os identificadores dos documentos retornados e uma pontuação referente ao nível de

relevância. Assim, ao se recuperar um elemento da lista, o documento é lido no índice e carregado na memória para manipulação.

3.3.4.4 Organização do índice

O índice é armazenado em sistema de arquivos como segmentos enumerados sequencialmente conforme os documentos são adicionados e para cada segmento há um único arquivo que contém metainformações, além de um arquivo que é registra todos os segmentos existentes. Cada segmento pode ser disposto num modelo chamado composto em que várias seções internas necessárias ao índice são representadas por: nome dos campos, termos, posição dos termos, frequência dos termos, vetor de termos, dentre outros tipos de dados relevantes. Outro arranjo é a distribuição de cada uma destas seções em um arquivo diferente. Dessa maneira, quanto mais segmentos existirem, mais descritores de arquivos são abertos pela aplicação do Lucene durante o processo de busca, podendo ser atingido um limite imposto pelo sistema operacional, situação que pode ser agrava caso os segmentos estejam dispostos de forma não composta.

O uso de segmentos foi escolhido para que a indexação fosse realizada de forma incremental, de forma assim que o segmento é salvo no sistema de arquivos ele fica disponível para consulta.

3.3.4.5 Manutenção dos Índices

Há duas formas de minimizar o problema de excesso de descritores de arquivos abertos, o primeiro é explicitamente invocando os métodos de otimização da API e o segundo é na mudança de parâmetros na classe que efetua a escrita do índice que controla a taxa de junções dos segmentos.

O processo de redução de segmentos por interface de programação é através da invocação da função de otimização, que tem como efeito colateral um possível aumento no uso de processamento de entrada e saída, assim como espaço em disco, e é apenas o processo de busca que obtém ganhos de desempenho e não causa nenhum efeito positivo ao processo de indexação.

Quatro métodos são oferecidos para realizar otimização: completa, parcial, que limita o número de arquivos segmentados resultantes, realização em *background* ou combinação da alternativa parcial com a operação realizada em *background*.

Não há ferramentas específicas para *backup* e restauração de um índice, sendo necessário, programaticamente com funções da API do Lucene, garantir consistência dos dados, ou seja, um congelamento nas classes de escrita para então realizar a cópia do estado atual do índice com subprocessos invocados explicitamente (como comandos de cópia do sistema operacional) que iterem sobre os arquivos do índice e realizem o *backup*. O processo de restauração consiste em simplesmente copiar o *backup* para o diretório de índices do Lucene.

3.3.4.6 Relevância de um Documento

A avaliação da relevância de um documento ocorre durante o processo de busca, e é escolhida de acordo com dois modelos, um booleano, que simplesmente verifica se o termo do documento encontrado é igual ao requisitado e um modelo conhecido como *Vector Space Model* (VSM) (Salton *et al.*, 1975), que transforma os termos dos documentos e os termos da busca em vetores e então calcula-se a distância entre os dois vetores. Desta forma, quanto menor a distância entre os vetores, maior é o grau de similaridade entre os termos buscados e os termos contidos em um documento. Esse segundo modelo é por padrão implementado no Lucene e requer que o número de ocorrências de cada termo seja registrado, bem como as posições onde o termo está presente. Durante a construção do campo é possível ajustar propriedades do vetor de termos para alterar a relevância.

O Lucene oferece interfaces para alteração direta da relevância do documento ou de um campo através do aumentando o valor de uma variável de ponto flutuante.

4. MÉTODO

Nesta seção são descritos os mecanismos para o método com SGBDR, já utilizado pelo STT/SC, e como foram implementados o método proposto, que visa o gerenciamento com Lucene de grande volume de dados em HDF5 de forma eficiente. Ambos utilizam ferramentas *open source* e foram implementados através das APIs em C e C++ das bibliotecas disponibilizadas e em ambiente GNU/Linux.

Os dados de cada nível a serem extraídos do exame em formato digital e sujeito às operações de busca, tanto pelo PostgreSQL e Lucene, seguindo o modelo de busca e recuperação DICOM estão descritos na Tabela 4. O tipo de dado do *data element* reflete no tipo de coluna no SGBDR, já no Lucene todos os dados são tratados como cadeia de caracteres.

Tabela 4 - Atributos de cada nível do modelo de busca e recuperação, com o estudo como raiz, empregadas nesta validação do trabalho

Nível	*Tag*	*Nome*	*Tipo de Dado*	*Tipo de Campo*
	0x0020, 0x000d	StudyInstanceUID	Identificador Único	Identificador Único Universal
	0x0010, 0x0010	PatientsName	Obrigatória	Nome de pessoa[7]
	0x0010,0x0020	Patient's ID	Obrigatória	
	0x0010, 0x0030	PatientsBirthDate	Opcional	Data
	0x0010, 0x0032	PatientsBirthTime	Opcional	Hora
Estudo	0x0010, 0x0040	PatientsSex	Opcional	Código
	0x0008, 0x0020	StudyDate	Obrigatória	Data
	0x0008, 0x0030	StudyTime	Obrigatória	Hora
	0x0008, 0x0050	AccessionNumber	Obrigatória	Palavra de até 16 caracteres
	0x0020, 0x0010	StudyID	Obrigatória	Palavra de até 16 caracteres
Série	0x0020, 0x000e	SeriesInstanceUID	Único	Identificador Único Universal

[7] O padrão DICOM descreve um tipo de dado destinado a tratar apenas variáveis cujo conteúdo é o nome pessoal e permite segmentar o conteúdo em primeiro nome, sobrenomes, assim como formas de tratamento.

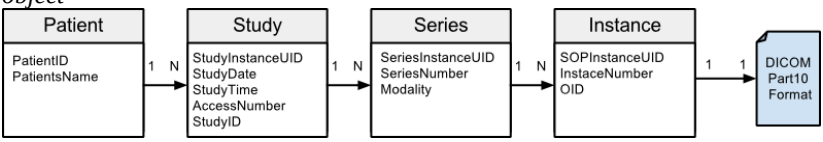
Nível	*Tag*	*Nome*	*Tipo de Dado*	*Tipo de Campo*
	0x0008, 0x0060	Modality	Obrigatória	Código
	0x0020,0x0011	SeriesNumber	Obrigatória	Até 16 Caracteres Numéricos
	0x0008, 0x0018	SOPInstanceUID	Único	Identificador Único Universal
Instância	0x0020, 0x0013	InstanceNumber	Obrigatória	Até 16 Caracteres Numéricos

4.1 SGBDR

Nesse trabalho, o banco de dados utilizado é o PostgreSQL versão 8.4, que é a versão estável da distribuição GNU/Linux Ubuntu Server 10.10 usado no ambiente de arquivamento. As tabelas, colunas e tipos de dados foram criados a partir do mapeamento do modelo de entidade-relacionamento fundamentado no documento de especificação dos serviços (NEMA, 2011d) nas seções destinadas a descrever o modelo de busca e recuperação. Foi mantido a configuração original do tamanho de página de 8,192 *bytes* e o tamanho do *buffer* compartilhado para acelerar o acesso às páginas foi fixado em 24 *megabytes*.

Os modelos de busca e recuperação DICOM, com os níveis de estudo e paciente na raiz da busca, são suportados pelo SCP, em que os atributos do paciente estão replicados no nível estudo. Cada um dos quatro níveis é representado por uma tabela, e os identificadores únicos de cada nível são indicados com a restrição de chaves-primárias, portanto não podem ter valores nulos, e chave estrangeira para o nível seguinte. Além da representação dos *data elements* por colunas de acordo com o tipo de dado, formalizado na descrição de codificação e estrutura do *data element* (NEMA, 2011e), o último nível contém um campo adicional de identificação do objeto binário DICOM, chamado *oid* (*Object Identifier*). O *oid* é referencia um *binary large object* (*BLOb*), que é uma facilidade, para armazenamento de arquivos binários, que não restringe o tamanho dos dados da mesma maneira que os campos nativos oferecidos pelo banco de dados e são adequados para arquivamento de formatos digitais textuais ou binários, como áudio, vídeo e, portanto, o arquivo do exame original em formato DICOM *part10*. O mapeamento entidade-relação dos níveis do DICOM para tabelas no banco de dados é visto na Figura 8.

Figura 8 - Mapeamento do modelo de busca e recuperação para tabelas e *large object*



4.1.1 C-Store

O exame recebido pela entidade SCP de armazenamento extraí todos os dados referentes às tabelas e os armazena através dos comandos SQL: *select*, *insert into* e *update*. Cada entrada na tabela tem o identificador do nível associado à chave única. Exceto pela tabela *Patients*, que tem como chave primária um identificador, do tipo de representação *Code String*[8], não garantido como único globalmente, as demais tabelas empregam os valores dos *data elements* com valores de representação UID.

Durante o processo de inserção das entradas em cada tabela, usa-se a chave única como confirmação se um nível já está presente, uma vez que pela natureza assíncrona na transmissão de um estudo completo constituído de IODs do tipo composto, um nível mais acima já pode ter sido registrado anteriormente. A chave única da entrada inserida é usada para criação da relação de integridade referencial, como chaves estrangeiras, entre os níveis do modelo de busca e recuperação. Entretanto, as relações entre os níveis não afetadas por nenhum dos cinco comandos de ações referenciais.

Após a inserção do último nível, isto é, o nível de instância, o conteúdo binário do exame em formato DICOM é armazenado, através da API em linguagem C do PostgreSQL, como *BLOb* e o *oid* resultante do sucesso da operação é atualizado na tabela. O algoritmo base está sumarizado na Figura 9.

[8] código alfanumérico de 16 caracteres no máximo para identificação do paciente localmente e gerado de acordo com regras da instituição onde o exame é produzido

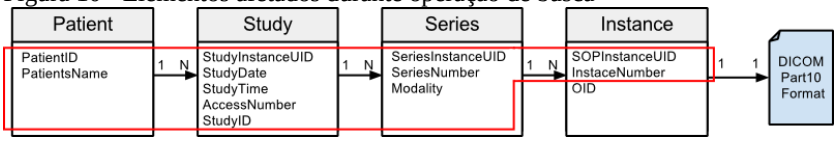
Figura 9 - Algoritmo para armazenamento de um objeto DICOM

```
ResultadoDaConsulta = SELECT "IDDoPaciente" FROM "pacientes" WHERE "IDDoPaciente" = 'valorDoIDDoPaciente'
se ResultadoDaConsulta ESTÁ VAZIO
    INSERT INTO "pacientes" VALUES ('valorDoIDDoPaciente'
ResultadoDaConsulta = SELECT "UIDDoEstudo" FROM "estudos"
                      WHERE "IDDoPaciente" = 'valorDoIDDoPaciente' AND "UIDDoEstudo"='valorDoUIDDoEstudo'
se ResultadoDaConsulta ESTÁ VAZIO
    INSERT INTO "estudos" VALUES ('valorDoIDDoPaciente','valorDoUIDDoEstudo')
ResultadoDaConsulta = SELECT "UIDDaSerie" FROM "series" WHERE "UIDDaSerie'" = 'valorDoUIDDaSerie'
se ResultadoDaConsulta ESTÁ VAZIO
    INSERT INTO "series" VALUES ('valorDoUIDDoEstudo','valorDoUIDDaSerie')
ResultadoDaConsulta = SELECT "UIDDaInstancia" FROM "imagens" WHERE "UIDDaInstancia" = 'valorDoUIDDaInstancia'
se ResultadoDaConsulta ESTÁ VAZIO
    INSERT INTO "imagens" VALUES ('valorDoUIDDaSerie','valorDoUIDDaInstancia')
    INSERT INTO "imagenspart10" VALUES ('valorDoUIDDaInstancia','IDDoPaciente','valorDoUIDDoEstudo','valorDoUIDDaSerie')
ResultadoDaConsulta = SELECT "oiddaimagem" FROM "imagenspart10" WHERE "UIDDaInstancia" = 'valorDoUIDDaInstancia'
se ResultadoDaConsulta ESTÁ VAZIO
   bloboid = InserirBinarioDoArquivoDICOMNaBaseDeDados()
   UPDATE "imagenspart10" SET "oiddaimagem" = bloboid WHERE "UIDDaInstancia" = 'valorDoUIDDaInstancia'
```

4.1.2 C-Find

O serviço *C-Find* requer o acesso sobre as colunas de todas as tabelas, exceto o *oid* da tabela *Instance*, como é realçado na Figura 10. O modelo de busca relacional é descrito no documento de especificação das classes de serviços (NEMA, 2011d). O serviço requisitante deve indicar em qual nível se encontra a informação e uma ou mais chaves que devem ser retornadas, podendo ser utilizadas chaves com valores para restringir resultados, em que irão se encontrar na clausula *WHERE* da declaração SQL.

Figura 10 - Elementos afetados durante operação de busca



A lista de *data elements* requisitados são mapeados para *query* no formato SQL, sendo a tabela para nível e as chaves para suas respectivas colunas. Tal operação tem como base o algoritmo contido na Figura 11.

Especificamente a busca por valor único de UIDs ou busca por lista de UIDs acessa o índice de árvore B do PostgreSQL, uma vez que são as chaves primárias das tabelas, já os demais tipos de dados são acessados de forma sequencial.

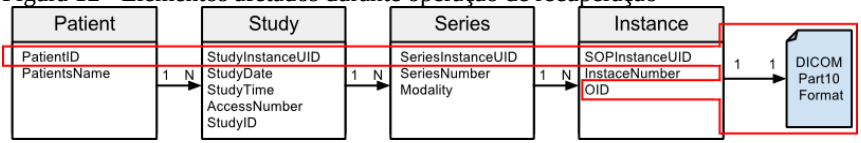
Figura 11 - Algoritmo para consulta de acordo com o nível

```
ResultadosDaConsulta = SELECT "chave0" [,...,"chaveN"] FROM "TabelaReferenteAoNivel"
                       WHERE "chave0" = 'valorDaChave0' [...AND "chaveN" = 'ValorDaChaveN']
ParaCada ResultadosDaConsulta
        RespostaAoRequisitante = ConstrutorDaResposta(ResultadosDaConsulta["chave0"],...,
                                 ResultadosDaConsulta["chaveN"])
        EnviaRespostaAoRequisitante(RespostaAoRequisistante)
```

4.1.3 C-Move e C-Get

Como pode ser visto na Figura 12, além das colunas das tabelas referentes às chaves primárias, há acesso também ao *oid* para obtenção do *BLOb* que contém o arquivo binário em formato DICOM *part 10*. Portanto, durante a operação de obtenção do *oid*, são criadas *queries* na sintaxe da linguagem SQL, que apenas contém chaves primárias do banco, da mesma forma que está ilustrada na Figura 11. Assim, com o identificador do objeto em *BLOb*, é possível obter, por chamadas da interface de programação do PostgreSQL, o arquivo DICOM com o conteúdo original. Em seguida o objeto DICOM é analisado e transformado, de acordo com a sintaxe de transferência negociada no processo de associação entre as entidades SCP e SCU, na estrutura IOD que é transmitida como resposta à requisição do cliente.

