MAYCO LUCIAN

IMPLEMENTAÇÃO DE UMA BIBLIOTECA DIGITAL PARA UENP

Trabalho de conclusão de curso apresentado à Universidade Estadual do Norte do Paraná – *campus* Luiz Meneghel – como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Me José Reinaldo Merlin

Bandeirantes

2012

MAYCO LUCIAN

IMPLEMENTAÇÃO DE UMA BIBLIOTECA DIGITAL PARA UENP

Trabalho de conclusão de curso apresentado à Universidade Estadual do Norte do Paraná – *campus* Luiz Meneghel – como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

COMISSÃO EXAMINADORA

_____
Prof. Me José Reinaldo Merlin
UENP – *Campus* Luiz Meneghel

_____
Profa. Me Cristiane Y. Hirabara de Castro
UENP – *Campus* Luiz Meneghel

_____
Prof. Me Carlos Eduardo Ribeiro
UENP – *Campus* Luiz Meneghel

Bandeirantes, __ de_____ de 2012.

Dedico este trabalho a Deus, pois sem Ele nada teria conseguido.

AGRADECIMENTOS

Agradeço a Deus e a Jesus por tudo.

Aos professores da UENP em especial ao meu orientador José Merlin pelas dicas e esclarecimentos, ao meu professor e amigo Welligton Della Mura pela ajuda prestada e confiança em mim depositada, ao professor Estevan Braz Brandt Costa que me ajudou esclarecendo algumas dúvidas e ao professor Christian Bussmann por materiais cedidos para pesquisas.

À minha família por me dar forças e apoio para buscar conhecimentos. À minha namorada Thainã que soube entender a minha ausência nos momentos de pesquisas abrindo mão de muitos momentos de lazer e por ter me dado forças nos momentos de fraqueza. Seu cuidado e carinho foram fundamentais para o sucesso dessa pesquisa.

Aos amigos de serviço em especial a Luana Vizotto que contribuiu com seu conhecimento em língua portuguesa.

Aos alunos da XV Turma de Sistemas de Informação (Marcelão, Marcelo Duarte, Vanessa, Elicreia, Alessando Luz, Ivan, Orestes, Gustavo Soares, Walter Secco, Thiago Orlandini, Bruno Preto, Ruhan, Luiz Paulo e Henrique Coppi), pelos momentos de agonia, trabalho e descontração que passamos ao longo de nossa jornada.

"O medo faz parte da vida da gente. Algumas pessoas não sabem como enfrentá-lo. Outras, acho que estou entre elas, aprendem a conviver com ele e o encaram não de forma negativa, mas como um sentimento de auto preservação." (Ayrton Senna).

RESUMO

Com o aumento das produções científicas geradas pelos alunos da UENP, há a necessidade de uma ferramenta para gerenciar e organizar tais produções. Uma das formas de se organizar documentos é a Biblioteca Digital. Para a construção de uma Biblioteca Digital, é necessário utilizar ferramentas de indexação, para otimizar a busca. A biblioteca Lucene é uma ferramenta livre que oferece recursos de indexação e busca, adequados para sistemas de biblioteca digital. Para que o sistema de busca seja efetivo é necessário o uso de metadados e por este motivo foi estudado o *Dublin Core* e o *MARC.* Para testar a indexação usou-se o Luke, uma biblioteca de testes do Lucene. Este trabalho apresenta a biblioteca digital desenvolvida para a UENP.

Palavras-chave: Lucene. Metadados. Biblioteca Digital. Indexação.

ABSTRACT

With the increase of scientific production generated by the students of UENP, there is a need for a tool to manage and organize such production. One way to organize documents is the Digital Library. For the construction of a Digital Library, is necessary to use indexing tools, to optimize the search. The Lucene library is a free tool that provides indexing and searching capabilities, suitable for digital library systems. For the search system to be effective requires the use of metadata and for this reason was studied the Dublin Core and MARC. To test the index used to Luke, a library of tests Lucene. This paper presents a digital library developed for the UENP.

Keywords: Lucene. Metadata. Digital Library. Indexing.

LISTA DE SIGLAS

API - Application Programming Interface

ASCII - American Standard Code for Information Interchange

ASF – Apache Software Foundation

CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior

CLM – Campus Luiz Meneghel

DAO – Data Access Object

DC – Dublin Core

DER – Diagrama Entidade Relacionamento

HTML - HyperText Markup Language

IDE - Integrated Development Environment

ISO – International Organization for Standardization

JPA - Java Persistence API

JSF – Java Server Faces

MVC – Model View Controller

PDF – Portable Document Format

RI – Recuperação de Informação

SGBD - Sistema de Gerenciamento de Banco de Dados

TCC – Trabalho Conclusão de Curso

UEL – Universidade Estadual de Londrina

UEM – Universidade Estadual de Maringá

UENP – Universidade Estadual do Norte do Paraná

MARC – Machine Readable Cataloging

OCLC – Online Computer Library Center

RDF – Resource Description Framework

LISTA DE FIGURAS

LISTA DE QUADROS

SUMÁRIO

1 INTRODUÇÃO

A quantidade de informações produzidas pelas organizações vem acarretando o problema de armazenamento e acesso a elas. Esse problema nasceu juntamente com as publicações impressas, surgindo à necessidade de armazenar e recuperar as informações já produzidas. O problema da gestão de documentos impressos afeta a maioria das organizações, especialmente as instituições de ensino, nas quais quase a totalidade da produção é materializada na forma de documentos escritos.

A Universidade Estadual do Norte do Paraná (UENP), por meio de seus professores e alunos tem gerado uma grande quantidade de documentos impressos, como monografias, dissertações e outras produções científicas.

Contudo, há uma grande deficiência na organização e no acesso às produções científicas. O espaço para guardar todos os trabalhos acaba sendo insuficiente e o acesso a eles pode ser dificultado, especialmente pelo fato de a UENP ser uma instituição multi campi.

Uma solução para o problema é a implantação de uma biblioteca digital. A biblioteca digital é uma ferramenta que fornece, por meio da internet, acesso a documentos armazenados em formato digital. Além de fornecer um excelente meio para organização de documentos, permite acesso *on-line* pelos usuários, tanto da comunidade interna quanto pelo público em geral. Por meio desse serviço, os trabalhos produzidos podem ser depositados em um formato padronizado, como *Portable Document Format* (*PDF*), não sendo necessário o deslocamento do usuário até a biblioteca onde se encontra o documento impresso.

Para tornar mais eficiente o armazenamento e recuperação de informações, foram desenvolvidas técnicas de organização e pesquisa. Uma das ferramentas mais importantes para isso se chama indexação, que é a elaboração de índices para documentos. Um índice é uma coleção de termos que indicam o local onde a informação desejada esta guardada.

O presente trabalho apresenta uma proposta para o desenvolvimento e implantação de uma biblioteca digital para a UENP.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo deste trabalho é implementar o sistema de biblioteca digital para a UENP, para armazenamento e recuperação dos diversos trabalhos desenvolvidos pela comunidade universitária e acesso a eles por consulta pública *on-line*.

1.2 JUSTIFICATIVA

A idéia de implantar uma biblioteca digital na UENP surgiu da necessidade de organizar as publicações e ao mesmo tempo tornar o acesso mais prático. A quantidade de publicações impressas geradas durante o ano letivo na Universidade exige muito espaço para armazená-las. O objetivo da biblioteca digital é transformar essas publicações em documentos digitais e torná-las disponíveis na internet para consulta, a qualquer hora e de qualquer lugar.

Além disso, existe uma regulamentação, especificamente uma portaria da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES)[1], que torna obrigatória a existência da biblioteca digital para instituições que mantém programas de mestrado e doutorado. Segundo a portaria:

> Art. 1º Para fins do acompanhamento e avaliação destinados à renovação periódica do reconhecimento, os programas de mestrado e doutorado deverão instalar e manter, até 31 de dezembro de 2006, arquivos digitais, acessíveis ao público por meio da Internet, para divulgação das dissertações e teses de final de curso.
>
> §1º Os programas de pós-graduação exigirão dos pós-graduandos, a entrega de teses e dissertações em formato eletrônico, simultânea à apresentação em papel, para atender ao disposto neste artigo.
>
> §2º Os arquivos digitais disponibilizarão obrigatoriamente as teses e dissertações defendidas a partir de março de 2006.

[1] CAPES é o órgão do Ministério da Educação e Cultura que regulamenta os programas de pós-graduação *stricto sensu* no Brasil.

> §3º A publicidade objeto deste artigo poderá ser assegurada mediante publicação através de sítio digital indicado pela CAPES, quando o programa não dispuser de sítio próprio.

1.3 MÉTODO

Este trabalho foi desenvolvido, inicialmente, por meio de pesquisa bibliográfica sobre bibliotecas digitais, mecanismos de indexação e busca.

Na etapa seguinte, foram analisadas, bibliotecas digitais de outras universidades, tais como Universidade Estadual de Londrina (UEL), Universidade Estadual de Maringá (UEM) e Universidade Estadual de Campinas (UNICAMP), entre outras.

As melhores tecnologias acessíveis para a construção de biblioteca digital foram selecionadas e aplicadas na elaboração de uma biblioteca digital para a UENP.

1.4 ORGANIZAÇÃO DO TRABALHO

O trabalho esta dividido em cinco seções: Introdução, Fundamentação Teórica, Sistema Proposto, Desenvolvimento do Sistema e Considerações Finais. Na Seção 2 são mostrados conceitos sobre biblioteca digital, metadados (padrão *Dublin Core* e padrão *MARC*), motores de busca e sobre a biblioteca Lucene, que foi utilizada no trabalho. Na Seção 3 é apresentada a proposta preliminar do sistema e os testes iniciais com as ferramentas. Na Seção 4, é detalhado o desenvolvimento do sistema, as funcionalidades e requisitos. Por último na seção 5, é feita a conclusão do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

No presente capítulo são abordados, na Subseção 2.1, conceitos biblioteca digital; na Subseção 2.2, metadados, principais conceitos e padrões *Dublin Core* e *MARC*; na Subseção 2.3, motor de busca e a biblioteca Lucene (conceitos e etapas).

2.1 BIBLIOTECA DIGITAL

De acordo com Nunes (2005), com o surgimento da escrita veio também a necessidade de um local para armazenar o material escrito, de forma organizada e segura. Esta necessidade deu origem às bibliotecas tradicionais existentes ainda nos dias atuais. Nessas bibliotecas, são guardados livros, revistas e outras publicações, a maioria impressa em papel.

Com o avanço das tecnologias, novas fontes de informações vêm surgindo, sobrecarregando as pessoas com informações. A principal desvantagem em tanta informação gerada é o difícil processo de armazená-las (espaço) e recuperá-las com facilidade. A biblioteca digital surgiu como uma ferramenta para solucionar estes problemas.

Bibliotecas digitais também conhecidas como "bibliotecas virtuais" ou "bibliotecas eletrônicas", são aplicações web que disponibilizam, por meio da internet, documentos eletrônicos de diversas áreas.

Segundo Assoreira e Mourão (2001), a biblioteca pode facilitar a pesquisa à distância, a disseminação de resultados e, talvez o mais importante, a disponibilidade de documentos para diversas áreas (exatas, letras, humanas, entre outras.).

Entre as vantagens de uma biblioteca digital, destaca-se sua capacidade de armazenar as informações e recuperá-las sem precisar de espaço físico e podendo ser acessado de qualquer lugar em qualquer momento (24 horas). Segundo Assoreira e Mourão (2001), “[...] é útil dispor de muita informação, desde que se consiga ‘manobrar’ através de instrumentos e ferramentas adequadas, no ‘mar de informação’ que uma biblioteca digital pode disponibilizar”.

Nunes (2005) propõe a divisão da biblioteca digital em 7 etapas. Neste trabalho foi feita uma adaptação da divisão proposta, reduzindo as etapas para 4, sendo elas:

1. Transformação dos documentos
   Essa etapa envolve decidir de qual maneira se irá trabalhar com os documentos (transformação de documentos não-digitais em documentos digitais), bem como decidir qual o formato de indexação na base de pesquisa (*index*).

2. Recuperação da Informação
   Essa etapa envolve processos de armazenamento para garantir a integridade dos documentos eletrônicos, definir a maneira como será feita a recuperação, se bibliográfica (título, autor, data, palavras-chave) ou informação textual (armazenar e recuperar todo o texto), e também a implementação da ferramenta de busca.

3. Definição do meio de compartilhamento
   Nesta fase será discutida a infra estrutura física de comunicação interligando os documentos com a biblioteca. Será feito também o treinamento aos usuários responsáveis pela biblioteca digital tornando a aplicação mais amigável aos usuários.

4. Proteção dos direitos autorais
   Esta fase é o processo que trata sobre como serão preservados os direitos autorais, respeitando o autor e a instituição.

Durante a implementação de uma biblioteca digital, deve-se tomar cuidado em preservar a segurança das obras armazenadas, evitando que ocorram plágios ou alterações. Segundo Brand, Daly e Meyers (2003), informação digital é frágil e pode ser corrompida ou alterada, intencionalmente ou não.

Segundo Assoreira e Mourão (2001), as bibliotecas digitais são um local de colaboração por excelência, pois o número de indivíduos potencialmente interessados

num determinado tópico é muito maior número do que aquele que se encontra numa biblioteca tradicional.

No próximo tópico são abordados metadados e seus principais padrões.

2.2 METADADOS

Os metadados ajudam na busca e recuperação de informação. Em várias fontes de pesquisas o significado de metadados é apresentado como “dados sobre dados”. Metadados de um arquivo podem dar informação (significado) sobre o conteúdo daquele arquivo para ser disponibilizado para outros utilizadores. Autores, títulos, palavras-chaves são exemplos de metadados.

Os metadados são estruturas de informações que descrevem, explicam, localizam ou tornam mais fácil de recuperar, usar ou gerenciar um recurso de informação (BRAND, DALY e MEYERS, 2003).

Metadados são usados por serviços *on-line* em diversas situações, como busca da informação, arquivamento e bibliotecas digitais, para administrar uma grande quantidade de dados.

Uma razão importante para a criação de metadados descritivos é facilitar descoberta de informações relevantes. Além de descoberta de informações, metadados podem ajudar a organizar recursos eletrônicos, facilitar interoperabilidade e de integração de recursos legados, fornecer identificações digitais, apoio e arquivamento e preservação (BRAND, DALY e MEYERS, 2003).

Segundo Rosetto e Nogueria (2002), da mesma forma que um catálogo de biblioteca fornece informação para localizar um livro, um metadado é usado para localizar uma informação (registro), que poderá ser um endereço internet, recursos web, livros, e-mail, eventos, artefatos, conjunto de dados, projetos, organizações.

Existem três tipos principais de metadados de acordo com a Brand, Daly, Meyers (2003):

- Metadados descritivos: descrevem um recurso para fins de descoberta. Exemplo: título, autor, resumo e palavras chaves.

- Metadados estruturais: indicam como objetos compostos são colocados juntos. Exemplo: como as páginas são ordenadas para formar capítulos.
- Metadados administrativos: fornecem informação para ajudar gerenciar um recurso. Exemplo: quando e como o documento foi criado.

Além de descoberta de recursos, metadados podem ajudar a organizar recursos eletrônicos, facilitar interoperabilidade, integração de recursos legado, apoio e arquivamento e preservação (BRAND, DALY e MEYERS, 2003).

Os metadados são classificados por padrões conforme a necessidade de suas aplicações, tais como vídeos, sons, imagens, textos e sites na web. Entre os padrões de metadados mais conhecidos estão:

- *Dublin Core Metadata Element Set* (DC) – usado para documentos eletrônicos;
- *Government Information Locator Service* (GILS) - informações governamentais;
- *Federal Data Geographic Committee* (FGDC) - descrição de dados geo-espaciais;
- *Machine Readeble Card* (MARC) - catalogação bibliográfica, muito usada nas bibliotecas tradicionais;
- *Consortium for the Interchange of Museum Information* (CIMI) - Informações sobre Muses;
- *Spatial Archive and Interchange Format* (SAIF);
- *Meta Content Format* (MCF);
- *Text Enconding Iniciative* (TEI);
- *Electronic Archive description* (EAD); e
- *Resource Description Framework* (RDF).

Contudo, os padrões mais utilizados para bibliotecas são *Dublin Core* e *MARC*, que serão discutidos nos tópicos a seguir.

2.2.1 Dublin Core

No ano de 1995, um grupo de organizações *(Libary of Congress*, universidades, empresas de Tecnologias da Informação e Comunicação, organizações não governamentais) liderado pela *Online Computer Library Center* (OCLC) reuniu-se na cidade de *Dublin*, no estado americano de *Ohio*, a fim de propor uma padronização para informações de arquivos digitais. O conjunto ficou conhecido como *Dublin Core* (DC) e é mantido pela *Dublin Core Metada Initiative* (DCMI).

Segundo Alves e Souza (2007), o DC é um padrão de metadados composto por 15 elementos, planejado para facilitar a descrição de recursos eletrônicos, que podem ser implementados livremente para atender as necessidades de cada usuário e é um formato padrão para efetuar a interoperabilidade.

Interoperabilidade é a maneira de um sistema comunicar de forma transparente com outro sistema seja ele semelhante ou não. Segundo Alves e Souza (2007), interoperabilidade é a capacidade de bases de dados trocarem e compartilharem documentos, consultas e serviços, usando diferentes plataformas de hardware e software, estrutura de dados e interfaces.

No Quadro1 são mostrados os 15 elementos que compõe o padrão DC.

1. Título	O nome dado ao recurso pelo autor ou editor.
2. Criador	Responsável pelo conteúdo intelectual do recurso, o autor.
3. Assunto	O tópico do recurso, também palavras-chave, frases que descrevem o assunto ou conteúdo do recurso.
4. Descrição	Descreve o conteúdo do recurso, como resumo e sumario.
5. Publicador	O responsável em tornar o recurso disponível.
6. Colaborador	Entidade responsável pela contribuição intelectual ao conteúdo do recurso mas de forma secundária.
7. Data	Data associada a um evento ou ciclo de vida do recurso.
8. Tipo	O tipo de recurso, tais como *home page*, romance, poema, documento de trabalho, relatório técnico, ensaio ou dicionário.

9. Formato	Como será representado o recurso, por exemplo, texto, HTML, ASCII, PDF, XLS.
10. Identificador	Referência não-ambígua (localizador) para o recurso dentro de dado contexto.
11. Fonte	Referência a um recurso do qual o presente é derivado.
12. Idioma	A língua em que esta o conteúdo do recurso.
13. Relação	Referência para um recurso relacionado.
14. Cobertura	Extensão ou escopo do conteúdo do recurso; pode ser temporal e espacial.
15. Direitos Autorais	Um aviso de direitos autorais dentro e sobre o recurso.

QUADRO1 – Elementos do Padrão *Dublin Core*

Fonte Adaptado: Alves e Souza, 2007.

No Quadro2 e Quadro3 é demonstrado um exemplo de metadados deste projeto no padrão *Dublin Core*.