Figura 12 - Elementos afetados durante operação de recuperação



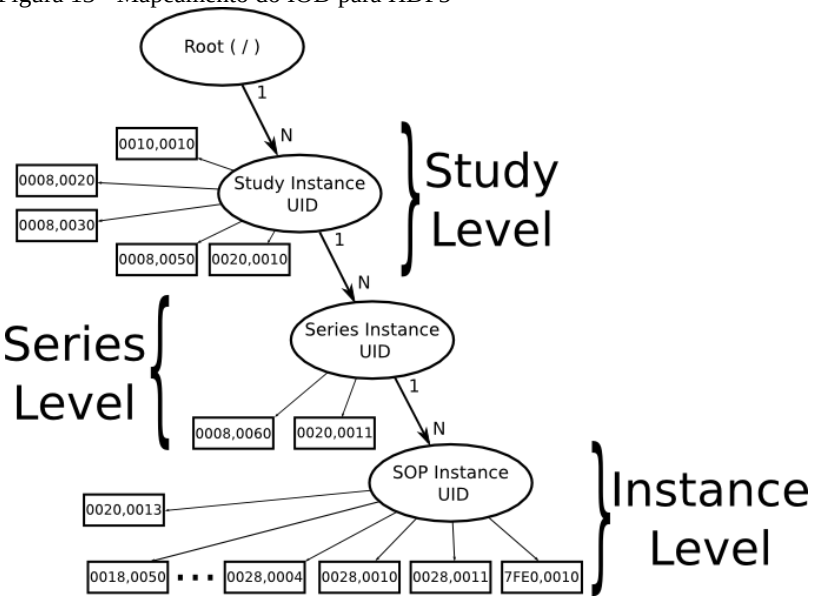
4.2 HDF5 com Lucene

O método de arquivamento descrito aqui se iniciou a partir do trabalho de conclusão de mestrado de (De Macedo *et al.*, 2009) e o código que realizada o serviço foi reescrito da linguagem C++ para C para possibilitar o uso futuro do MPI. A necessidade de portabilidade do código foi necessária pois a escolha de uma configuração da biblioteca HDF5 com suporte à processamento paralelo é conflita com a disponibilização de interfaces em alto nível através da linguagem C++. Os aspectos da biblioteca HDF5 devem ser selecionados no momento de configuração antes do processo de compilação.

O formato HDF5 permite expressar a relação entre os níveis internos dos objetos DICOM hierarquicamente. Cada nível é

representado por um grupo, em que seu valor é o conteúdo do UID. Os *datasets* contidos nos grupos são a representação dos *data elements* de cada nível. A disposição da hierarquia dos dados presentes no IOD é traduzida para a representação do formato HDF5, que é listrada na Figura 13. Como é visto, optou-se pela utilização somente do modelo de estudo como raiz do nível de busca e recuperação, de modo que a justificativa para tal decisão é baseada na premissa de que os grupos do HDF5 devem ter nomes únicos para identificação global e isso é alcançado com o tipo de dado UID de cada nível. O elemento de dado *patientID* usado no modelo SGDBR não atende a esse requisito pois é do tipo *Code String*.

Figura 13 - Mapeamento do IOD para HDF5



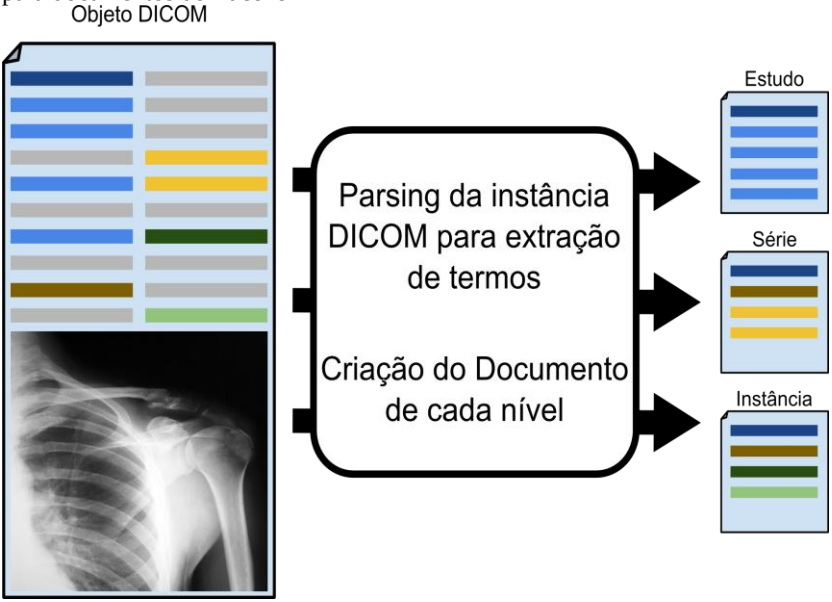
As elipses representam os grupos que contém um ou mais grupos do próximo nível e zero ou mais *datasets*, que por sua vez estão na forma de retângulos. Logo, abaixo do grupo raiz (“/”), há o nível de estudo, que além de conter os metadados sobre o exame e o paciente, contém os grupos das séries pertencentes ao exame. No último nível, ligado à série, há uma instância que é composta pelos metadados.

O mecanismo de busca de dados é realizado pelo *Lucene*, em que os termos que apontam para o documento são os *data elements*. Os índices

contêm três tipos diferentes de documentos, dos quais cada um deles representa os níveis estabelecidos no modelo de busca e recuperação do padrão DICOM. Cada qual contém os identificares únicos dos níveis acima, assim como as chaves obrigatórias do nível. Portanto, como é ilustrado na Figura 14, a abstração do nível estudo contem o conteúdo do *data element* nominado *study instance UID*. O próximo nível série contém os valores dos *data elements study instance UID* e *series instance UID* e, por fim o documento do nível de instância contém o identificador *sop instance UID*, assim como os demais identificadores do nível acima.

Os campos que são destinados aos valores dos *data elements* tem como modificadores *stored*, *indexed* e não são *tokenized*, ou seja, a combinação destas três características faz com que conteúdo original seja passível de busca e recuperação.

Figura 14 - Mapeamento dos níveis do modelo de busca e recuperação DICOM para documentos do Lucene



4.2.1 C-Store

O exame transmitido para o SCP é analisado e os *data elements* são mapeados para o formato HDF5 seguindo uma estrutura contida em um arquivo descritor, visto na Figura 15, em formato XML e carregado durante o início da execução da entidade de aplicação. O conteúdo deste arquivo também tem a função de identificar quais informações de cada nível vão ser indexadas. Os *data elements* que não estiverem associados a nenhum dos níveis no arquivo descritor sao automaticamente inseridos no último grupo da instância.

Figura 15 - Arquivo descritor do HDF5 e Lucene

```
<?xml version="1.0" ?>
<root>
  <!-- Study Level  -->
  <group qrlevel="STUDY" dataElementTag="0x0020,0x000d" > <!-- StudyInstanceUID - U - UI -->
    <dataset dataElementTag="0x0010,0x0010"  /> <!-- PatientsName - R - PN-->
    <dataset dataElementTag="0x0010,0x0030"  /> <!-- PatientsBirthDate - O - DA -->
    <dataset dataElementTag="0x0010,0x0032"  /> <!-- PatientsBirthTime - O - TM -->
    <dataset dataElementTag="0x0010,0x0040"  /> <!-- PatientsSex - O - CS -->
    <dataset dataElementTag="0x0008,0x0020"  /> <!-- StudyDate - R - DA -->
    <dataset dataElementTag="0x0008,0x0030"  /> <!-- StudyTime - R - TM -->
    <dataset dataElementTag="0x0008,0x0050"  /> <!-- AccessionNumber - R - SH -->
    <dataset dataElementTag="0x0020,0x0010"  /> <!-- StudyID - R - SH -->
    <dataset dataElementTag="0x0008,0x0061"  /> <!-- ModalitiesinStudy - O - CS -->
    <dataset dataElementTag="0x0008,0x0062"  /> <!-- SOPClassesinStudy - O - UI -->
    <dataset dataElementTag="0x0008,0x0090"  /> <!-- ReferringPhysiciansName - O - PN -->
    <dataset dataElementTag="0x0008,0x1030"  /> <!-- StudyDescription - O - LO -->
    <dataset dataElementTag="0x0008,0x1060"  /> <!-- NameofPhysician(s)ReadingStudy - O - PN -->
    <dataset dataElementTag="0x0008,0x1080"  /> <!-- AdmittingDiagnosesDescription - O - LO -->
    <dataset dataElementTag="0x0010,0x1010"  /> <!-- PatientsAge - O - AS -->
    <dataset dataElementTag="0x0010,0x1020"  /> <!-- PatientsSize - O - DS -->
    <dataset dataElementTag="0x0010,0x1030"  /> <!-- PatientsWeight - O - DS -->
  </group>

  <!-- Series Level  -->
  <group qrlevel="SERIES" dataElementTag="0x0020,0x000e" > <!-- SeriesInstanceUID - U - UI -->
    <dataset dataElementTag="0x0008,0x0060"  /> <!-- Modality - R - CS -->
    <dataset dataElementTag="0x0020,0x0011"  /> <!-- SeriesNumber - R - IS-->
  </group>

  <!-- Image Level  -->
  <group qrlevel="IMAGE" dataElementTag="0x0008,0x0018"  > <!-- SOPInstanceUID - U - UI -->
    <dataset dataElementTag="0x0020,0x0013"  /> <!-- InstanceNumber - R - IS -->
  </group>
</root>
```

As chamadas às funções da API do HDF5 de leitura ou escrita sempre retornam um valor positivo inteiro ou -1, o primeiro para indicar sucesso e é usado como identificador local do objeto durante a manipulação do elemento (grupo ou *dataset*) e o segundo acusa que ocorreu uma falha, como a inexistência do dado a ser acessado.

A operação de armazenamento consiste na tentativa de acesso ao grupo correspondente ao nível da informação de acordo com o modelo de busca e recuperação. Caso o retorno da função da interface de programação indique falha, então é inferido que o grupo deve ser criado e, similarmente, os *data elements* deste grupo são inseridos da mesma maneira, a *tag* é utilizada para tentar acessá-lo, em caso de falha um novo *data*set é criado e associado ao grupo corrente.

Junto com essa etapa é feita a criação dos índices usando chamadas às funções da API do CLucene, sendo que primeiramente é feita uma busca pelo identificador único do nível atual para cada tipo de documento, do nível correspondente, inserido com o intuito de verificar se os dados correntes já foram indexados, isto é, é executada uma consulta por termo com o valor do UID. Se é constatado que há necessidade de indexação é então criado um documento correspondente ao nível e os campos são nomeados de acordo com a *tag* do *data element* e os valores são processados pelo analisador padrão do Lucene e inseridos no documento, que por fim é adicionado ao índice.

Todos os valores do campo têm as características de armazenamento para possibilitar sua recuperação, e, exceto pelo *data element* que contenha a informação do tipo "nome de pessoa", o valor em si é interpretado como um único *token* e não há tipo algum de manipulação do campo ou documento para alterar a sua relevância.

Ao final da etapa de inserir os documentos é chamado a função de otimização para agrupar os segmentos de dados do Lucene para, possivelmente, acelerar as consultas e diminuir o número de arquivos do índice carregados pelo sistema.

4.2.2 C-Find

A entidade provedora do serviço *C-Find* realiza busca através da leitura dos índices previamente armazenados no formato Lucene durante a operação de *C-Store*. São retornados todos os conjuntos de atributos que pertencem ao nível requerido e cada um dos cinco tipos de busca são descritos em detalhes a seguir.

A consulta no Lucene é expressada por '*+GGGGEEEE: valor*', em que o símbolo de '+' denota que a presença do termo é obrigatória. Esta notação do nome do campo é uma forma simplificada das *tags* dos *data elements*, que originalmente são nominadas como 0xGGGG,0xEEEE ou GGGG,EEEE.

4.2.2.1 Operador Universal

Diferente do uso da linguagem SQL dos SGDBRs, o Lucene não apresenta bom resultado para buscas semelhantes à uma *query* que selecione todos os valores de uma coluna específica, por exemplo, para ilustrar a busca de todos os nomes de todos os pacientes em SQL seria 'SELECT patientsName from patients'. Tal consulta se fosse realizada de forma análoga no Lucene seria com um parâmetro de busca '+00100010:*', em que o símbolo '+' denota obrigatoriedade do campo, '00100010' o identificador do elemento de dado nome de paciente e o '*' qualquer valor para este campo.

Dessa maneira, a orientação encontrada para tal situação é a recuperação de todos os termos, na seção destinada a eles presente nos segmentos do índice, e posteriormente a filtragem pelo termo de acordo com o campo requerido.

4.2.2.2 Wildcard

Embutido no Lucene há suporte para buscas que contenham caracteres coringas, também conhecidos como *wildcards*, com operadores "*" e "?". Similar às expressões regulares o uso do primeiro implica na recuperação de 0 ou ocorrências na posição em que o operador se encontra, como por exemplo, a busca por '*+00100010:Am**' retorna todos os termos que contenham o prefixo Am, como Amélia, Amanda, etc. O operador "?" tem um mecanismo similar, a diferença é que substitui apenas uma ocorrência, como em '*+00100010:Marc?s*' que retorna documentos que contenham os termos como Marcus e Marcos. O uso do *wildcard* como primeiro caractere é desabilitado por padrão porque torna a consulta uma operação extremamente cara já que a busca por força bruta de todos os *tokens* no índice. A classe responsável por esse tipo de consulta é chamada *WildCardQuery*.

4.2.2.3 Valor Único e Lista de UID

A busca por valor único utiliza a classe mais comum que é a *TermQuery*. Um termo é associado à consulta e todos os documentos que contenham tal termo são recuperados. Similarmente a busca por lista de UIDs é realizada através da combinação de *TermQuery* de forma booleana e associada à classe *BooleanQuery*. Para ilustrar, suponha que é necessário a busca de três séries diferentes que possuam como UID,

id1, id2 e id3 respectivamente, a *query* resultante será '*+(0020000e:id1 0020000e:id2 0020000e:id3)*', em que o separador de espaço tem a semântica do operador booleano *OR*.

4.2.2.4 Intervalo de dados

De acordo com o padrão DICOM, há três tipos de consultas para intervalo temporal:

Data1-Data2 / Hora1-Hora2 – Intervalo explícito;
Data1- /Hora1 – Intervalo de um tempo inicial até data corrente;
-Data2/-Hora2 – Intervalo de uma data final até data do primeiro exame armazenado.

O Lucene suporta busca para intervalo de qualquer tipo de dado ordenado. Para isso, basta que a consulta seja expressada pela notação do tipo +campo:[início-fim]. Dessa maneira a consulta por intervalo temporal no Lucene é similar a '+00080020: [Data1-Data2], '+00080020: [Data1-*] ou '+00080020: [*-Data2].

4.2.3 C-Move e C-Get

A operação de recuperar os arquivos DICOM requisitados pelo cliente dá-se em duas etapas. A primeira é a busca das chaves únicas no índice invertido a partir da chave única fornecida do nível requisitado, ou seja, consulta por *TermQuery* ou combinação com *BooleanQuery*.

Um exemplo de recuperação, é o fornecimento da chave única de um estudo, sendo então necessário obter todas as chaves de todas as séries e de todas as instâncias, e em seguida, a partir das chaves obtidas, é iniciada a recuperação do DICOM através da API do HDF5. O acesso aos grupos para recuperação dá-se através de acesso pelo caminho, como em um sistema de arquivos. Para ilustrar, através do CLucene recuperam-se as triplas do estudo, série e instância e se acessa o pixel Data (07FE, 0010), por exemplo, através do caminho/estudo/série/instância/07FE0010.

Com todos os UIDs necessários, recursivamente, os metadados e o conteúdo principal de cada instância (imagem, vídeo, PDF, etc) é recuperado e o IOD é reconstruído sob demanda. Quando um IOD tem toda a informação original completa, então é transmitido do SCP para o SCU indicando o número de respostas pendentes.

5. RESULTADOS

Os experimentos têm como objetivo cobrir e avaliar o comportamento de todas as operações que a entidade SCP pode oferecer em relação a armazenamento, busca e recuperação. O ambiente de teste utilizado foi composto de um servidor para atender as requisições do cliente com configuração disponível 1GB de RAM, processador Intel de 1.6 GHz e sistema operacional GNU/Linux Ubuntu Server versão 10.10. Em todos os cenários apenas um SCU foi associado à um SCP, ou seja, todo processamento de dado foi feito de forma sequencial, ambientes com múltiplos clientes serão alvo de avaliações futuras. Foram descartadas as latências de rede, uma vez que para ambos os modelos foram avaliados em ambiente local. Todos os testes foram executados em 25 iterações para garantir significância estatística dos resultados e as versões das ferramentas foram:

- PostgreSQL - Versão 8.4;
- HDF5 - Versão 1.8.8;
- CLucene - Escrito em linguagem C a partir da versão do Lucene 2.4.0.

Os tempos mensurados na utilização do modelo empregado pelo PostgreSQL foram obtidos através da execução de *queries* durante os experimentos tendo como prefixo da consulta o comando *EXPLAIN ANALYZE*, portanto, para análise de uma *query* similar à "SELECT coluna FROM tabela" executa-se "*EXPLAIN ANALYZE* SELECT coluna FROM tabela". Como resultado da consulta é apresentado o plano de execução da *query* e o tipo de acesso feito aos dados (índice, sequencial, etc.) juntamente com o tempo de execução em milissegundos. Esse tipo de comando é utilizado durante a fase de análise das *queries* a fim de verificar a necessidade de otimizações (Douglas, 2005).

O as bibliotecas HDF5 e CLucene não dispõe de uma ferramenta de gerenciamento, portanto a coleta de tempo foi através da diferença do tempo inicial da execução do comando (de armazenamento, busca ou recuperação) com o tempo final. Para apresentar as mensurações com obtidos através da função dos sistemas *UNIX-like gettimeofday*, em que o tempo retornado é expresso de duas maneiras, em segundos e milissegundos desde o dia 1 de janeiro de 1970.

O primeiro experimento foi o armazenamento das imagens médicas para povoamento da base de dados que foi utilizada posteriormente nos demais experimentos. A fim de avaliar a escalabilidade do sistema, foram realizados testes variando-se o volume na base de dados, sendo esses valores 1000, 2500, 5000 e 10000 imagens, constituídas de tomografias computadorizadas, angiografias, ultrassons, ressonâncias magnéticas e raios-x. Em seguida, foram executados os cinco tipos de buscas do modelo de busca e recuperação do DICOM. Por fim, três cenários distintos para recuperação: a obtenção de centenas de imagens pertencente à um estudo, de dezenas de imagens de uma série e por uma única instância.

Uma vez que a avaliação foi por meio de comparação das médias do tempo de execução dos serviços realizados modelo proposto e o modelo atual foi necessária a obtenção de variáveis estatísticas a partir dos tempos coletados por imagem (no caso do armazenamento e recuperação) e por operação de busca durante a fase de consulta. Desta forma a partir do conjunto de dados coletados durante os experimentos foram extraídas a média, mediana e variância de acordo com a distribuição dos dados. Para obtenção do tipo de distribuição foram aplicados testes estatísticos ou análise de histograma.

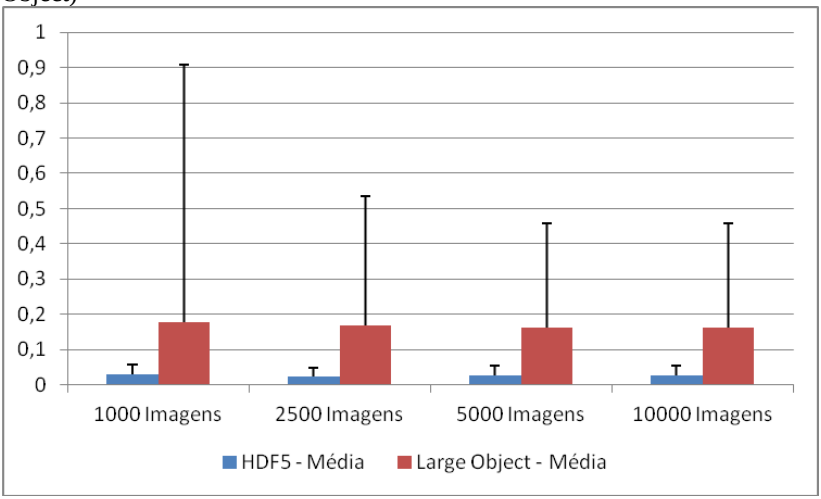
O cenário de busca contém poucos dados sobre o tempo de execução e por isso se optou pela aplicação do teste de hipótese nula chamado Shapiro Wilk (Shapiro e Wilk, 1965), sumarizado no APÊNDICE D - Análise Estatística e Distribuição dos Resultados de Consulta. Nos APÊNDICE B - Distribuição dos Resultados de Armazenamento e APÊNDICE C - Distribuição dos Resultados de Recuperação encontram-se os gráficos de distribuição dos resultados para análise do histograma. Destes processos foram encontrados dois tipos de distribuição: a normal e a exponencial. Desta forma as variáveis estatísticas foram obtidas por meio de fórmulas de acordo com a distribuição e sumarizadas na forma de gráficos a fim de comparar visualmente as diferenças dos dois modelos apresentados nesta dissertação.

Nas próximas seções, os gráficos dos resultados são apresentados da seguinte maneira: o eixo horizontal é a variação do volume dos exames enviados ou presentes na base; o eixo vertical é a média de tempo em segundos da execução de todas as iterações. Cada coluna é apresentada com sua barra de erro correspondente para identificar a variabilidade dos dados presentes nos resultados.

5.1 Armazenamento

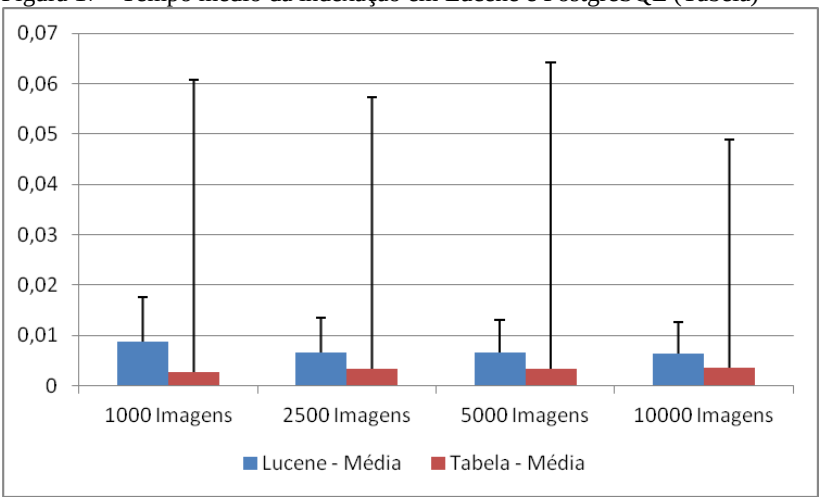
Os gráficos da Figura 16 comparam os tempos médios entre HDF5 e as funções que manipulam os *BLObs* do PostgreSQL, onde se pode concluir que para os experimentos realizados o HDF5 teve desempenho em torno de 600% mais eficiente e mostra uma menor variabilidade. Em contraste com o HDF5, o *BLOb* é suscetível ao tipo de modalidade a ser armazenada, por exemplo um exame de raio-x tem dimensões em torno de 3.000 x 3.000, entretanto uma ressonância magnética tem dimensão de 128 x 128.

Figura 16 - Tempo médio do armazenamento em HDF5 e PostgreSQL (Large Object)



A Figura 17 apresenta um comportamento de variabilidade alta na operação de inserção em tabela e índices do PostgreSQL, porém tem desempenho de 260 % superior ao CLucene. Também nota-se que a média no PostgreSQL apresenta um leve crescimento de acordo com o tamanho da base, enquanto que o CLucene uma leve declinação.

Figura 17 - Tempo médio da indexação em Lucene e PostgreSQL (Tabela)



5.2 Busca

A quantidade de dados referentes ao resultado das buscas são considerados pequenos (25 amostras de tempo por tipo de consulta), consequentemente em uma analise por histograma é difícil classificar corretamente à qual distribuição os dados se enquadram. Para verificação da normalidade da distribuição foi aplicado o teste Shapiro Wilk (Shapiro e Wilk, 1965) através da linguagem de programação R, uma linguagem voltada para estatística[9]. Os resultados dos testes encontram-se no anexo D e a maior parte dos testes rejeitaram a normalidade da distribuição, porém isso não significa que a distribuição não seja normal, apenas indica que com os dados presentes não é possível realizar a confirmação.

Dessa maneira, para não haver cálculos errôneos da variância dos dados, os gráficos do resultados apresentam a média, juntamente com barra de erros que contém o mínimo e máximo do tempo médio de busca, que para o CLucene se encontravam no intervalo de dados de milissegundos e o acesso às tabelas, exceto no operador universal de busca, no intervalo de microssegundos.

As *queries* utilizadas apresentam o seguinte formato no PostgreSQL: SELECT coluna1, ..., colunaN FROM tabela [WHERE

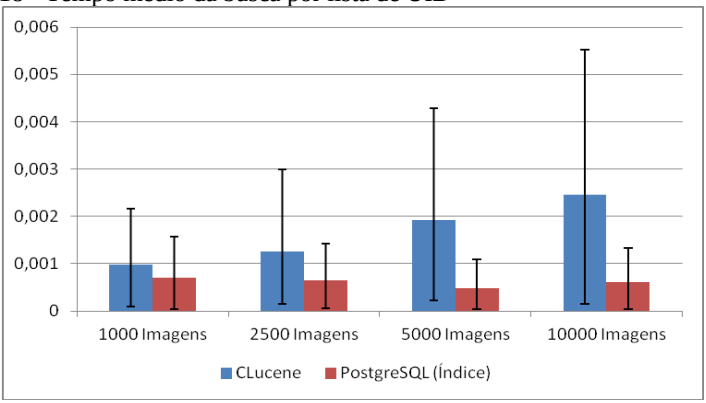
[9] http://www.r-project.org/

restrição]. Já no CLucene a busca tem o formato: +(CAMPO1:VALOR1). Isto é, o caractere '+' significa que o dado é obrigatório, o CAMPO1, que é o identificador único do *data element* (também chamado de *tag*), seria análogo a coluna do banco de dados relacional e o VALOR1 é similar a utilização da restrição WHERE na linguagem SQL.

A Figura 18 apresenta a busca por lista de UIDs e apresenta um crescimento linear, assim como uma variação maior que no banco de dados. Como os UIDs são chaves primárias das tabelas no PostgreSQL, os dados são buscados na estrutura de árvore B e, pela natureza desta estrutura, o tempo médio se manteve estável.

No PostgreSQL a busca de uma lista contendo 3 UIDs se dá com a *query* SELECT "studyInstanceUID" FROM "studies" WHERE ("studyInstanceUID" = 'UID1' OR "studyInstanceUID" = 'UID2' OR "studyInstanceUID" = 'UID3') e no CLucene com a *query* "+(0020000d:UID1 0020000d:UID2 0020000d:UID3)".

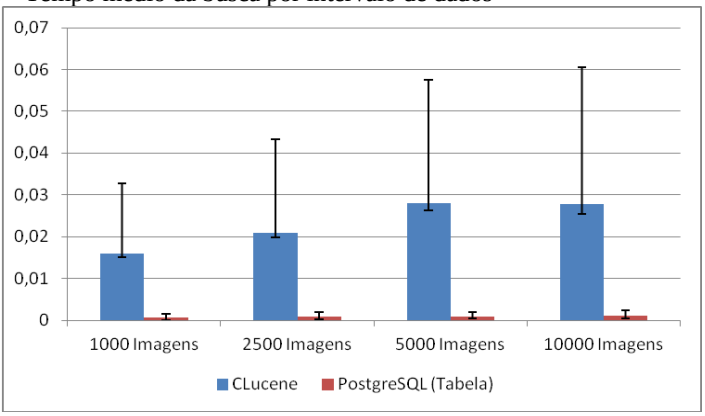
Figura 18 - Tempo médio da busca por lista de UID



A busca por intervalo de dados, na Figura 19, apresentou pior performance para a biblioteca CLucene, com diferenças no tempo médio de execução entre 140 % e 250 %. Para avaliar o desempenho buscou-se todos os exames desde 2009 até a data presente, ou seja, a maior parte dos exames presentes na avaliação levando-se em consideração que é uma amostra da base mostrada na Figura 1. No PostgreSQL a execução do teste se deu através da *query* "SELECT "studyDate" FROM "studies" WHERE "studyDate" >= '2009-1-1'" e no CLucene esta mesma busca é

feita com a *query* "+00080020:[20090101 TO 20111005]" em que a data 20111005 representa a data do dia da busca.