Título = "Biblioteca Digital"
Criador = "Lucian, Mayco"
Assunto = "implementação de uma biblioteca digital para a UENP"
Descrição = "serão abordados conceitos e ferramentas para implementação da biblioteca digital"
Publicador = "UENP"
Colaborador = “Merlin, Jose/Della Mura, Wellington”
Data = "2012"
Tipo = "texto"
Formato = "aplicação/pdf"
Liguagem = "pt"

QUADRO2 – Exemplo *Dublin Core*

Fonte Adaptado: Brand, Daly, Meyers (2003).

```
< head profile ="http://dublincore.org">
< title> … </ title>

< link rel ="schema.DC" href ="http://purl.org/dc/elements/1.1/" />
< link rel ="schema.DCTERMS" href ="http://purl.org/dc/terms/" />
< meta name ="DC.Identifier" schema ="DCterms:URI"
content ="http://tutorialsonline.info/Common/DublinCore.html" />
< meta name ="DC.Creator" content ="Mayco Lucian" />
< meta name ="DC.Subject" xml:lang ="PT" content ="implementação de uma
biblioteca digital para a UENP" />
< meta name ="DC.Publisher" content ="UENP" />
< meta name ="DC.Contributor" content ="Merlin, Jose/Della Mura,Wellington"
/>
< meta name ="DC.Date" scheme ="ISO8601" content ="2012-06-17" />
< meta name ="DC.Type" content ="text/html" />
< meta name ="DC.Description" xml:lang ="PT"
content ="serão abordados conceitos e ferramentas para implementação da
biblioteca digital." / >
< meta name ="DC.Rights" content ="Copyright 2005, NTI. Todos direitos
reservados." / >
< meta name ="DC.Format" content ="aplicação/pdf" />
< meta name ="DC.Language" scheme ="dcterms:RFC1766" content ="PT" / >
```

QUADRO3 – Exemplo *Dublin Core XML*

2.2.2 MARC

MAchine-Readable Cataloging (MARC), segundo Alves e Souza (2007), é um conjunto de padrões para identificar, armazenar e comunicar informações bibliográficas em formato legível por máquina.

Segundo Maranhão e Mendonça (2010) um registro MARC é composto por três elementos: estrutura do registro (implementação de padrões internacionais *ANSI* Z39.2 e *ISO* 2709), indicação de conteúdo (são códigos e convenções estabelecidos para

identificar e caracterizar os dados dentro do registro e permitir sua manipulação) e o conteúdo propriamente dito, os conteúdos dos dados que compõe um registro MARC geralmente são definidos por padrões externos ao formato.

2.2.3 MARC x DUBLIN CORE

Em reunião realizada com a bibliotecária da UENP para levantamento de requisitos foi decidido que o DC seria adotado como padrão de metadados. A opção pelo DC foi pelo fato do MARC não se enquadrar nos padrões da biblioteca digital. Outro fator que contribuiu para escolha foi a biblioteca da UENP ser organizada de acordo com o padrão DC sendo que este se encaixa perfeitamente em uma Biblioteca Digital.

2.3 MOTOR DE BUSCA

Motores de busca são sistemas cujo objetivo é facilitar a busca de informações, sendo que, para isso, indexam documentos. O objetivo em indexar dados é torná-los disponíveis de maneira rápida e direta para aqueles que desejam fazer uso desses arquivos, poupando assim tempo e trabalho desnecessários para a localização dos mesmos (FERREIRA, 2010).

O funcionamento do motor de busca dá-se da seguinte maneira: ao pesquisar um arquivo informa-se alguma frase ou alguma palavra relacionada ao assunto de interesse. Segundo Ferreira (2010) uma abordagem para pesquisar este arquivo seria a varredura sequencial no arquivo em busca da palavra desejada, porém esta abordagem é ineficiente tanto em confiabilidade quanto no custo de desempenho para a execução da pesquisa. A utilização de índices na recuperação de informações auxilia a reduzir o problema dessa abordagem.

Para uma melhor compreensão dos motores de busca o próximo tópico irá demonstrar o Lucene, um motor de busca da *Apache*.

2.3.1 Lucene

Lucene é uma biblioteca que auxilia na Recuperação de Informação (RI). RI refere-se ao processo de busca de documentos, informações dentro de documentos ou metadados sobre documentos.

Segunda Ferreira (2010), Lucene é uma *Application Programming Interface* (API) para indexação de arquivos textos desenvolvida em Java pela *Apache Software Foundation* (ASF)[2].

Lucene foi criado por Doug Cutting no ano de 2000 e disponibilizado como software livre. A partir de 2001 passou a fazer parte da família ASF. Vários projetos utilizam o Lucene como mecanismo de busca web, como o Nutch e o IDE Eclipse (*help online*), Hatcher, Gospodnetic e McCandless (2009).

Segundo Ferreira (2010), Lucene tem como característica ser um indexador de altíssimo desempenho, não importando a origem dos dados, seu formato ou a linguagem na qual foram escritas. O único requisito para a indexação dos arquivos é que estes possam ser convertidos em um arquivo no formato texto (mais detalhes sobre indexação no tópico 2.3.1.1).

Uma das principais vantagens em utilizar o Lucene para indexar documentos é que não é necessário um conhecimento aprofundado sobre indexação e recuperação de informação.

É importante compreender o que uma aplicação de pesquisa precisa para atender as necessidades, levando em consideração que existem diversos tipos de bibliotecas digitais cada qual com uma finalidade diferente. Lucene pode manipular separadamente cada parte da indexação. Um equívoco comum é pensar que Lucene é um aplicativo de pesquisa inteira, quando na verdade ele é simplesmente a indexação do núcleo e busca de componentes (HATCHER; GOSPODNETIC; MCCANDLESS, 2009), como mostrado na Figura2.1.

A indexação é o processo que a Lucene realiza para fazer um *upload* do arquivo para seu *índex* (repositório de dados). Para realizar tal processo é necessário

[2] ASF é uma organização sem fins lucrativos criada para suportar os projetos *Apache*.

extrair os dados, por exemplo, pegar o arquivo em formato PDF e transformá-lo em texto. Lucene não é responsável pela extração de documentos, caberá ao programador implementar essa etapa. Contudo, há ferramentas que realizam este tipo de serviço, como *PDFBox,* que é uma API que extrai o conteúdo textual de documentos PDF (FERREIRA, 2010).

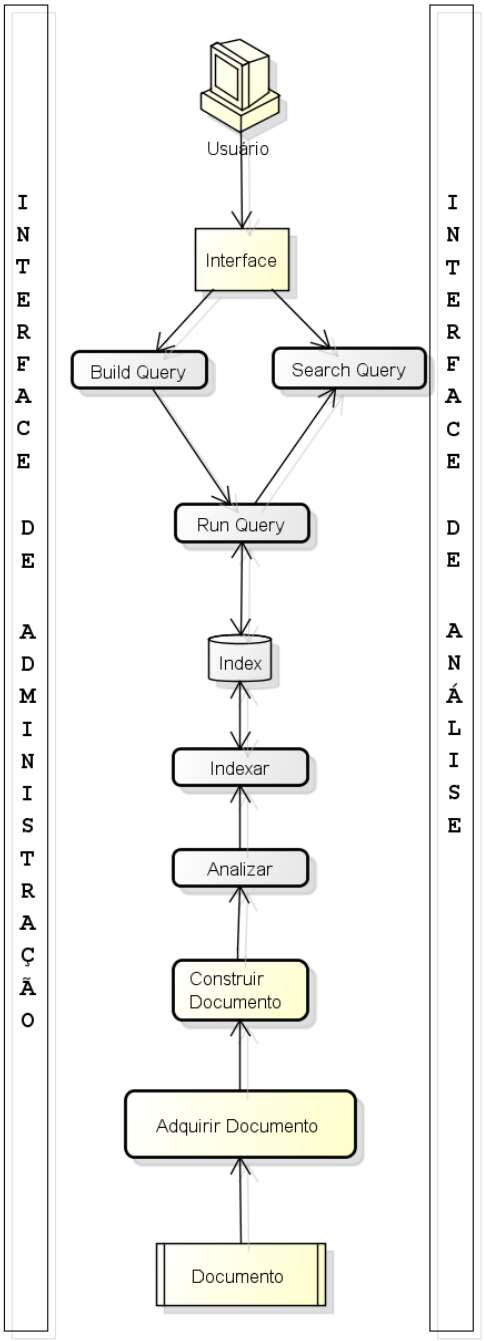

FIGURA2.1 Componentes de uma Aplicação Lucene.
Fonte Adaptado: Hatcher, Gospodnetic, McCandless (2009).

Na Figura 2.1 são mostrados os componentes de uma biblioteca digital. Os componentes em cinza são os componentes do Lucene os demais devem ser implementados.

Hatcher, Gospodnetic e McCandless (2009) explicam os componentes mostrados na Figura2.1 da seguinte maneira:

Componentes de Indexação

- **Adquirir Documento:** Nesta etapa será reunido todo o conteúdo que precisa ser indexado (exceto imagens), para isso usam-se ferramentas de rastreamento. Ferramentas de rastreamento como o próprio nome diz, irão rastrear a informação no momento da busca. Como já mencionado anteriormente, Lucene é apenas uma biblioteca de pesquisa não fornecendo funções para realizar esta atividade. Existem duas maneiras pelo qual pode ser feito: construir um aplicativo de rastreamento ou usar algum rastreador *open-source*.
- **Construir Documento:** Nesta etapa o documento será convertido em texto simples caso não esteja. Os documentos devem ser extraídos para texto simples, para então o Lucene criar um documento para ser analisado. O documento será dividido em vários campos como texto, autor, resumo, entre outros. Tudo isso deve ser implementado pelo programador, pois o Lucene não oferece este tipo de suporte. Nesta etapa, pode-se usar o padrão de metadados DC e ferramentas como *PDFBox*, para converter o conteúdo em texto.
- **Analisar:** Nesta etapa o texto é tratado. Por exemplo: Como são tratadas palavras compostas? Deve-se aplicar a correção ortográfica (se o seu conteúdo em si tem erros de digitação)? Deveria incluir sinônimos, de modo que uma busca por "*laptop*" também retorna produtos mencionando "*notebook*". Depois do texto extraído, o Lucene irá primeiro analisar o texto, um processo que divide os dados textuais em um fluxo de *tokens*, e realiza uma série de operações opcional sobre eles. Por exemplo, para fazer pesquisas *case-sensitive*, usa-se a classe *LowerCaseFilter* Lucene. Normalmente é também desejável remover

todas as palavras de paradas (*stops words*), que são inseridas frequentemente a partir da entrada de uma pesquisa (por exemplo, um, um, o, na, na...). Devido ao fato de que todos os arquivos possuem estas palavras, é necessário criar um analisador (uma classe para tratar as *stops words*), inserindo manualmente quais serão as *stops words* a serem tratadas, pois Lucene tem somente a *StopFilter* para textos em inglês.

- **Indexar:** Assim que o texto for analisado o mesmo estará pronto para ser indexado ao índice. Lucene armazena a entrada em uma estrutura de dados conhecida como um índice invertido. Esta estrutura de dados faz uso eficiente de espaço em disco, permitindo pesquisas rápidas com palavras-chave. O que torna esta estrutura invertida é que ele usa *tokens* extraídas de documentos de entrada como chaves de pesquisa em vez de tratar os documentos como as entidades centrais. Indexação com Lucene se divide em três operações principais: extrair texto de documentos de origem, analisá-lo e salvá-lo ao índice. Na Figura2.2 são apresentadas as 3 operações.

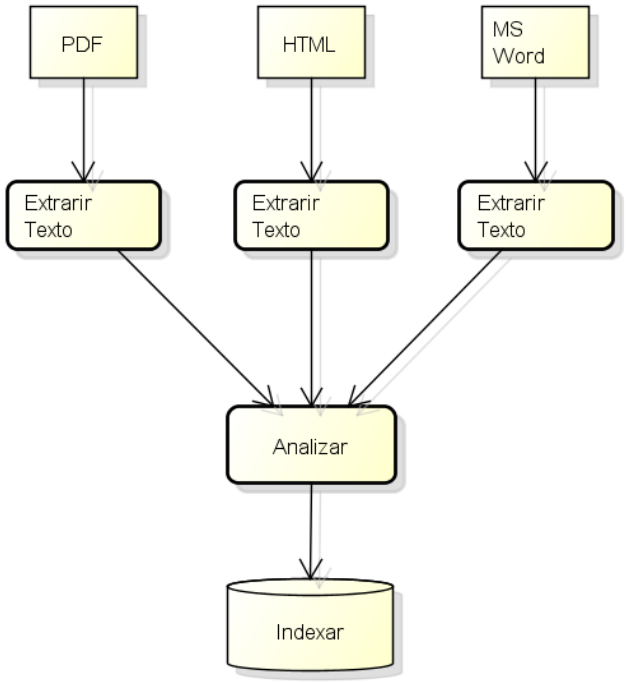

FIGURA 2.2 Três principais operações do Lucene.

Fonte Adaptado: Hatcher, Gospodnetic, McCandless (2009).

A entrada de documentos nos formatos PDF, HTML e MS Word serão transformadas em texto depois analisadas e indexadas.

Componentes de Pesquisa

Pesquisa é o processo de procurar palavras em um índice para localizar documentos em que aparecem. Os componentes de pesquisa são:

- **Interface com o Usuário:** A interface é o que o usuário vê no navegador web quando o mesmo interage com o aplicativo de busca. A interface é a parte mais importante do sistema, de nada adianta o sistema ter um dos melhores motores de busca e a interface ser "poluída", cheia de propagandas, mal organizada, fazendo com que o usuário perca a vontade de utilizar o sistema;
- ***Build Query*:** Objetos de consulta podem ser muito simples ou muito complexos. Lucene fornece um pacote, chamado *QueryParser*, para processar texto do usuário em um objeto de consulta usando uma sintaxe de busca comum;
- ***Search Build*:** Este é o processo de consulta ao índice de pesquisa e recuperação dos documentos correspondentes a consulta, ordenados na ordem de classificação solicitada. Este é o funcionamento muito complexo que ocorre no interior do motor de busca;
- ***Run Query*:** Esta operação controla o *Build Query* e o *Search Build.*
- **Interface de Administração:** Um motor de busca moderno é uma peça complexa de software e tem diversos controles que necessitam de configuração. Usando um rastreador para descobrir o conteúdo, a interface de administração deve deixá-lo definir o *Uniform Resource Locator (URL)* de partida, criar regras para alcance quais sites o rastreador deve visitar ou que tipos de documentos ele deve carregar e definir a velocidade que é permitida a leitura de documentos; e
- **Interface de Análise:** É muitas vezes uma interface baseada a Web, talvez em execução em um servidor separado hospedando um mecanismo de relatório.

Na Figura 2.1 é demonstrada uma aplicação mais comum utilizando o Lucene, nem todas as aplicações têm todas estas etapas, isso irá depender das necessidades (funções) do sistema.

Segundo Hatcher, Gospodnetic e McCandless (2009), nos últimos anos Lucene tornou-se extremamente popular e hoje é a biblioteca de recuperação de informação mais utilizada. Embora seja escrita em Java, graças a sua popularidade, há agora um número de integrações com outras linguagens de programação (C / C++, C #, Ruby, Perl, Python, PHP).

Como foi visto Lucene não é uma aplicação completa e sim uma biblioteca de pesquisa. Utilizando-se Lucene, é possível economizar tempo no desenvolvimento de aplicações, uma vez que já oferece soluções prontas de indexação e busca.

3 SISTEMA PROPOSTO

O desenvolvimento deste trabalho tem por objetivo informatizar os processos de armazenamento e busca de documentos bibliográficos na UENP. A melhor maneira para se fazer isso é a implementação de uma biblioteca digital. Para isso serão utilizadas algumas bibliotecas livres da ASF, *Lucene e o PDFBox*. Nesta seção, será detalhado como será implementada a biblioteca e os testes feitos com as ferramentas.

3.1 USUÁRIOS E FUNÇÕES

A biblioteca digital da UENP armazenará documentos no formato PDF, possibilitando o acesso à distância por meio da internet. Na Figura3.1 são mostrados os usuários e as funções da aplicação.

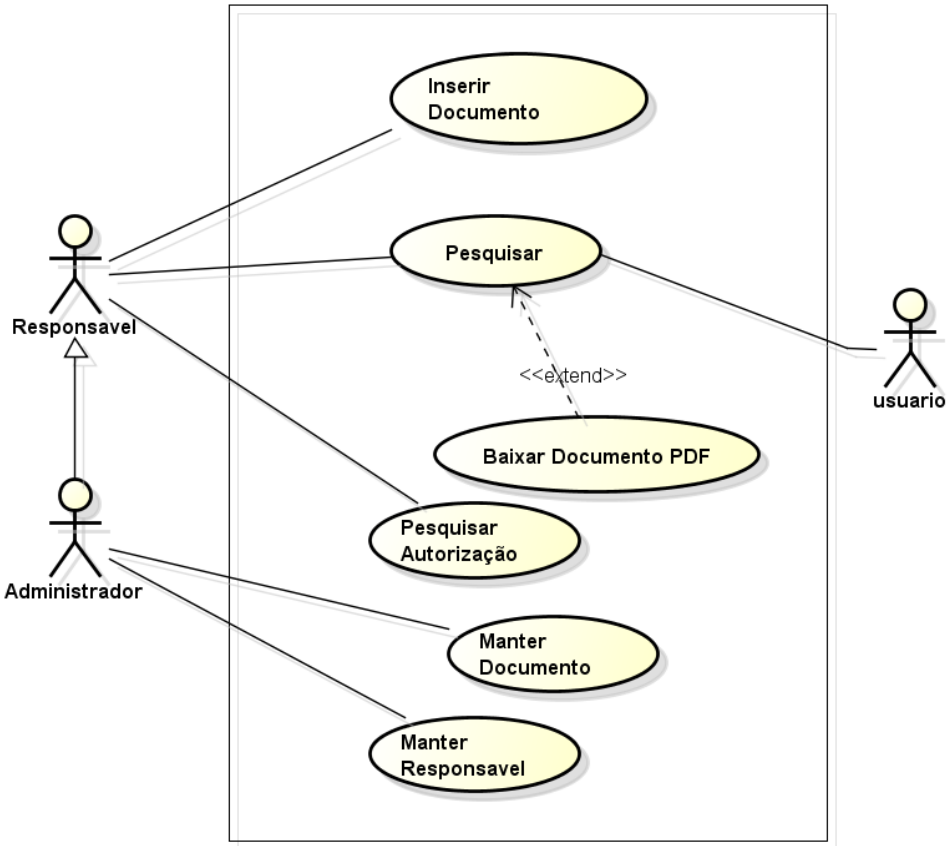

Figura 3.1 – Funções da Biblioteca Digital

O ator “Responsável” será escolhido pela instituição para gerenciar a biblioteca. Suas funções incluem a inserção de documentos, busca por documentos eletrônicos e busca por autorizações assinadas por aluno.

O ator “Usuário” apenas irá pesquisar no sistema podendo fazer o *download* do arquivo. Por exemplo: Usuário busca documentos sobre “Biblioteca Digital”, o sistema retorna informações do tipo descritas abaixo.