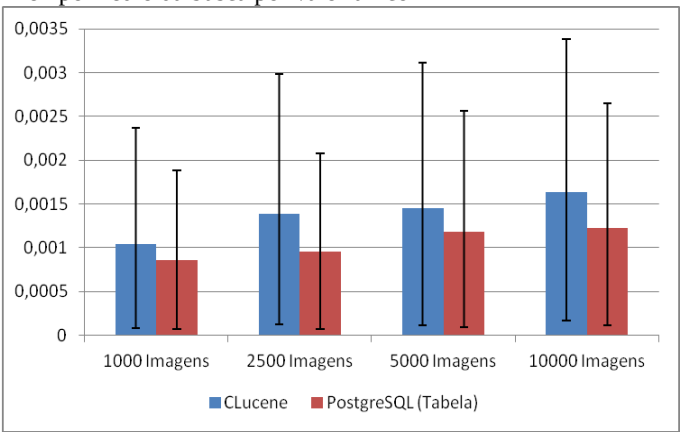
Figura 19 - Tempo médio da busca por intervalo de dados



A Figura 20 apresenta o resultado das buscas por valor único e o desempenho do PostgreSQL sobre o CLucene foi em 20 % melhor. Ambas abordagens apresentaram resultados com variações semelhantes.

Para este experimento buscou-se os exames que foram realizados na data 2011-03-21. Este tipo de busca de um único dado foi executada no PostgreSQL através da *query* "SELECT "studyDate" FROM "studies" WHERE "studyDate" = '2011-3-21'" e no CLucene por "+00080020:20110321".

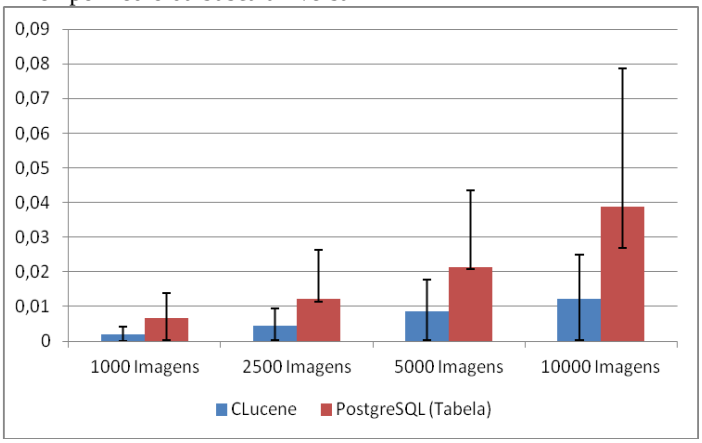
Figura 20 - Tempo médio da busca por valor único



A consulta pelo operador universal, isto é, por todos os valores pertencentes a um *data element* específico é realizada no PostgreSQL com *query* similar a "SELECT 'sopInstanceUID' from images" que realizada a busca de todos os UIDs de instância na tabela imagens, ou seja, procura por todos os identificadores únicos de imagens no banco de dados, já no CLucene não foi aplicado a *query* correspondente pois há um detrimento exponencial no desempenho conforme o tamanho da base aumenta. O Lucene armazena todos os termos de um campo em um formato específico, ou seja, para buscar todos os valores de um campo específico apenas é necessário ler esse arquivo que contêm os termos e filtrar pelo campo desejado.

A busca utilizando o operador universal apresenta um desempenho melhor no CLucene que em relação ao PostgreSQL em 350 % . Nota-se também que, a partir do gráfico na Figura 21, há um crescimento linear dos tempos de consulta conforme varia-se a quantidade de metadados a serem recuperados para as duas estratégias de armazenamento.

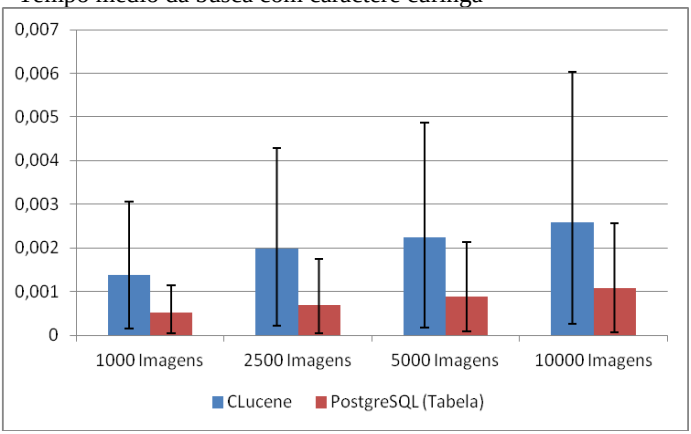
Figura 21 - Tempo médio da busca universal



Na consulta empregando caracteres curinga, os resultados da Figura 22 mostram que as duas abordagens tem um pequeno crescimento com o aumento do tamanho da base de dados e que o PostgreSQL executa a operação de forma mais rápida em 285 % e com menor variação na média. Este experimento foi realizado no PostgreSQL através da *query* "SELECT "patientsName" FROM

"patients" WHERE "patientsName" ILIKE '%TO'" e no CLucene através da *query* "+00100010:*TO".

Figura 22 - Tempo médio da busca com caractere curinga



5.3 Recuperação

A operação de recuperação no PostgreSQL incluí a recuperação dos *oids* de cada exame dos cenários de estudo (centenas de exames), série (dezenas de exames) e instância (um único IOD). O mesmo ocorre no Lucene, em que são recuperados os UIDs dos níveis para construção do caminho para extração do conteúdo do HDF5.

Mesmo com a sobrecarga do tempo de execução na leitura dos índices pela biblioteca CLucene durante a recuperação dos dados no HDF5, este tem uma performance aproximada de 580% melhor em relação ao PostgreSQL, em que o serviço de recuperação realiza as operações de acesso ao índice e leitura dos *large objects*. Os resultados do tempo médio por recuperação de cada exame de estudo e série são apresentados respectivamente na Figura 23 e Figura 24.

Figura 23 - Tempo médio de recuperação por exame de um estudo completo de acordo com o volume da base

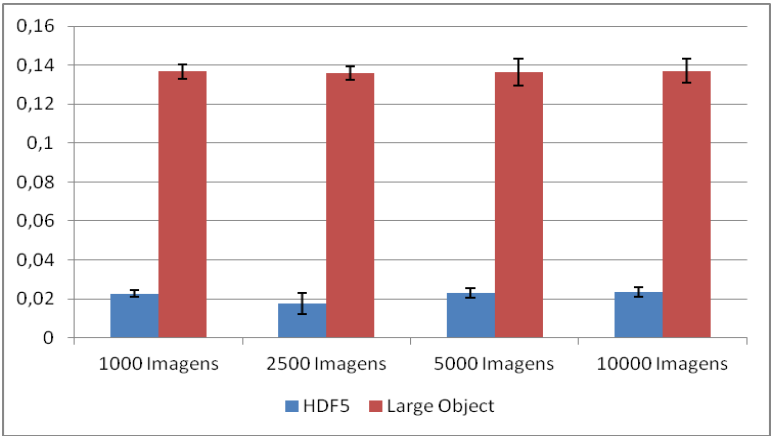
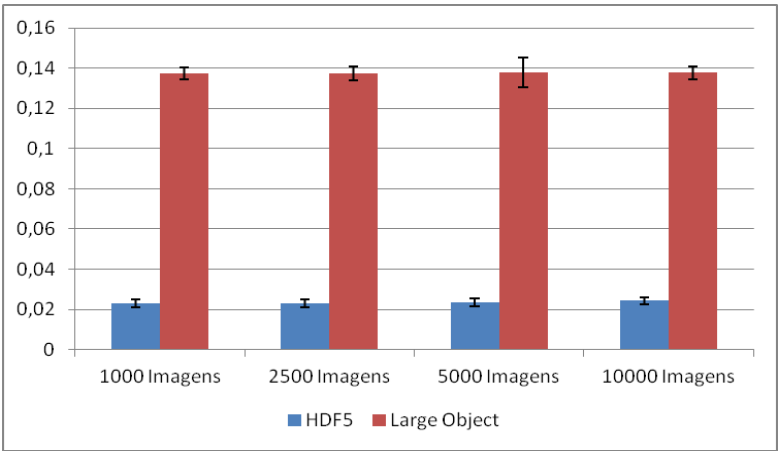
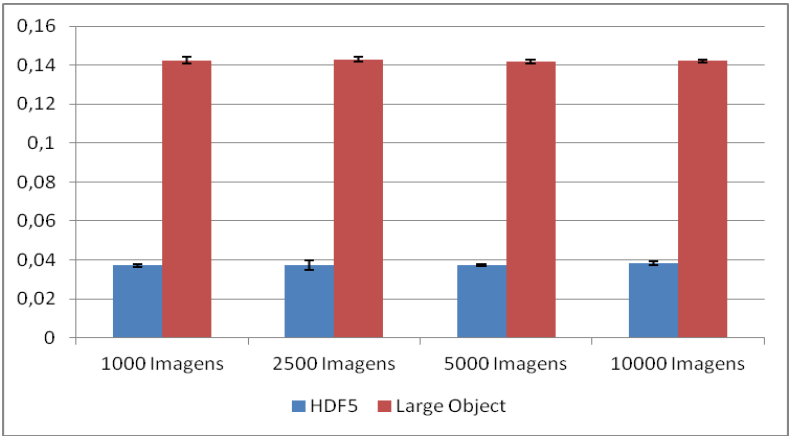


Figura 24 - Tempo médio de recuperação por exame de uma série completa de acordo com o volume da base



A Figura 25 representa a operação de um único exame recuperado e tem superioridade na performance pelo HDF5 e Lucene em 380%. Diferente do armazenamento, no geral, os resultados apresentaram uma variação baixa.

Figura 25 - Tempo médio de recuperação de uma instância de acordo com o volume da base



O tempo total dos serviços executados, incluindo as latências de rede, para cada cenário e para cada volume da base está sumarizado na Tabela 5. Em geral os tempos de busca, desde a requisição do serviço pela SCU, envio de resposta do SCP e finalização da associação entre as entidades, não impactaram no desempenho do serviço como um todo, uma vez que é na escala de milésimos de segundo. O maior impacto se concentrou no serviço *C-Store*.

Tabela 5 - Tempo total da execução dos serviços (em segundos)

		HDF5-Lucene				*PostgreSQL*			
Nº de Imagens		1000	2500	5000	10000	1000	2500	5000	10000
C-Store		180	420	1020	2160	480	1260	2400	5100
C-Move	**Estudo**	17	17	16	15	20	20	24	18
	Série	14	14	13	12	16	16	19	16
	Instância	4	4	4	4	4	4	5	10
C-Find	**Lista de UID**	4	4	4	4	4	4	4	4
	Intervalo	5	5	5	5	4	5	5	5
	Valor Único	4	4	4	4	4	4	4	4
	Universal	3	4	5	8	5	6	7	20
	WildCard	4	4	4	4	4	4	4	4

O tamanho das imagens armazenadas em HDF5, índice no formato do Lucene, PostgreSQL e no sistema de arquivos em DICOM

part 10 para cada conjunto de imagens de teste encontra-se na Tabela 6. O tamanho reduzido no PostgreSQL em relação ao sistema de arquivos, dá-se pelo tamanho do bloco de dados, que é 2 bytes (Douglas, 2005), e o overhead de 25 % do HDF5 em relação às imagens armazenadas em DICOM part 10 deve-se ao tamanho alocado pelo HDF5 para 4 estruturas internas de metadados, sendo elas: Árvore B, *Heap*, Grupos e *Dataset.*

Tabela 6 - Tamanho da base de dados

Tipo de Armazenamento	Volume (em nº de imagens)			
	1000	2500	5000	10000
HDF5	695 MB	1.7 GB	3.4 GB	6.6 GB
CLucene	1004 k	1.2 MB	2 MB	3 MB
PostgreSQL (Tabela / *Blob*)	488 MB	1.2 GB	2.4 GB	4.7 GB
Sistema de Arquivos	562 MB	1.3 GB	2.7 GB	5.3 GB

O overhead em megabytes das estruturas internas do HDF5 encontra-se na Tabela 7 e foram obtidos através da ferramenta *h5stat*. Os metadados do HDF5 são compostos pela estrutura Árvore-B, que tem como função indexar os dados internamente para acesso dos grupos, a *heap* armazena os endereços e o nome dos links dos objetos em uma tabela de símbolos e por fim os dois últimos metadados são as estruturas que armazenam as informações sobre os grupos e *datasets*. De acordo com a documentação da API do HDF5, estas estruturas são alocadas em blocos de 2 *kilobytes*, que podem ser manipulados a fim de otimizar o espaço ocupado, através da alteração de uma lista de propriedades durante a criação dos grupos e *datasets*.

Tabela 7 - Overhead das estruturas internas do HDF5 em *megabytes*

Nº de Imagens	*Árvore B*	*Heap*	*Grupos*	*Datasets*	*Total*
1000	73,25034	11,04489	2,920837	54,96855	142,1846
2500	158,6162	24,13305	6,276169	122,5712	311,5965
5000	338,1469	51,05531	13,44059	256,0062	658,6491
10000	673,8131	101,5483	26,76872	509,7211	1311,851

6. CONCLUSÕES E TRABALHOS FUTUROS

Nesta dissertação foi apresentada uma nova abordagem para o desenvolvimento de uma camada para persistência de dados usando o formato de dados hierárquicos HDF5. Entretanto, como já visto, nativamente este formato não dispõe de mecanismos de manipulação dos dados armazenados similar ao banco de dados relacional que emprega a linguagem SQL. Dessa maneira, foi necessária a criação de um mecanismo de indexação dos termos referentes ao modelo de busca e recuperação de objetos DICOM, usando o arquivo invertido Lucene.

Durante a pesquisa do modelo proposto, procurou-se abranger todas as operações oferecidas pelo provedor de serviço em conformidade com o padrão DICOM que são relevantes a serem utilizadas em um ambiente PACS em produção. Os experimentos foram realizados na RCTM, com dados radiológicos heterogêneos, para desta forma, simular com a maior proximidade possível um ambiente em operação.

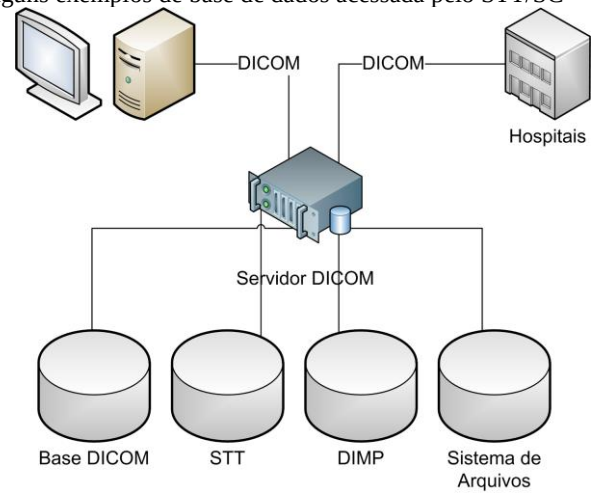
Foram realizados experimentos que mostraram que as operações de armazenamento e recuperação foram até seis vezes mais rápidas usando HDF5, em conjunto com o CLucene, que as mesmas executadas em PostgreSQL. No decorrer da pesquisa foi constatada uma sobrecarga no modelo proposto causada pelo uso do CLucene, portanto, há a necessidade de investigação a fundo para um aproveitamento melhor da solução de índice invertido em relação aos resultados promissores do PostgreSQL. É importante salientar que mesmo com esta sobrecarga na operação de busca, o processo de recuperação completo, que inclui a operação de busca pelo identificador do exame, teve um desempenho superior, o que mostra que mesmo assim o resultado final ainda é positivo.

É necessária também a avaliação em um cenário mais fiel com a realidade da RCTM, como o comportamento da SCP diante de múltiplas requisições de serviço por diferentes SCUs.

Trabalhos futuros podem aproveitar as características do formato de dados HDF5, como por exemplo, o acesso randômico aos dados em ferramentas de auxílio ao diagnóstico, como visualização multimodal ou processamento digital de imagens. Por exemplo, no modelo atual visto na Figura 26 são mostradas algumas entidades pertencentes ao PACS utilizado no STT/SC. Cada entidade pode acessar uma ou mais bases de acordo com a necessidade. Cada base contém dados já extraídos e processados para manipulação posterior, como por exemplo, uma

aplicação *web* chamada DIMP que manipula imagens médicas dinamicamente via *browser*. Desta maneira, todas as bases podem compartilham de metadados em comum, criando uma redundância de informação. Consequentemente também é necessário manutenção de múltiplas bases. Já com o modelo proposto poderia haver uma redução no número de bases, uma vez que é possível acessar qualquer metadado diretamente.

Figura 26 - Alguns exemplos de base de dados acessada pelo STT/SC



Como é visto na Figura 27, se tal operação é aplicada no modelo atual é necessária a obtenção do objeto DICOM, que está como *large object*, realização da análise (*parsing*) da instância do exame, extração dos dados e processamento. Já com o modelo proposto, na Figura 28, aproveitando a propriedade de acesso randômico, isto é, leitura a qualquer metadado armazenado no HDF5, permite, por exemplo, a execução de extrair o metadado e processamento imediato. A validação de cenários específicos é um dos temas para trabalho futuro e pode minimar o número de bases existentes.

Figura 27 - Exemplo de fluxo de processamento no modelo atual

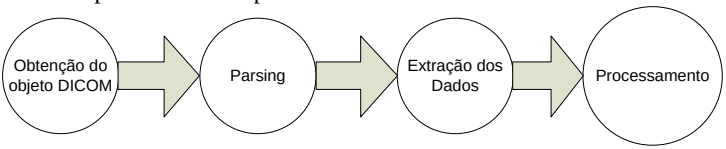
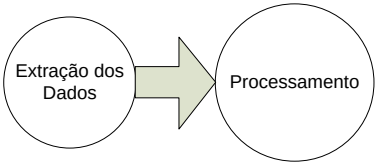


Figura 28 - Exemplo de fluxo de processamento no modelo proposto



Também deve ser analisada a incorporação do laudo estruturado do padrão DICOM (DICOM SR) ao formato HDF5 juntamente com o estudo relacionado ao diagnóstico. Ainda, a avaliação deste tipo de modelo em cenários complexos e distribuídos figura como uma proposta de trabalho futuro a ser executada.

REFERÊNCIAS

ABDULJWAD, F.; NING, W.; DE, X. SMX/R: Efficient way of storing and managing XML documents using RDBMSs based on paths. 2010. IEEE. p.V1-143-V1-147.

ACR. **ACR PRACTICE GUIDELINE FOR THE PERFORMANCE OF SCREENING AND DIAGNOSTIC MAMMOGRAPHY**: American College Of Radiology. 24**:** 1 p. 2008.

AMARAL, E. C., E.; DANTAS, M.A.R.; MACEDO, DOUGLAS D. J. DE. **Replicação Distribuída de Imagens Médicas sob o Formato de Dados HDF5**. 8th International Information and Telecommunication Technologies Symposium Florianópolis, SC - Brazil: I2TS 2009.

BLANQUER, I.; HERNANDEZ, V.; SEGRELLES, D. TRENCADIS – A Grid Architecture for Creating Virtual Repositories of DICOM Objects in an OGSA-Based Ontological Framework Biological and Medical Data Analysis. In: MAGLAVERAS, N.;CHOUVARDA, I., *et al* (Ed.): Springer Berlin / Heidelberg, v.4345, 2006. p.183-194. (Lecture Notes in Computer Science). ISBN 978-3-540-68063-5.

BMJ. **Regulamento sobre a Prevenção de Danos por Raios X**: Ministério Federal da Justiça Alemã. 28 1987.

CFM. **RESOLUÇÃO CFM nº 1.639/2002**. CFM: Conselho Federal de Medicina. 1639 2002.

_____. **RESOLUÇÃO CFM Nº 1.821/07**. CFM: Conselho Federal de Medicina. 1821 2007.

CHAN, C. Y.; IOANNIDIS, Y. E. Bitmap index design and evaluation. 1998. ACM. p.355-366.

CLUCENE. CLucene - lightning fast C++ search engine. 2011. Disponível em: < http://clucene.sourceforge.net/ >. Acesso em: 2011.

CODD, E. F. A relational model of data for large shared data banks. **Communications of the ACM,** v. 13, n. 6, p. 377-387, 1970. ISSN 0001-0782.

COHEN, S. et al. Scientific formats for object-relational database systems: a study of suitability and performance. **SIGMOD Rec.,** v. 35, p. 10-15, 2006. Disponível em: < http://doi.acm.org/10.1145/1147376.1147378 >.

COSTA, C. et al. Indexing and retrieving DICOM data in disperse and unstructured archives. **International Journal of Computer Assisted Radiology and Surgery,** v. 4, p. 71-77, 2009. Disponível em: < http://dx.doi.org/10.1007/s11548-008-0269-7 >.

DE MACEDO, D. D. J. et al. An Improvement of a Different Approach for Medical Image Storage. Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2011 20th IEEE International Workshops on, 2011. 27-29 June 2011. p.140-142.

_____. An architecture for DICOM medical images storage and retrieval adopting distributed file systems. **International Journal of High Performance Systems Architecture,** v. 2, n. 2, p. 99-106, 2009. ISSN 1751-6528.

DOUGLAS, K. **PostgreSQL**. Sams, 2005. ISBN 0672327562.

ERBERICH, S. G. et al. Globus medicus-federation of dicom medical imaging devices into healthcare grids. **Studies in Health Technology and Informatics,** v. 126, p. 269, 2007. ISSN 0926-9630.

ERIK, H.; OTIS, G.; MICHAEL, M. C. **Lucene in action**: Manning Publica—tions Co 2005.

FOLINO, G.; SHAH, A. A.; KRANSNOGOR, N. On the storage, management and analysis of (multi) similarity for large scale protein structure datasets in the grid. Computer-Based Medical Systems, 2009. CBMS 2009. 22nd IEEE International Symposium on, 2009. **http://dx.doi.org/10.1109/CBMS.2009.5255328**, 2009/aug. p.1-8.

FOLK, M.; POURMAL, E. Balancing performance and preservation lessons learned with HDF5. 2010. ACM. p.11.

GARCIA RUIZ, M. et al. mantisGRID: A Grid Platform for DICOM Medical Images Management in Colombia and Latin America. **Journal of Digital Imaging,** v. 24, p. 271-283, 2011. Disponível em: < http://dx.doi.org/10.1007/s10278-009-9265-x >.

GOSINK, L. et al. HDF5-FastQuery: Accelerating Complex Queries on HDF Datasets using Fast Bitmap Indices. Scientific and Statistical Database Management, 2006. 18th International Conference on, 2006. **http://dx.doi.org/10.1109/SSDBM.2006.27**, 2006. p.149-158.

HAERDER, T.; REUTER, A. Principles of transaction-oriented database recovery. **ACM Computing Surveys (CSUR),** v. 15, n. 4, p. 287-317, 1983. ISSN 0360-0300.

HDFGROUP. The HDF Group - Information, Support, and Software. 2011a. Disponível em: < http://www.hdfgroup.org/ >. Acesso em: 22/10/2011.

_____. Who uses HDF? , 2011b. Disponível em: < http://www.hdfgroup.org/users.html >. Acesso em: 22/10.

HUANG, H. **PACS and imaging informatics: basic principles and applications**. Wiley-Blackwell, 2010. ISBN 0470373725.

ISO/IEC. **Information technology -- Database languages**. Part 11: Information and Definition Schemas (SQL/Schemata): ISO/IEC 2008a.

_____. **Information technology -- Database languages**. Part 2: Foundation (SQL/Foundation): ISO/IEC 2008b.

_____. **Information technology -- Database languages**. Part 1: Framework (SQL/Framework): ISO/IEC 2008c.

KITCHENHAM, B. Procedures for performing systematic reviews. **Keele, UK, Keele University,** v. 33, 2004.

KORNACKER, M.; MOHAN, C.; HELLERSTEIN, J. M. Concurrency and recovery in generalized search trees. 1997. ACM. p.62-72.

LEE, K. P.; SPENCE, P. L. View_HDF: visualization and analysis tool for Hierarchical Data Format files. 2002. IEEE. p.744-750 vol. 2.

LEVENSHTEIN, V. I. Binary codes capable of correcting deletions, insertions, and reversals. 1966. p.707-710.

LUCENE, A. Lucene. 2011a. Disponível em: < http://lucene.apache.org >.

_____. Powered by. 2011b. Disponível em: < wiki.apache.org/lucene-java/PoweredBy >.

MACEDO, D. D. J. D. et al. **Armazenamento distribuído de imagens médicas DICOM no formato de dados HDF5**. Proceedings of the 14th Brazilian Symposium on Multimedia and the Web. Vila Velha, Brazil: ACM**:** 20-27 p. 2008.

MAIA, R. S.; WANGENHEIM, A. V.; NOBRE, L. F. **A Statewide Telemedicine Network for Public Health in Brazil**. Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems: IEEE Computer Society**:** 495-500 p. 2006.

MONTAGNAT, J. et al. A Secure Grid Medical Data Manager Interfaced to the gLite Middleware. **Journal of Grid Computing,** v. 6, n. 1, p. 45-59, 2008. ISSN 1570-7873. Disponível em: < http://dx.doi.org/10.1007/s10723-007-9088-2 >.

NAM, B.; SUSSMAN, A. Improving access to multi-dimensional self-describing scientific datasets. Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on, 2003. 12-15 May 2003. p.172-179.

NEMA. **Digital Imaging and Communications in Medicine (DICOM) Part 1: Introduction and Overview** 2011a

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 2: Conformance**. 2011b

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 3: Information Object Definition**. 2011c

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 4: Service Class Specifications**. 2011d

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 5: Data Structures and Encoding**. 2011e

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 6: Data Dictionary**. 2011f

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 7: Message Exchange** 2011g

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 7: Message Exchange**. 2011h

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 10: Media Storage and File Format for Media Interchange**. 2011i

_____. **Digital Imaging and Communications in Medicine (DICOM) Part 18: Web Access to DICOM Persistent Objects (WADO)**. 2011j

ORACLE. Oracle Database 11g DICOM Medical Image Support. 2011. Disponível em: < http://www.oracle.com/technetwork/database/multimedia/overview/dicom11gr2-wp-medimgsupport-133109.pdf >.

PIANYKH, O. S. DICOM and Teleradiology
Digital Imaging and Communications in Medicine (DICOM). In: (Ed.): Springer Berlin Heidelberg, 2012. p.281-317. ISBN 978-3-642-10850-1.

POSTGRESQL. SQL Conformance. 2009a. Disponível em: < http://www.postgresql.org/docs/8.4/static/features.html) >.

_____. TOAST. 2009b. Disponível em: < http://www.postgresql.org/docs/8.4/static/storage-toast.html >.

_____. Users. 2011. Disponível em: < http://www.postgresql.org/about/users/ >.

POWER, D. et al. **A relational approach to the capture of DICOM files for Grid-enabled medical imaging databases**. Proceedings of the 2004 ACM symposium on Applied computing. Nicosia, Cyprus: ACM**:** 272-279 p. 2004.

PYTABLES. PyTables - Getting the most *out* of your data. 2011. Disponível em: < www.pytables.org/ >. Acesso em: 20/11/2011.

RCR. **Retention and Storage of Images and Radiological Patient Data**. RCR: The Royal College Of Radiologists. BFCR(06)4**:** 3 p. 2008.

RON, E. Cancer risks from medical radiation. **Health physics,** v. 85, n. 1, p. 47, 2003. ISSN 0017-9078.

SAHOO, S.; AGRAWAL, G. Supporting XML Based High-Level Abstractions on HDF5 Datasets: A Case Study in Automatic Data Virtualization. In: EIGENMANN, R.;LI, Z., *et al* (Ed.). **Languages and Compilers for High Performance Computing**: Springer Berlin / Heidelberg, v.3602, 2005. p.922-922.

SALTON, G.; WONG, A.; YANG, C. S. A vector space model for automatic indexing. **Commun. ACM,** v. 18, n. 11, p. 613-620, 1975. ISSN 0001-0782.

SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples). **Biometrika,** v. 52, n. 3/4, p. 591-611, 1965. ISSN 0006-3444.

SHASHARINA, S. G. et al. Distributed Technologies for Remote Access of HDF Data. Enabling Technologies: Infrastructure for Collaborative Enterprises, 2007. WETICE 2007. 16th IEEE International Workshops on, 2007. 18-20 June 2007. p.255-260.

SPHINX. Sphinx. 2011. Disponível em: < http://sphinxsearch.com >.

VAN DE WETERING, R.; BATENBURG, R. A PACS maturity model: a systematic meta-analytic review on maturation and evolvability of PACS in the hospital enterprise. **International journal of medical informatics,** v. 78, n. 2, p. 127-140, 2009. ISSN 1386-5056.

WALLAUER, J. et al. Building a national telemedicine network. **IT Professional,** v. 10, n. 2, p. 12-17, 2008. ISSN 1520-9202.

XAPIAN. The Xapian Project. 2011. Disponível em: < http://xapian.org >. Acesso em: 2011.

YU, H. et al. High performance file I/O for the Blue Gene/L supercomputer. 2006. IEEE. p.187-196.