Arquivos Encontrados: 1
Local: link para fazer download
Ultima modificação :2011/11/08

Exemplo dados de retorno de uma pesquisa

***Pode conter mais recursos, como resumo, titulo, com a implementação dos metadados.**

O “Administrador” poderá executar qualquer tarefa e é responsável pela manutenção do sistema. No tópico 4.1.4.2 pode-se acompanhar uma descrição mais detalhada dos casos de uso demonstrado na figura3.1.

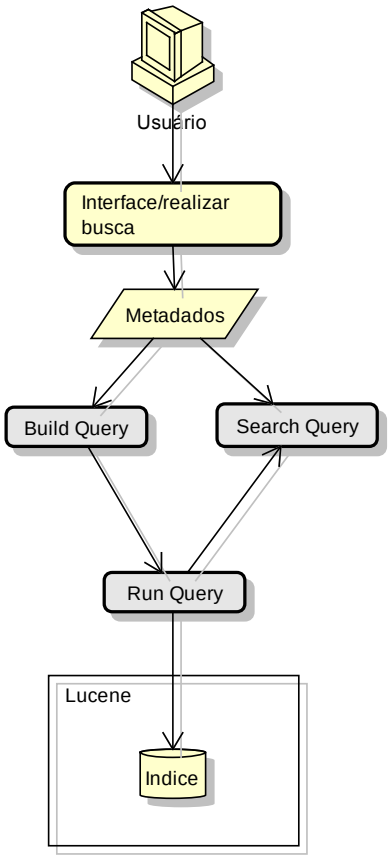

Figura 3.2 – Sistema de Busca

Na Figura 3.2 observa-se o processo de busca. O usuário pode entrar com nome do texto, ou fazer busca por meio de metadados como, nome do autor, título, etc., e o Lucene fará a busca no índice.

O Lucene será responsável por fazer toda a parte de busca, porém o mesmo só trabalha com dados em texto puro. Como os documentos que serão anexados estão em formatos *PDF*, será preciso transformar esses documentos em texto puro e para isso será usado outra biblioteca, conhecida como *PDFBox*. Cabe ao *PDFBox* pegar o texto em *PDF* e transformar em *Txt* para que o Lucene consiga anexar em seu índice e quando necessário fazer a busca.

Após anexar o documento no índice do Lucene o mesmo será salvo no banco de dados, para que se, futuramente, ocorrer algum erro com o índice do Lucene, terá um *backup* no banco de dados, a manutenção se aplica somente na indexação. O processo é ilustrado na Figura 3.3.

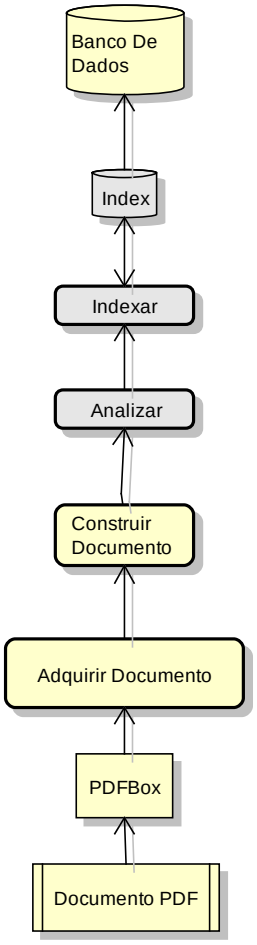

Figura 3.3 Indexando Documento no Lucene

3.1.1 PDFBox

Foram analisadas para este trabalho as bibliotecas o *PDFBox* e *Tika*. A escolha pela utilização do *PDFBox* foi por estar a mais tempo no mercado e ser mais completo e no *Tika* pode ocorrer perda de informações no momento da conversão para *txt*. Esta conclusão foi por meio de testes realizados com essas duas bibliotecas.

O *PDFBox* funciona como ilustrado na Figura 3.4. A extensão do texto é identificada para conferir se termina com *.pdf* (linha 26), cria um documento novo PDF (linha 48) para receber o documento *.pdf* que esta sendo indexado e a partir da classe *PDFtextStripper* (classe já pronta do PDFBOX) é realizado a extração do formato PDF para Txt (linha 50,51).

```
/*23*/ public static String extraiTexto(File arquivo) {
/*24*/   String textoExtraido = "";
/*25*/   try {
/*26*/     if(arquivo.getName().toLowerCase().endsWith(".pdf")) {
/*27*/       textoExtraido = extraiTextoDoPDF(arquivo.getAbsolutePath());
/*28*/     }
```

```
/*45*/    public static String extraiTextoDoPDF(String caminho) {
/*46*/        PDDocument pdfDocument = null;
/*47*/        try {
/*48*/            pdfDocument = PDDocument.load(caminho);
/*49*/            PDFTextStripper stripper = new PDFTextStripper();
/*50*/            String texto = stripper.getText(pdfDocument);
/*51*/            return texto;
/*52*/        }
```

Figura 3.4 – Código de Extração Com PDFBox

3.1.2 Sistema de Indexação

O sistema de indexação trabalha da seguinte maneira: o primeiro passo é criar um novo arquivo no diretório de índice (linha 41, Figura 3.5), após isso na linha 48 (Figura 3.5) será analisado todo o texto, retirando as *stops words* (palavras de paradas, por exemplo como, o, é, no, que, e, em), porém Lucene tem pronto somente a *Analyzer* que trata o idioma inglês. Para suprir esta lacuna será necessário construir uma classe *Analyzer* para o idioma português, inserindo palavra por palavra. Na linha 50 e 51 (Figura 3.5) o *índex* é configurado definindo qual a versão do Lucene que esta sendo usada e passando o analisador de texto junto.

Na linha 55 (Figura 3.5) é chamado outro método, o qual será responsável por usar o *PDFBox* para extrair o conteúdo e indexar, linhas 91, 93 respectivamente, (Figura 3.6).

```
/*38*/ public void indexaArquivosDoDiretorio() {
/*39*/    try {
/*40*/        /* diretorio de arquivos que serão indexados */
/*41*/        File diretorio = new File(diretorioDosIndices);
/*42*/        // apaga o diretorio temporario para novas indexação
/*43*/        apagaIndices(diretorio);
/*44*/        // Abri o indece
/*45*/        Directory d = new SimpleFSDirectory(diretorio);
/*46*/        System.out.println("Indice: " + diretorioDosIndices);
/*47*/        // Analiza o texto
/*48*/        Analyzer analyzer = new StandardAnalyzer(Version.LUCENE_32);
/*49*/        // Cria o Indice
/*50*/        IndexWriterConfig config;
/*51*/        config = new IndexWriterConfig(Version.LUCENE_32,analyzer);
/*52*/        writer = new IndexWriter(d, config);
/*53*/        // inicia a indexação do diretorio
/*54*/        long inicio = System.currentTimeMillis();
/*55*/        indexaArquivosDoDiretorio(new File(diretorioParaIndexar));
/*56*/        writer.optimize();
/*57*/        writer.close();
/*58*/        long fim = System.currentTimeMillis();
/*59*/ System.out.printf("Tempo para indexar: %d seg",((fim - inicio)/ 1000));
/*60*/    } catch (IOException e) {
/*61*/        JOptionPane.showMessageDialog(null, e);
/*62*/      }
/*63*/ }
```

Figura 3.5 – Sistema de Indexação - Método Que Indexa Arquivo no Diretório

```
/*80*/ public void indexaArquivosDoDiretorio(File raiz) {
/*81*/  FilenameFilter filtro = new FilenameFilter() {
/*82*/    public boolean accept(File dir, String name) {
/*83*/        return name.toLowerCase().endsWith("pdf")||
/*84*/               name.toLowerCase().endsWith("txt");
/*85*/        }
/*86*/    };
/*87*/    for (File arquivo : raiz.listFiles(filtro)) {
/*88*/      if (arquivo.isFile()) {
/*89*/        System.out.printf("Indexando arquivo %s, %d KB\n",
/*90*/                    arquivo.getAbsoluteFile(), arquivo.length() / 1000);
/*91*/ String textoExtraido = UtilExtratorDeTexto.extraiTexto(arquivo);
/*93*/        indexaArquivo(arquivo, textoExtraido);
/*94*/ }
```

Figura3.6 – Sistema de Indexação - Método de Indexação com Extração de Dados

E para finalizar, na Figura 3.7 é mostrado a indexação por campos (*fields*), nas linhas 111 a 116. Quanto mais campos de especificação houver, mais lento fica o índice.

```
/*105*/private void indexaArquivo(File arquivo, String textoExtraido) {
/*106*/ // Grava a data em foi indexado
/*107*/ SimpleDateFormat formatador = new SimpleDateFormat("yyyyMMdd");
/*108*/ String ultimaModificacao = formatador.format(arquivo.lastModified());
/*109*/ // preenche os campos que serão indexados
/*110*/ Document documento = new Document();
/*111*/ documento.add(new Field("UltimaModificacao", ultimaModificacao,
/*112*/               Field.Store.YES, Field.Index.NOT_ANALYZED));
/*113*/ documento.add(new Field("Caminho", arquivo.getAbsolutePath(),
/*114*/               Field.Store.YES, Field.Index.NOT_ANALYZED));
/*115*/ documento.add(new Field("Texto", textoExtraido, Field.Store.YES,
/*116*/               Field.Index.ANALYZED));
```