APÊNDICE A - Publicações

Título: Replicação Distribuída de Imagens Médicas sob o Formato de Dados HDF5
Autores: AMARAL, E. ; PRADO, T. C. ; COMUNELLO, E ; MACEDO, D. D. J. ; DANTAS, M. A. R.
Ano: 2009
Evento: 8th International Information and Telecommunication Technologies Symposium
Local: Florianópolis
Estrato Qualis: B4

Título: A Study Comparison of Store and Retrieve of DICOM Images using High Performance Data Format
Autores: MACEDO, D. D. J. ; COELHO, T. P ; COMUNELLO, E ; DANTAS, M. A. R.
Ano: 2009
Evento: High Performance Computing Symposium
Local: Kingston
Estrato Qualis: B3

Título: An Improvement of a Different Approach for Medical Image Storage
Autores: MACEDO, D. D. J. ; CAPRETZ, M. ; PRADO, T. C. ; VONWANGENHEIM, A ; DANTAS, M. A. R.
Ano: 2011
Evento: 20th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises
Local: Paris
Estrato Qualis: B1

Título: A Study of NetCDF as an Approach for High Performance Medical Image Storage
Autores: MAGNUS, M. S. ; PRADO, T. C. ; VONWANGENHEIM, A; MACEDO, D. D. J. ; DANTAS, M. A. R.
Ano: 2012
Local: Journal of Physics: Conference Series
Estrato Qualis: B4

APÊNDICE B - Distribuição dos Resultados de Armazenamento

1. PostgreSQL

Entre as Figura 29 e Figura 32 encontram-se os histograma, que apresentam uma distribuição normal, dos resultados de armazenamento de metadados pelo PostgreSQL.

Figura 29 - Gráficos de distribuição dos tempos de armazenamento dos metadados de 1000 imagens em tabela

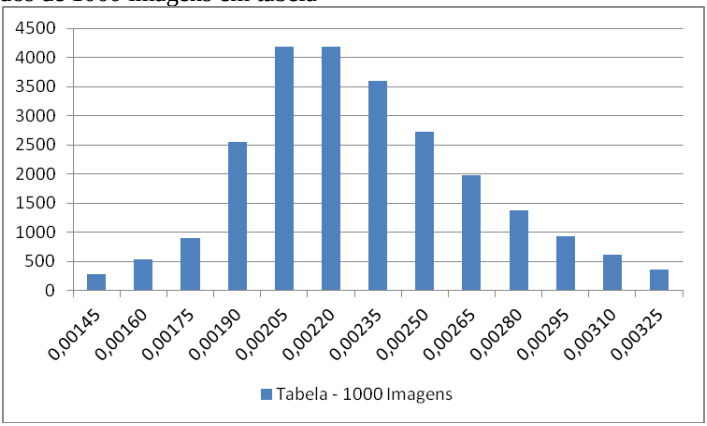


Figura 30 - Gráficos de distribuição dos tempos de armazenamento dos metadados de 2500 imagens em tabela

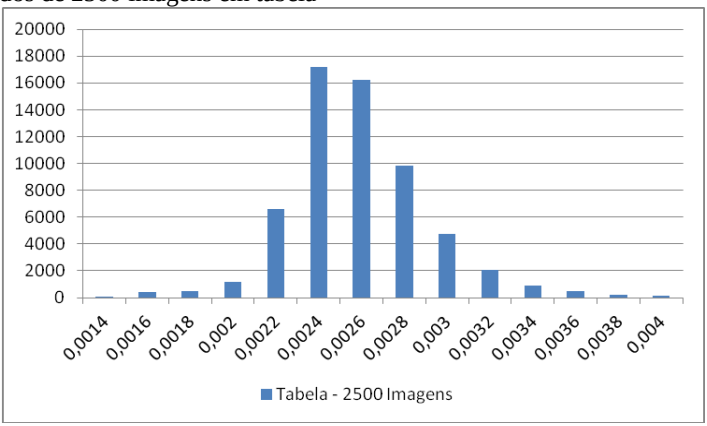


Figura 31 - Gráficos de distribuição dos tempos de armazenamento dos metadados de 5000 imagens em tabela

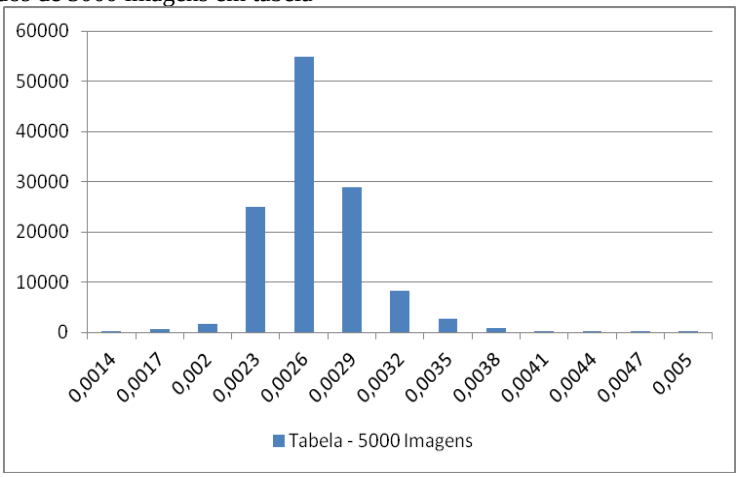
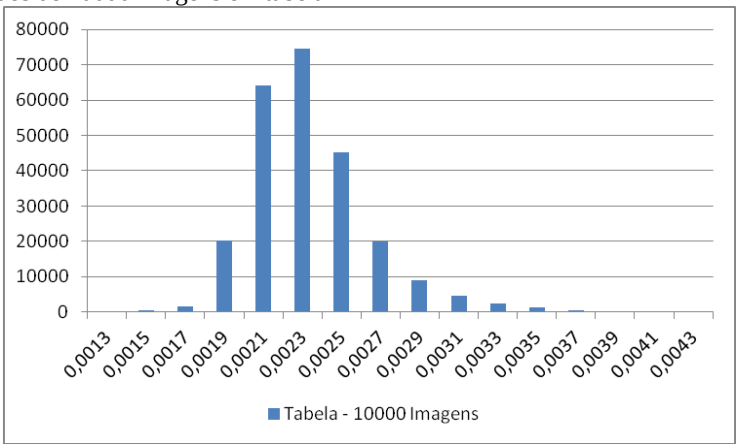


Figura 32 - Gráficos de distribuição dos tempos de armazenamento dos metadados de 10000 imagens em tabela



Entre as Figura 33 e Figura 36 encontram-se os histograma, que apresentam uma distribuição normal, dos resultados de armazenamento de *large objects* pelo PostgreSQL.

Figura 33 - Gráficos de distribuição dos tempos de armazenamento de 1000 imagens em Large Object

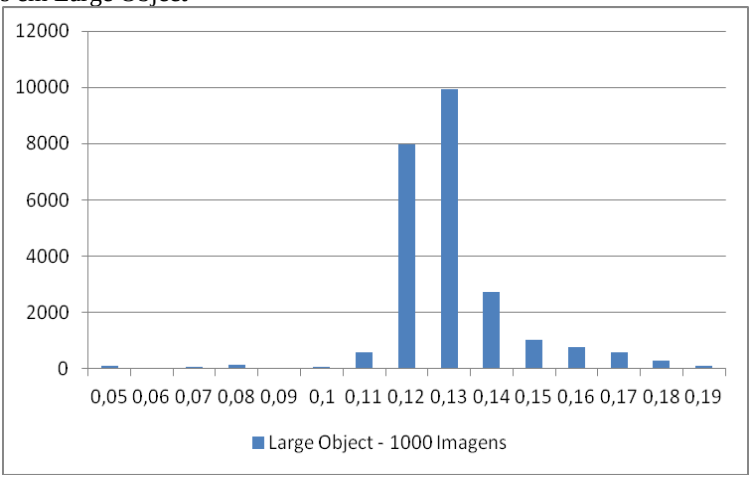


Figura 34 - Gráficos de distribuição dos tempos de armazenamento de 2500 imagens em Large Object

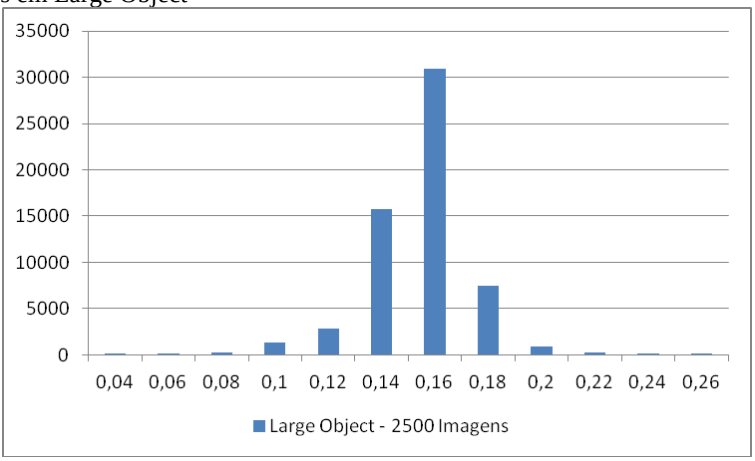


Figura 35 - Gráficos de distribuição dos tempos de armazenamento de 5000 imagens em Large Object

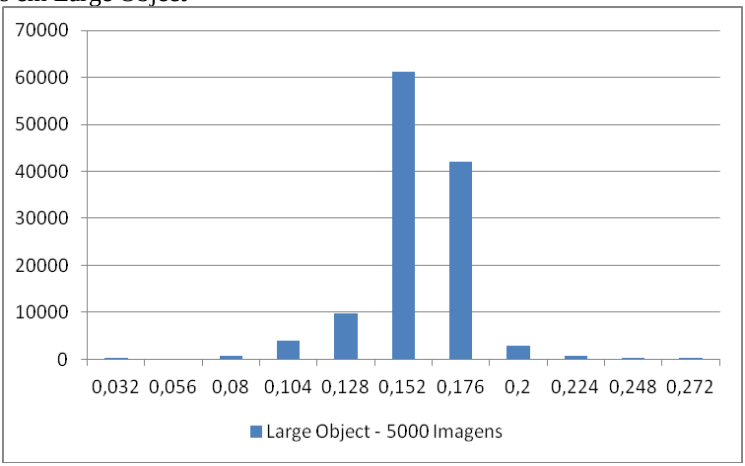
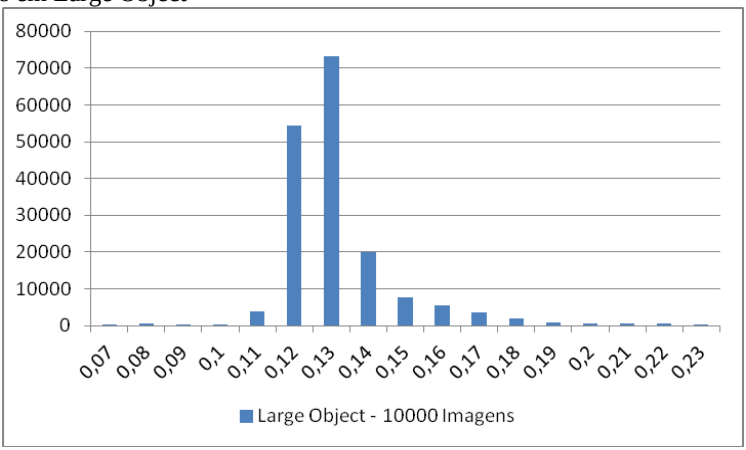


Figura 36 - Gráficos de distribuição dos tempos de armazenamento de 10000 imagens em Large Object



2. HDF5

Entre as Figura 37 e Figura 40 encontram-se os histograma, que apresentam uma distribuição exponencial, dos resultados de armazenamento no formato HDF5.

Figura 37 - Gráficos de distribuição dos tempos de armazenamento de 1000 imagens em HDF5

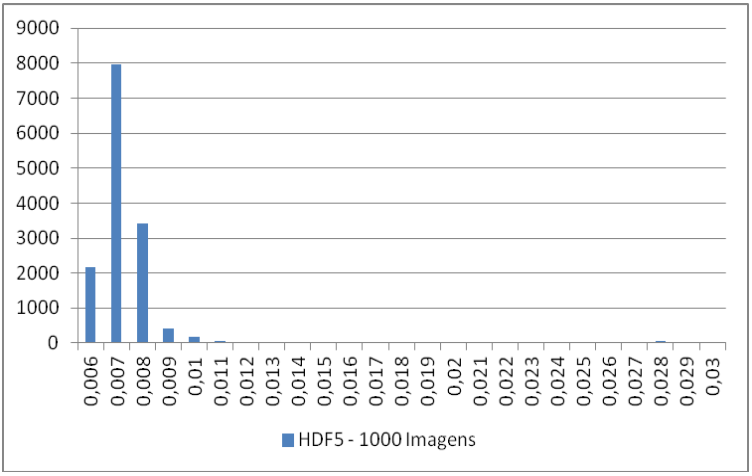


Figura 38 - Gráficos de distribuição dos tempos de armazenamento de 2500 imagens em HDF5

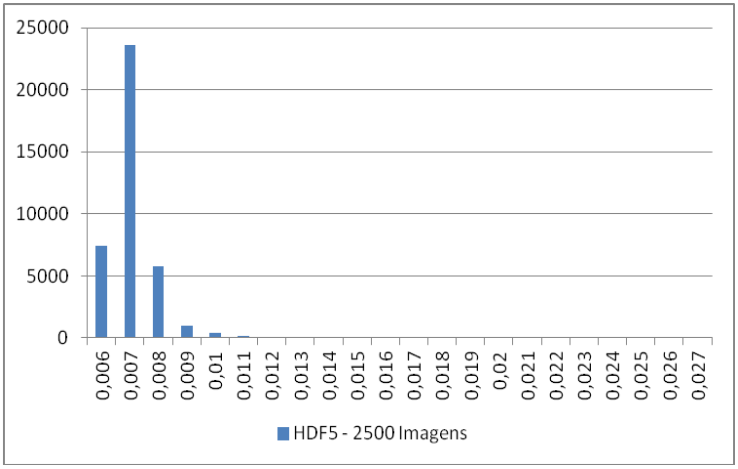


Figura 39 - Gráficos de distribuição dos tempos de armazenamento de 5000 imagens em HDF5

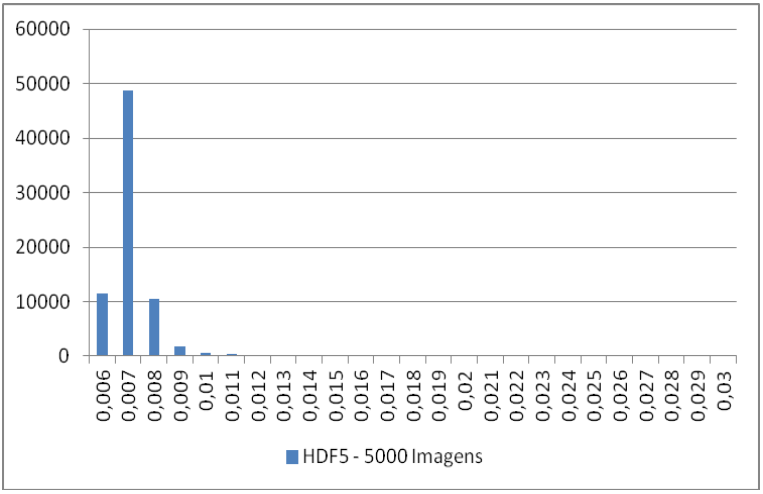
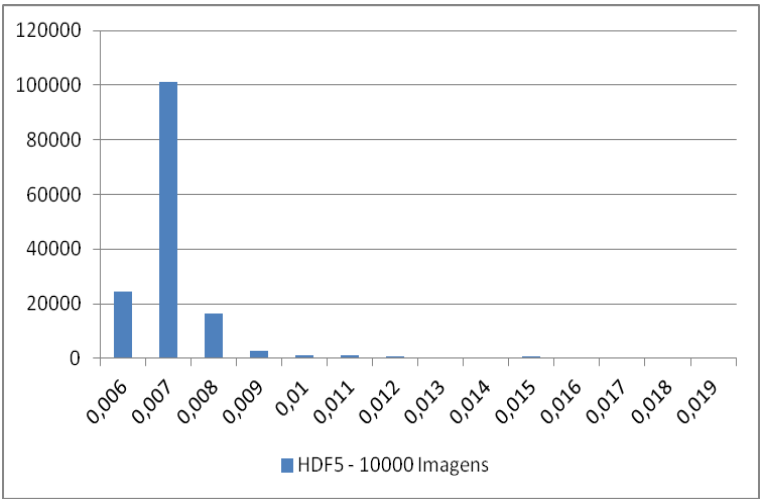


Figura 40 - Gráficos de distribuição dos tempos de armazenamento de 10000 imagens em HDF5



3. CLucene

Entre as Figura 41 e Figura 44 encontram-se os histograma, que apresentam uma distribuição exponencial, dos resultados de armazenamento no índice Lucene.