Figura 3.7 – Sistema de Indexação - Fields

3.1.3 Sistema de Busca

Nesta seção são mostrados alguns dos principais comandos de busca do Lucene, é explicado e ilustrado por meio da Figura 3.8.

```
/*32*/ Directory diretorio = new SimpleFSDirectory(new File(diretorioDoIndice));
/*33*/ IndexSearcher buscador = new IndexSearcher(diretorio);
/*34*/ Analyzer analisador = new StandardAnalyzer(Version.LUCENE_32);
/*35*/
/*36*/ QueryParser parser = new QueryParser(Version.LUCENE_32, "Texto",analisador);
/*37*/ Query consulta = parser.parse(parametro);
/*38*/
/*39*/ TopDocs resultado = buscador.search(consulta, 20);
/*40*/ int totalDeOcorrencias = resultado.totalHits;
/*41*/ System.out.println("Arquivos Encontrados:" + totalDeOcorrencias);
/*42*/
/*43*/ for (ScoreDoc sd : resultado.scoreDocs) {
/*44*/  Document documento = buscador.doc(sd.doc);
/*45*/  System.out.println("Local: " + documento.get("Caminho"));
/*46*/  System.out.println("Ultima modificacao: "+
/*47*/                     documento.get("UltimaModificacao"));
/*48*/  System.out.println("Score: " + sd.score);
/*49*/  System.out.println("--------");
/*50*/ }
/*51*/ buscador.close();
/*52*/}
```

Figura 3.8 – Comandos de Busca

Na linha 33 é criado um objeto do tipo *IndexSearcher* que é a classe responsável por fazer a busca de um documento. O *QueryParser,* na linha 36, realiza busca por datas, ao invés de procurar um texto pelo título, pode procurar pela data de

publicação. As linhas 45, 46, 47 e 48 exibem dados de documentos que foram encontrados no índice por meio dos dados fornecidos. Este é um simples teste feito para uma pequena aplicação desktop, a saída de uma busca pela palavra UENP pode ser acompanhada na Figura 3.9.

```
Arquivos Encontrados: 2
Local: C:\Users\Mayco\Documents\conteudo note\7 semestre\teste\FichaInscriProjI.pdf
Ultima modificação: 20110801
Score: 0.109375
--------
Local: C:\Users\Mayco\Documents\conteudo note\7 semestre\teste\monografia.pdf
Ultima modificação: 20110921
Score: 0.03617238
```

Figura 3.9 – Exemplo de Dados Retornados de uma Consulta.

4 DESENVOLVIMENTO DO SISTEMA

Nesta seção apresenta-se uma descrição do sistema Pegasus, isso será demonstrado por meio das modelagens do sistema (diagramas de caso de uso, diagrama de classe e diagrama de pacotes), levantamentos de requisitos e casos de uso.

4.1 ANÁLISE DO SISTEMA

4.1.1 Descrição

Pegasus foi desenvolvido para gerenciar as informações geradas pela comunidade acadêmica UENP, o sistema trata-se de uma Biblioteca Digital (ver seção 2.1), é um sistema *WEB* que utiliza tecnologias mais recentes e ao mesmo tempo *free*[3].

O sistema é desenvolvido na linguagem Java com a tecnologia *Java Service Faces (JSF). JSF* é um *framework* baseado no padrão MVC (*Model View Control* – Modelo Visão e Controle) que auxilia no desenvolvimento web para aplicações Java, ele simplifica o desenvolvimento de interfaces de usuário. A versão JSF utilizada é a 2.0 que é a versão mais recente.

O sistema usa o SGBD (*Sistema de Gerenciamento de Banco de Dados*) PostgreSQL, que é um banco gratuito e robusto, para fazer a persistência é usado o JPA (*Java Persistence API),* é uma API do Java para realizar persistência dos dados. O JPA utiliza um mapeamento conhecido no Java como *Beans*, o *Hibernate* implementa o JPA.

A tela inicial, também chamada de *Home Page* terá *login* aos responsáveis pelo sistema e aberto aos visitantes e alunos. Cabe ao administrador do sistema criar senhas para os responsáveis do sistema, estes responsáveis serão os coordenadores de projetos de cada curso, os responsáveis terão que fazer o *upload* dos trabalhos no sistema para que os alunos e visitantes possam realizar as pesquisas. Os trabalhos só

[3] Tecnologias *free*, são tecnologias onde a utilização não implica o pagamento de licenças de uso ou *royalties*.

poderão ser indexados se estiverem acompanhados de uma autorização do autor liberando para pesquisas e se estiverem no formato *.pfd*.

4.1.2 Declaração do Problema

O problema é:	- Organizar informação gerada pelos alunos; - Armazenamento das informações; e - Acesso as informações.
Afeta:	- Alunos; e - Professores.
Cujo impacto é:	- O processo de acesso as informações ficam mais lentos por ser manual; e - Uma falta de espaço na biblioteca da Universidade.
Uma boa solução seria	Criar um sistema que informatize o processo de armazenar e pesquisar os documentos.
Para	Instituição de ensino - UENP.
Quem	Bibliotecária, alunos, visitantes, professores.

Quadro 4 – Detalhamento do Problema.

4.1.3 Levantamento de Requisitos

Foram levantados os requisitos e os atores responsáveis por executar os requisitos funcionais do sistema.

4.1.3.1 Requisitos

- Manter Documento;
- Inserir Documento;
- Pesquisar;
- Pesquisar Autorização;
- Baixar Documento PDF; e
- Manter Responsável.

4.1.3.2 Atores

- Responsável;
- Administrador; e
- Usuário

4.1.3.3 Necessidades e Funcionalidades

Necessidade	Prioridade	Lançamento Planejado
-Cadastrar/Indexar dados sobre documento	Alta	Meio Projeto.
- Cadastrar dados sobre os responsável do sistema;	Alta	Início do Projeto.
- Pesquisa	Alta	Fim Projeto.
- Excluir/Alterar dados responsável;	Média	Meio Projeto.
- Excluir/Alterar dados documento;	Média	Meio Projeto.
-Download Documento	Alta	Fim Projeto.
- Backup Documentos	Alta	Meio Projeto.

Quadro 5 – Necessidades e Funcionalidades

4.1.4 Relação dos casos de uso

A especificação de casos de uso abaixo é mostrada no Quadro 6.

Caso de Uso	Ator	Descrição
Manter Documento	Administrador	Insere, altera ou exclui documentos da biblioteca.
Inserir Documento	Responsável/Administrador	Insere novas documentos na biblioteca.
Pesquisar	Usuário/Responsável/Administrador	Pesquisa documentos cadastradas no sistema.
Baixar Documento PDF	Usuário/Responsável/Administrador	Baixa documento que foi pesquisada
Manter Usuário	Administrador	Insere, altera ou exclui responsável.
Pesquisar Autorização	Administrador/Responsável	Pesquisa a autorização do autor do documento.

Quadro 6 – Relação Caso de Uso.

4.1.4.1 Descrição dos Requisitos

F1 Inserir Documento	**Oculto ( )**
Descrição: Capturar a informação e salvar no banco.	

F2 Alterar Documento	**Oculto ( )**
Descrição: Realizar a alteração no documento solicitado	

F3 Excluir Documento	**Oculto ( )**
Descrição: Exclui documento quando solicitado	

F4 Indexar Documento	**Oculto (X)**
Descrição: Deve prover um mecanismo persistente de armazenamento (índice).	

F5 Pesquisar	**Oculto ( )**
Descrição: Capturar a informação do item sendo comprado através de métodos do Lucene.	

F6 Download	**Oculto (X)**
Descrição: Fornece o arquivo solicitado.	

F7 Pesquisar Autorização	**Oculto ( )**
Descrição: Pesquisa no índex pelo titulo da obra informada.	

F8 Inserir Usuário	**Oculto ( )**
Descrição: Capturar a informação e salvar no banco.	

F9 Excluir Usuário	**Oculto ( )**
Descrição: Exclui usuário do Banco de Dados quando solicitado.	

F10 Alterar Usuário	**Oculto ( )**
Descrição: Deve ser capaz de alterar informações referentes ao usuário.	

4.1.4.2 Relação dos casos de uso

Caso de Uso: Manter Documento

Inserir Documento

Ator Primário: Responsável
Ator Secundário: Administrador
Descrição: Serão cadastradas as informações do documento e salvas no banco.

Fluxo Básico:
1. O sistema exibe a tela de cadastro de documento (metadados). 2. Responsável/Administrador insere os dados nos campos exigidos. 3. Passa para a tela de Upload. 4. Responsável/Administrador irá enviar o arquivo para a Biblioteca. 5. Exibi uma tela de confirmação de Dados. 6. Responsável/Administrador salva as informações.
Fluxos Alternativos: 2 a. Se algum campo não for preenchido exibi uma mensagem pedindo para preencher todos os campos. b. Vai para o passo 3. 5 dados incorretos a. Sistema volta para o passo 2.

Alterar Documento

Atores: Administrador
Descrição: Caso seja necessário a alteração de dados nos documentos inseridos.
Fluxo Básico: 1. O responsável da Biblioteca Digital solicita alteração de dados ao administrador. 2. Administrador entra no sistema e altera os dados solicitados. 3. Administrador comunica a alteração para o responsável. 4. Responsável confirma os dados alterados. 5. Os dados são alterados
Fluxos Alternativos: 4 dados ainda não conferem a. Responsável avisa novamente o Administrador para enviar os dados. b. Passo 2.

Excluir Documento

Atores: Administrador
Descrição: Exclui o documento da Biblioteca Digital

Fluxo Básico:
1. Direção da Universidade solicita exclusão de um determinado documento. 2. Direção entrega a ordem de exclusão assinada pelo reitor contendo o nome da obra, nome do autor e o motivo da exclusão. 3. Administrador realiza a exclusão dos dados do sistema.

Caso de Uso: Pesquisar

Ator Primário: Aluno/Visitante
Ator Secundário: Administrador e Responsável
Descrição: Realiza a busca de documentos na biblioteca.
Fluxo Básico: 1. É informado na tela de pesquisa um determinado parâmetro (nome autor, título, parte do conteúdo). 2. Sistema realiza a busca através de funções do Lucene. 3. O Sistema retorna uma tela com os resultados. 4. Segue caso de uso baixar documento PDF.
Fluxo Alternativo: 4 a. Nenhum documento de interesse/nenhum documento encontrado. b. É informada uma nova pesquisa. c. Segue passo 2.

Caso de Uso: Baixar Documento PDF

Ator Primário: Aluno/Visitante
Ator Secundário: Administrador/Responsável.
Descrição: Realiza download do documento pesquisado.
Fluxo Básico: 1. Caso de uso Pesquisar. 2. Escolhe-se o arquivo desejado para download.

Caso de Uso: Pesquisar Autorização

Ator Primário: Responsável
Ator Secundário: Administrador
Descrição: Busca uma autorização do documento.
Fluxo Básico: 1. É solicitado a autorização que o autor permitiu a divulgação do documento. 2. Responsável do sistema informa o título do documento do autor. 3. Sistema retorna o documento para fazer o download.

Caso de Uso: Manter Responsável

Inserir Responsável

Atores: Administrador
Descrição: insere um novo responsável no sistema.
Fluxo Básico: 1. Administrador insere os dados do coordenador do curso. 2. Confirma o cadastro e salva no banco de dados. 3. Administrador envia email com o login e senha para o responsável do curso.
Fluxos Alternativos: 1 Dados informados inválido, ex: CPF a. Administrador envia email para o responsável solicitando que envie os dados novamente.

Alterar Responsável

Atores: Administrador
Descrição: Altera dados do responsável.
Fluxo Básico: 1. Responsável solicita alteração de dados. 2. Responsável informa seu login e CPF. 3. Administrador altera os dados solicitados.

Excluir Usuário

Atores: Responsável
Descrição: Exclui o responsável, todo ano muda o responsável pelos projetos do curso, a administrador tem a função de confirma que saiu do cargo e excluir os dados.
Fluxo Básico: 1. Administrador verifica se o houve alteração de cargo devido o novo ano letivo. 2. Administrador informa o login do responsável e exclui todos os dados do responsável.

4.2 MODELAGEM DO SISTEMA

Neste tópico será apresentado algumas funções do sistema por meio de diagramas UML (*Unified Modeling Language*) elaborados com a ferramenta *Astah*.

4.2.1 Diagrama de Classe

Como mencionado anteriormente o sistema foi implementado em Java que é uma linguagem Orientada à Objeto (OO). Em Programação Orientada a Objetos (POO) deve-se pensar em com dividir o projeto, por este motivo é considerado uma boa prática separar o projeto em objetos também conhecido como classes. Objetos são usados para representar uma entidade do "mundo real", composto por nomes, atributos e métodos, exemplos: Grande Área, Área e Sub Área.

Para ilustrar as operações entre as classes será usado o diagrama de classe que é a base para os demais diagramas, por meio dele consegue-se esboçar todos relacionamentos entre as classes.

A Figura 4.1 pode-se acompanhar o Diagrama de Classes do Sistema Pegasus.

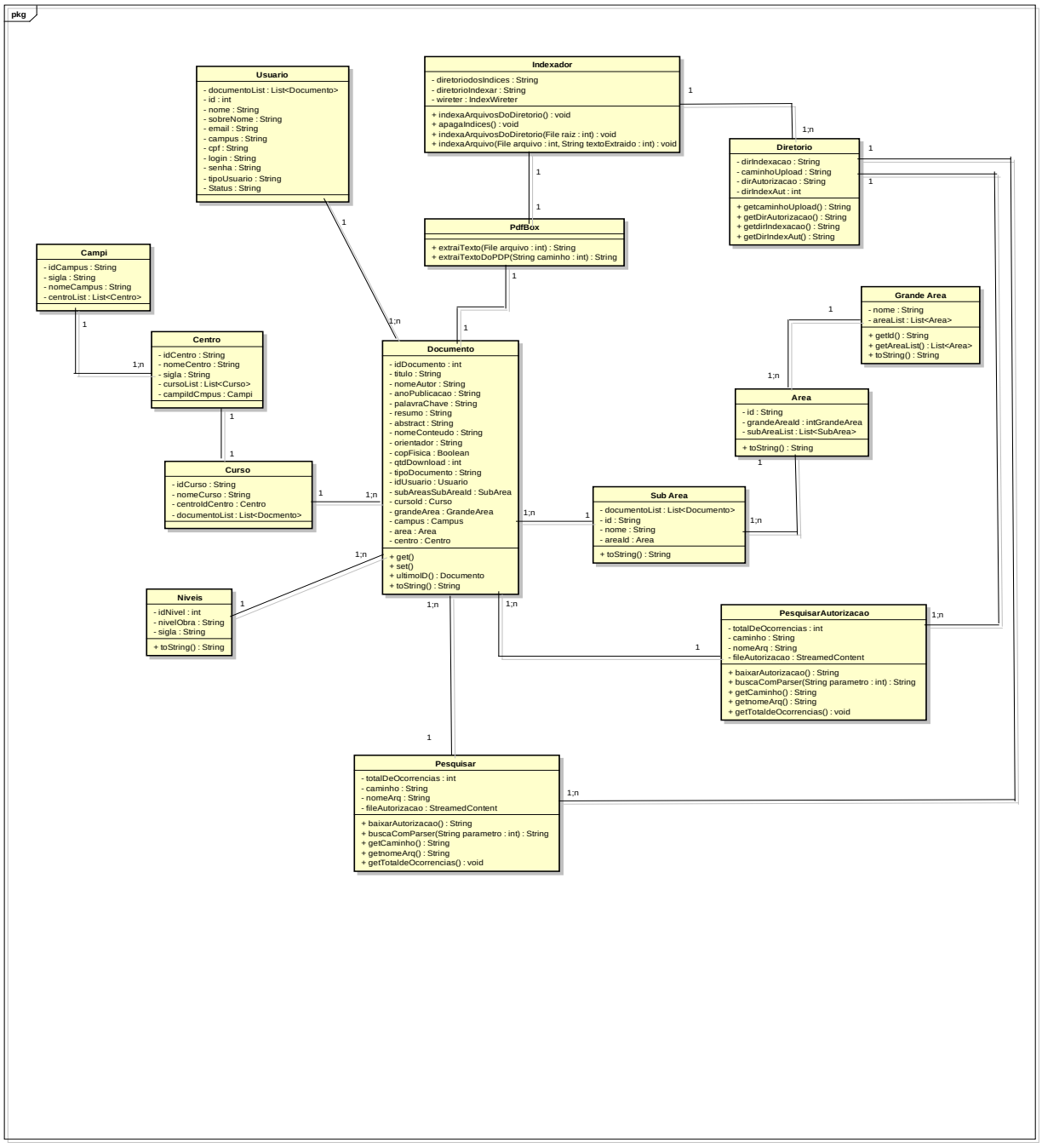

Figura 4.1 Diagrama de Classe – Sistema Pegasus

Por meio do diagrama observa-se que a classe documento contém além de atributos próprios atributos de outras classes que estão relacionadas.

4.2.2 Diagrama de Entidade Relacionamentos

O Diagrama de Entidade Relacionamento (DER) descreve o modelo de dados do sistema, expressa graficamente a estrutura lógica de um banco de dados. Na Figura 4.2 é mostrado o DER do Sistema Pegasus.

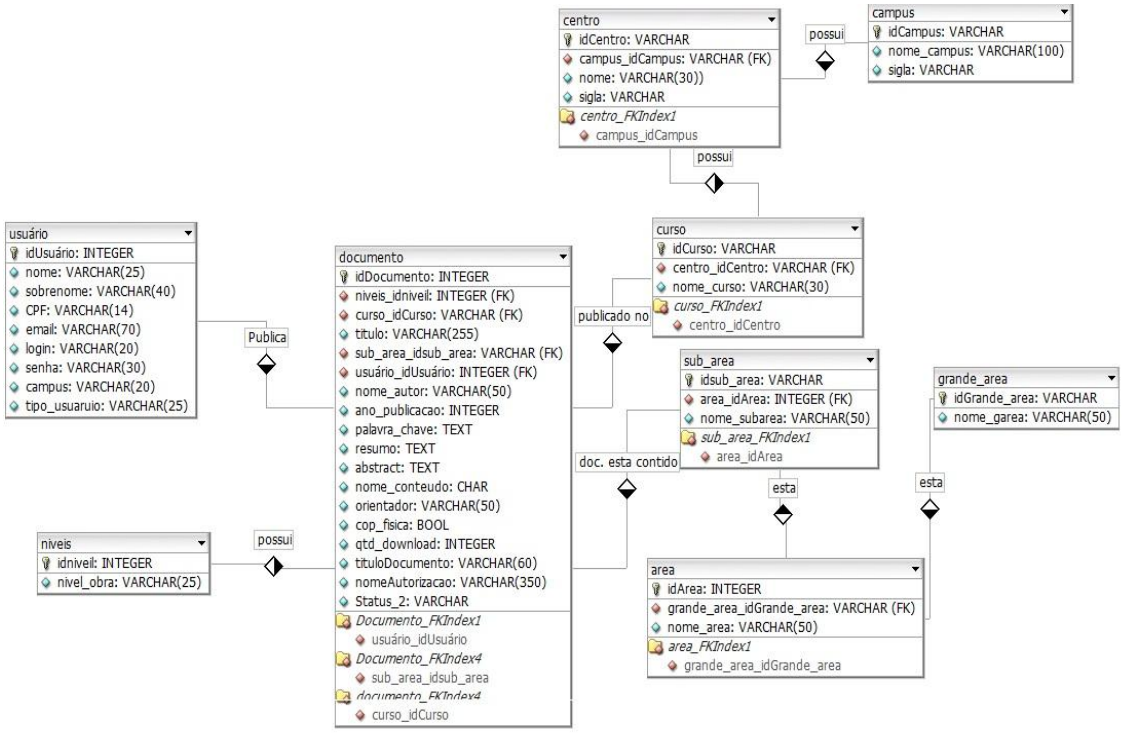

Figura 4.2 DER – Diagrama de Entidade Relacionamento

4.2.3 Diagrama de Pacotes

O diagrama de pacotes descreve os pacotes do sistema mostrando as dependências entre eles. Na Figura 4.3 pode-se observar que para o funcionamento das classes contidas no pacote WEB é preciso das classes contidas no pacote Beans que por sua vez precisa das classes do pacote Model.