Figura 41 - Gráficos de distribuição dos tempos de indexação dos metadados de 1000 imagens em Lucene

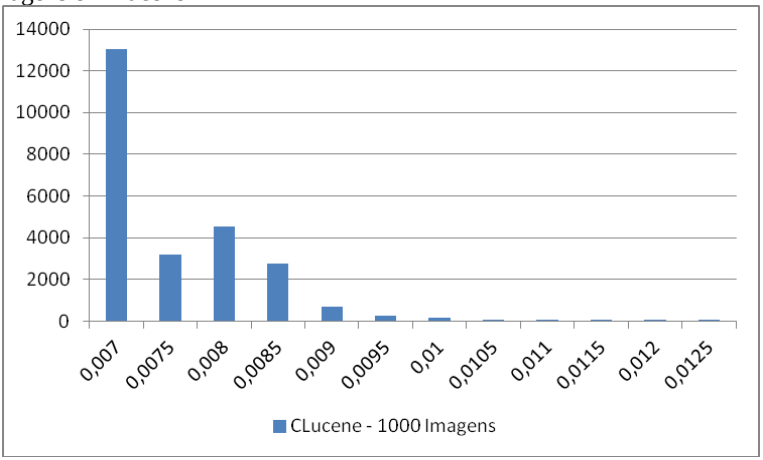


Figura 42 - Gráficos de distribuição dos tempos de indexação dos metadados de 2500 imagens em Lucene

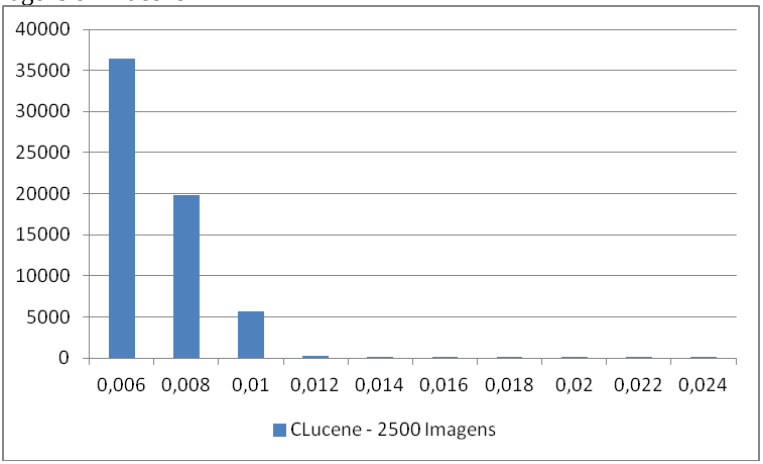


Figura 43 - Gráficos de distribuição dos tempos de indexação dos metadados de 5000 imagens em Lucene

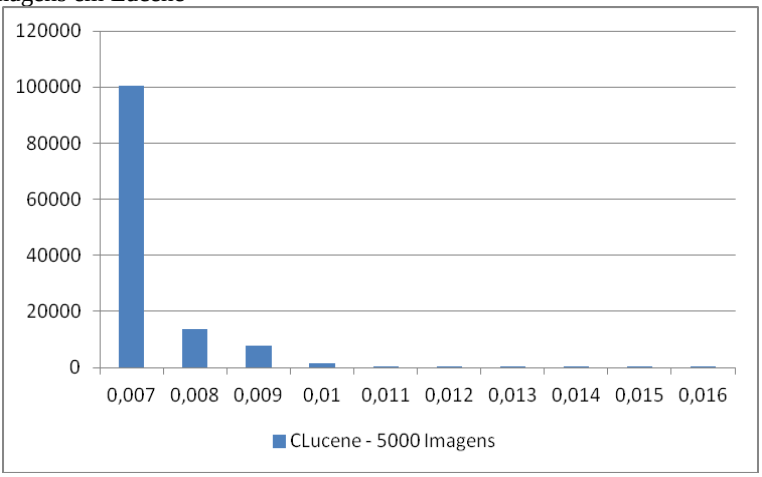
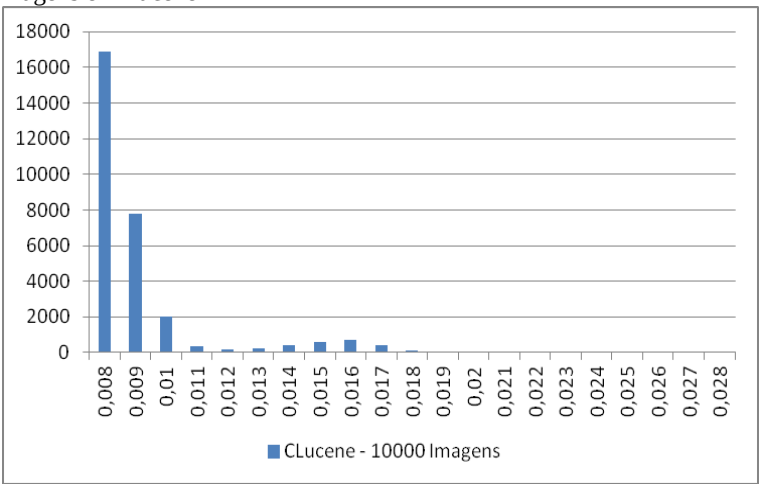


Figura 44 - Gráficos de distribuição dos tempos de indexação dos metadados de 10000 imagens em Lucene



APÊNDICE C - Distribuição dos Resultados de Recuperação

1. PostgreSQL

Entre as Figura 45e Figura 52 encontram-se os histograma, que apresentam uma distribuição normal, dos resultados de recuperação de estudo e série armazenados no PostgreSQL.

1.1 Estudo

Figura 45 - Distribuição dos resultados de recuperação de estudo do Large Object com base de dados contendo 1000 imagens

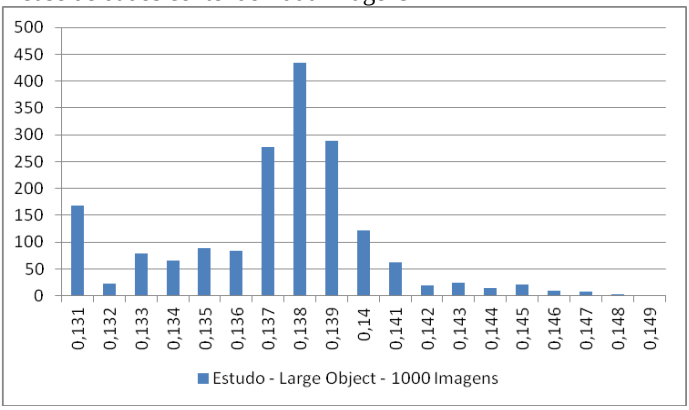


Figura 46 - Distribuição dos resultados de recuperação de estudo do Large Object com base de dados contendo 2500 imagens

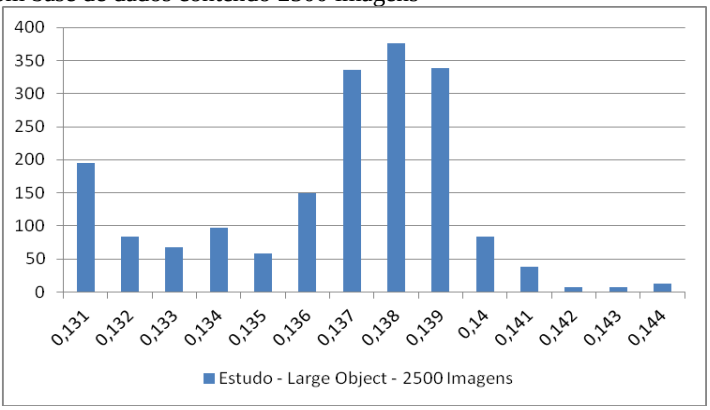


Figura 47 - Distribuição dos resultados de recuperação de estudo do Large Object base de dados contendo 5000 imagens

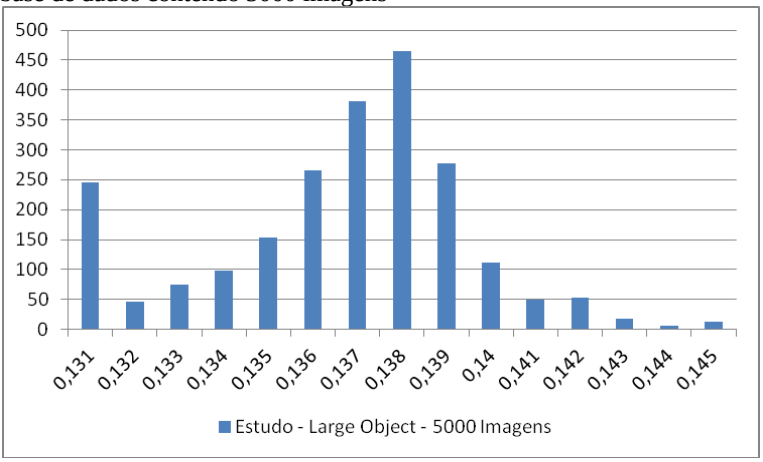
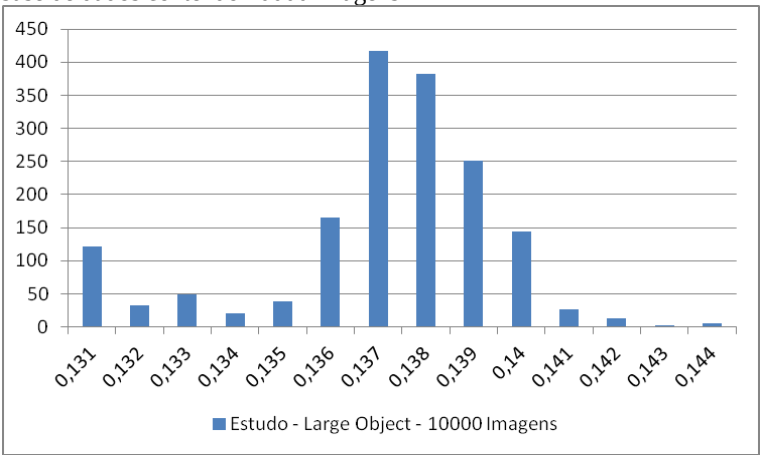


Figura 48 - Distribuição dos resultados de recuperação de estudo do Large Object base de dados contendo 10000 imagens



1.2 Series

Figura 49 - Distribuição dos resultados de recuperação de série do Large Object com base de dados contendo 1000 Imagens

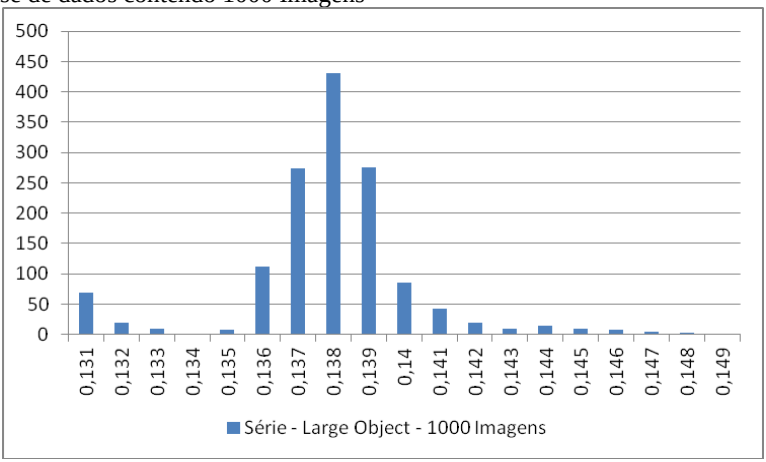


Figura 50 - Distribuição dos resultados de recuperação de série do Large Object com base de dados contendo 2500 Imagens

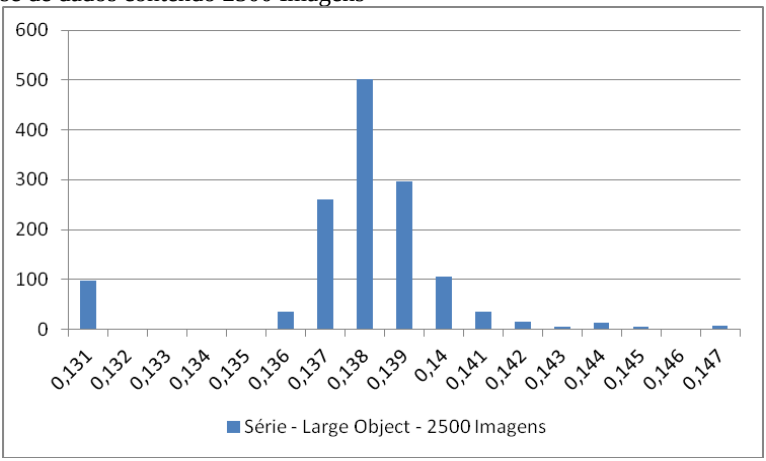


Figura 51 - Distribuição dos resultados de recuperação de série do Large Object com base de dados contendo 5000 Imagens

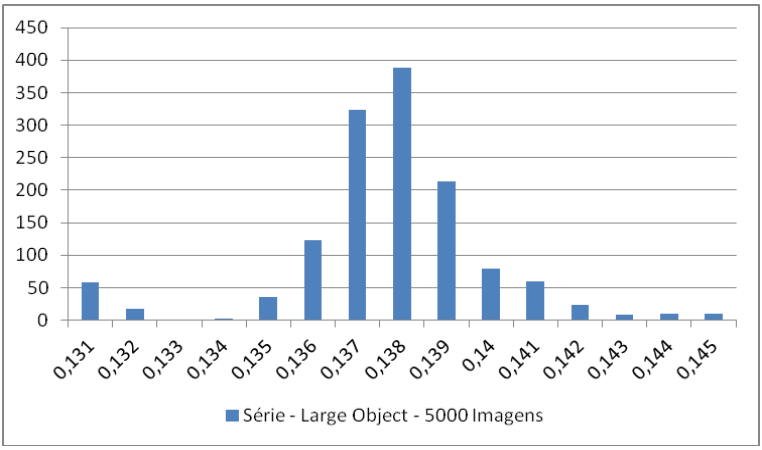
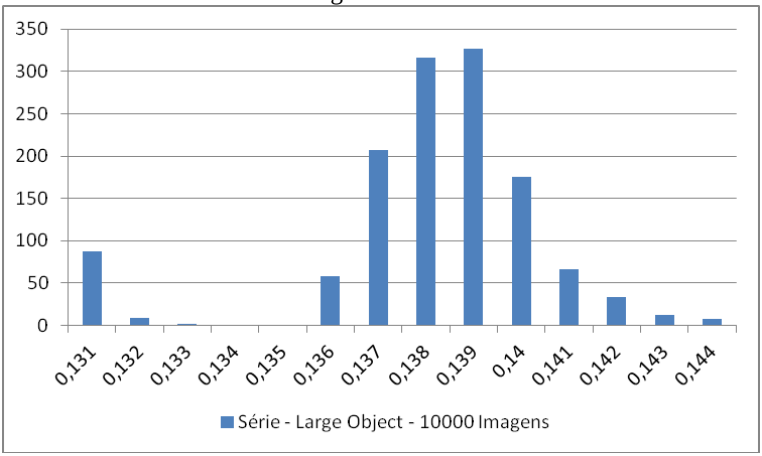


Figura 52 - Distribuição dos resultados de recuperação de série do Large Object com base de dados contendo 10000 Imagens



2. HDF5

Entre as Figura 53 e Figura 60 encontram-se os histograma, que apresentam uma distribuição normal, dos resultados de recuperação de estudo e série armazenados em HDF5.

2.1 Estudo

Figura 53 - Distribuição dos resultados de recuperação de estudo em HDF5 de com base de dados contendo 1000 Imagens

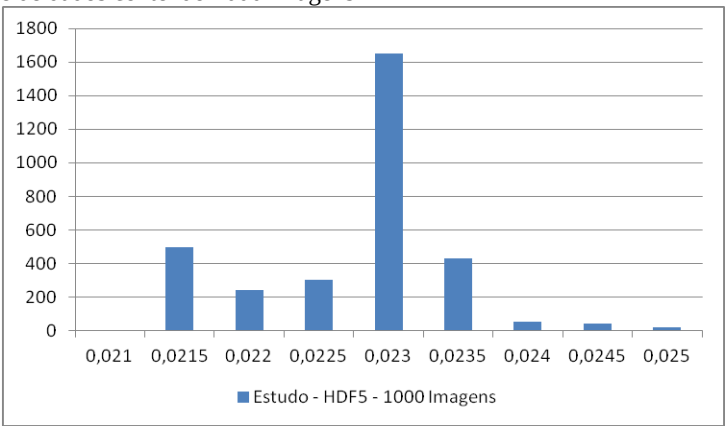


Figura 54 - Distribuição dos resultados de recuperação de estudo em HDF5 de com base de dados contendo 2500 Imagens

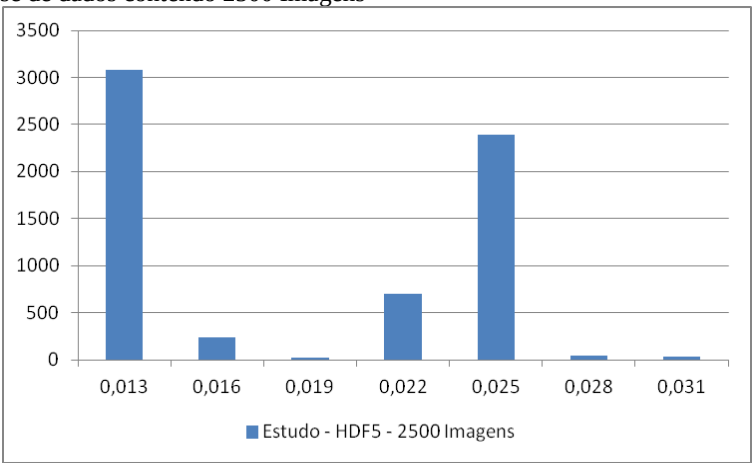


Figura 55 - Distribuição dos resultados de recuperação de estudo em HDF5 de com base de dados contendo 5000 Imagens

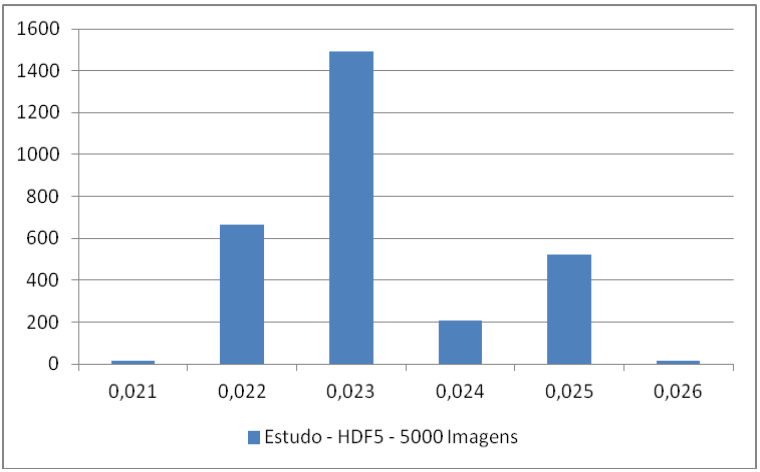
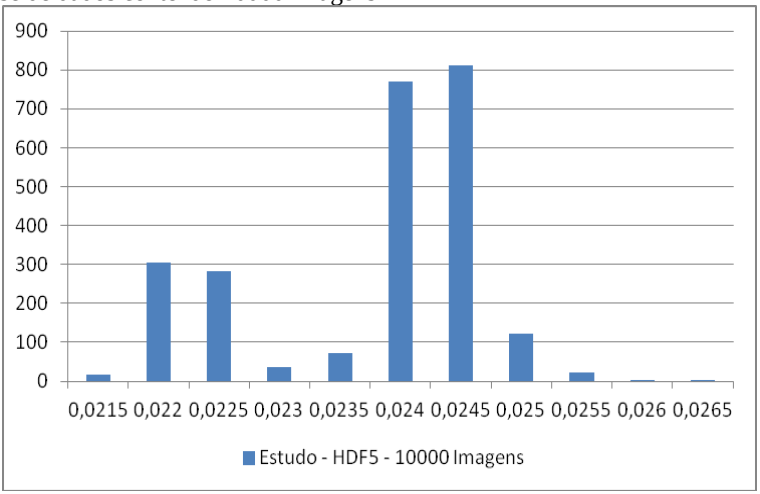


Figura 56 - Distribuição dos resultados de recuperação de estudo em HDF5 de com base de dados contendo 10000 Imagens



2.2 Série

Figura 57 - Distribuição dos resultados de recuperação de série em HDF5 de com base de dados contendo 1000 Imagens

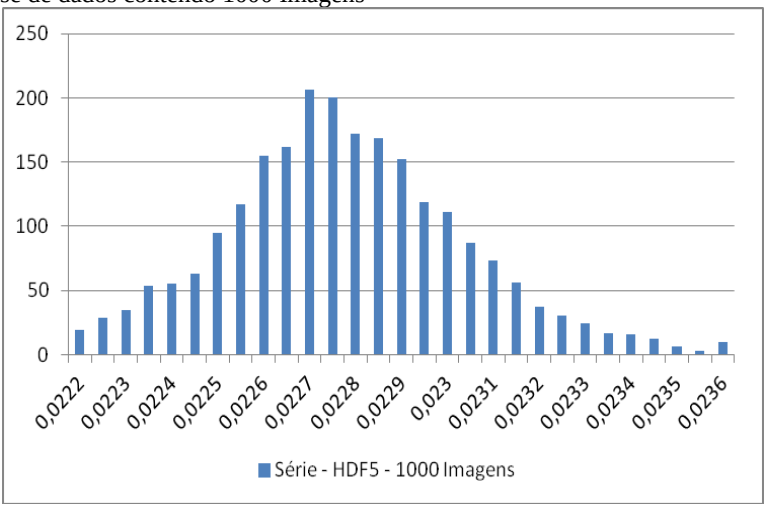


Figura 58 - Distribuição dos resultados de recuperação de série em HDF5 de com base de dados contendo 2500 Imagens

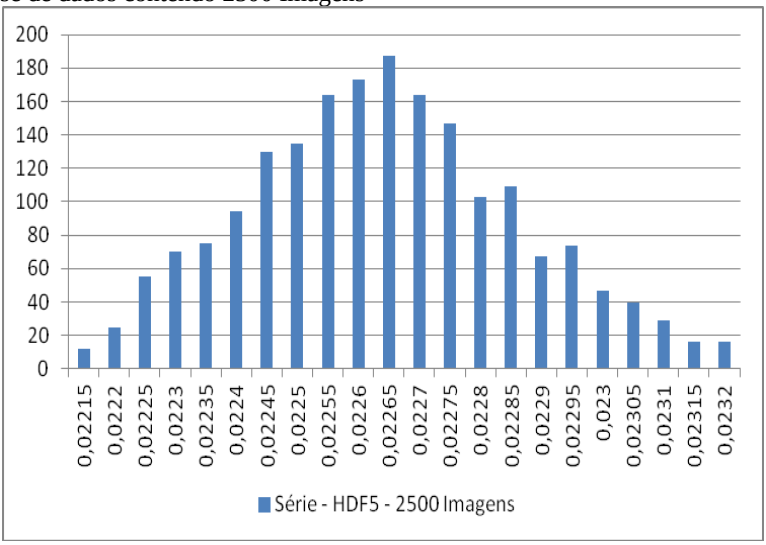


Figura 59 - Distribuição dos resultados de recuperação de série em HDF5 de com base de dados contendo 5000 Imagens

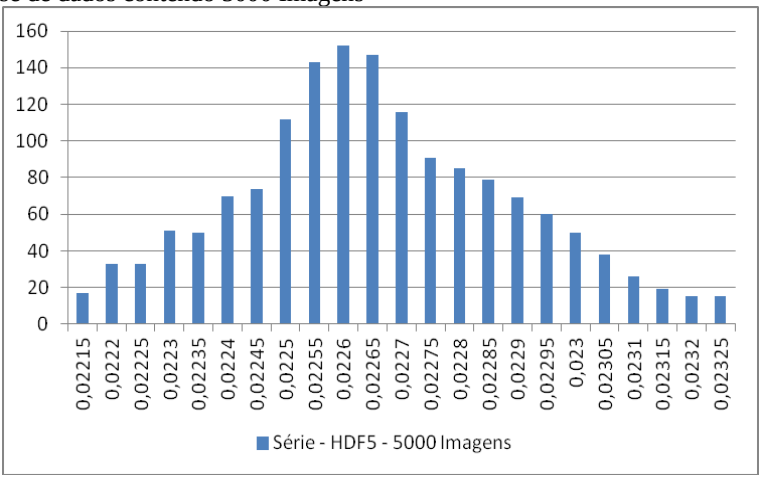
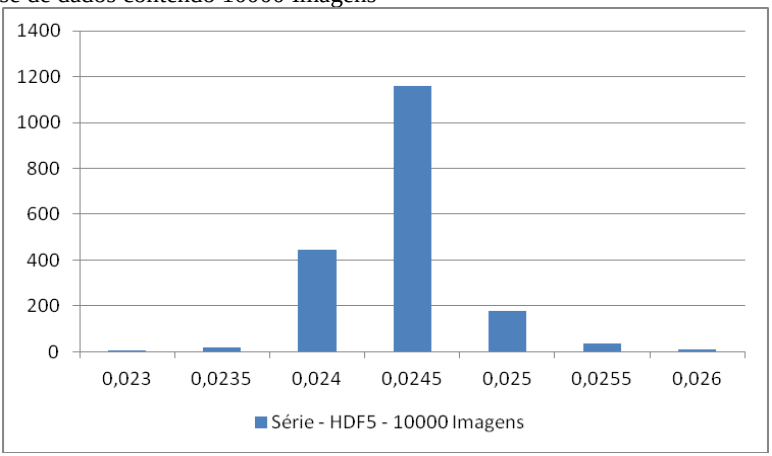


Figura 60 - Distribuição dos resultados de recuperação de série em HDF5 de com base de dados contendo 10000 Imagens



APÊNDICE D - Análise Estatística e Distribuição dos Resultados de Consulta

O resultado Shapiro Wilk pode ser interpretado da seguinte maneira: se a variável W é relativamente pequeno a hipótese de normalidade da distribuição pode ser rejeitada. O mesmo acontece se o *p-value* for menor que o valor de significância 0,05. Na Tabela 8, as células escuras mostram as buscas que apresentam resultados que se enquadram na distribuição normal.

Tabela 8 - Resultado da aplicação do teste Shapiro Wilk para verificação de normalidade da distribuição

		CLucene		*PostgreSQL (Tabela)*	
		W	p-value	W	p-value
1000	Lista de UID	0.8532	0.002513	0.7504	0.00005048
	Intervalo	0.9715	0.7036	0.8048	0.0003534
	Valor Único	0.7713	0.0001036	0.7508	0.00005124
	Universal	0.9608	0.4548	0.7568	0.00006271
	Wildcard	0.7714	0.0001039	0.6065	0.0000007315
2500	Lista de UID	0.8596	0.003315	0.694	0.000008412
	Intervalo	0.9808	0.9103	0.5943	0.0000005336
	Valor Único	0.8474	0.00196	0.6732	0.000004551
	Universal	0.9174	0.0513	0.8646	0.004125
	Wildcard	0.9337	0.118	0.7375	0.00003292
5000	Lista de UID	0.9668	0.5894	0.8929	0.01523
	Intervalo	0.9706	0.6807	0.736	0.00003136
	Valor Único	0.799	0.0002834	0.7395	0.00003512
	Universal	0.9249	0.07486	0.8087	0.0004109
	Wildcard	0.8591	0.003247	0.5115	0.00000007307
10000	Lista de UID	0.7431	0.00003957	0.7314	0.00002701
	Intervalo	0.928	0.08794	0.6069	0.0000007379
	Valor Único	0.5539	0.0000001964	0.6424	0.000001907
	Universal	0.9516	0.2932	0.6893	0.009062
	Wildcard	0.7964	0.0002576	0.8192	0.0006174