As informações do pacote *Model* são salvas no Banco com auxilio das classes do pacote CRUD, o pacote CRUD é responsável por salvar (***C****reate*), acessar informações do banco (***R****ead*), alterar (***U****pdate*) e excluir (***D****elete*).

Para realizar a Indexação do sistema o pacote Lucene comunica-se com o pacote *Index* que contém os diretórios onde se encontram os arquivos indexados e anexados.

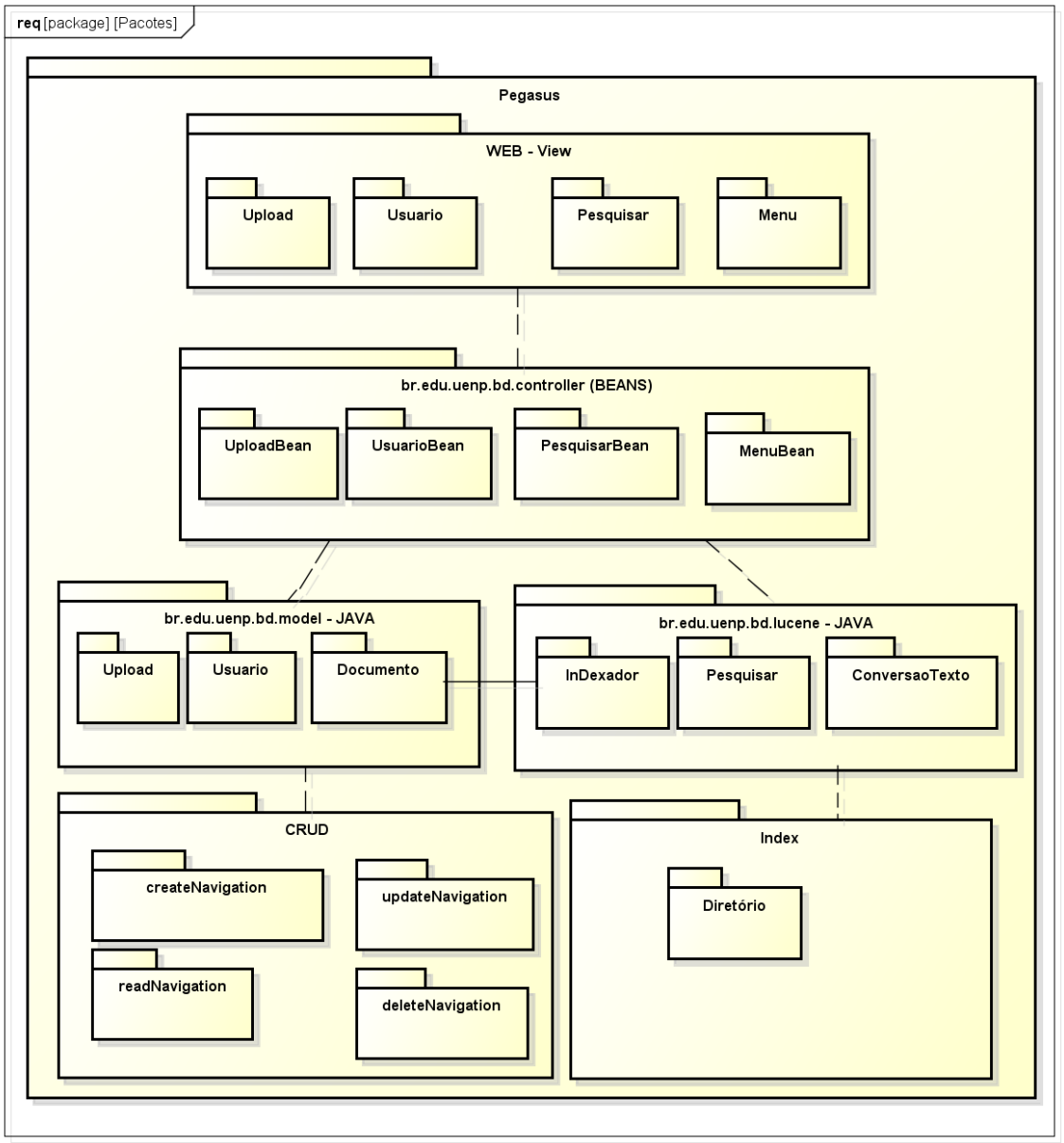

Figura 4.3 Diagrama de Pacotes

4.3 IMPLEMENTAÇÃO DO SISTEMA

4.3.1 Tela Inicial

O sistema desenvolvido é acessado como pegasus.uenp.edu.br. A página inicial é implementada por uma classe *xhtml* chamada *index.xhtml*. A classe *index.xhtml* realiza a interface com o usuário na qual alunos ou visitantes podem ter acesso apenas a pesquisas de teses, enquanto administradores do sistemas terão que realizar o *login*

para ter acesso a outras funções do sistema, como cadastrar usuários e cadastrar teses. Na Figura 4.4 é exibida a página inicial do sistema.

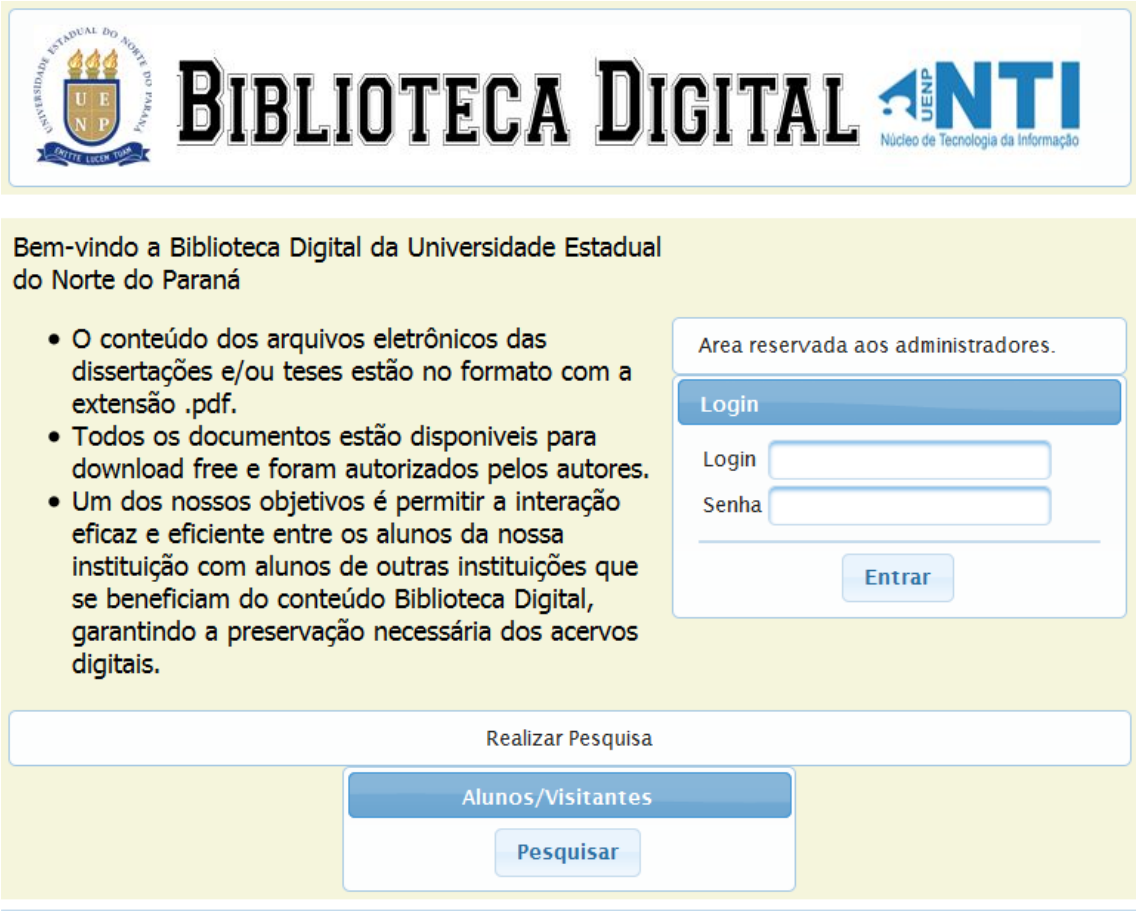

Figura 4.4 - Página Inicial da Biblioteca Digital "Pegasus".

O *login* é verificado com usuários cadastrados no Banco de Dados do sistema, caso a senha informada não conferira com o *login* será dado uma mensagem de erro para usuário, o mesmo ocorre se o usuário tentar acessar sem informar *login* e senha (Figura 4.5).

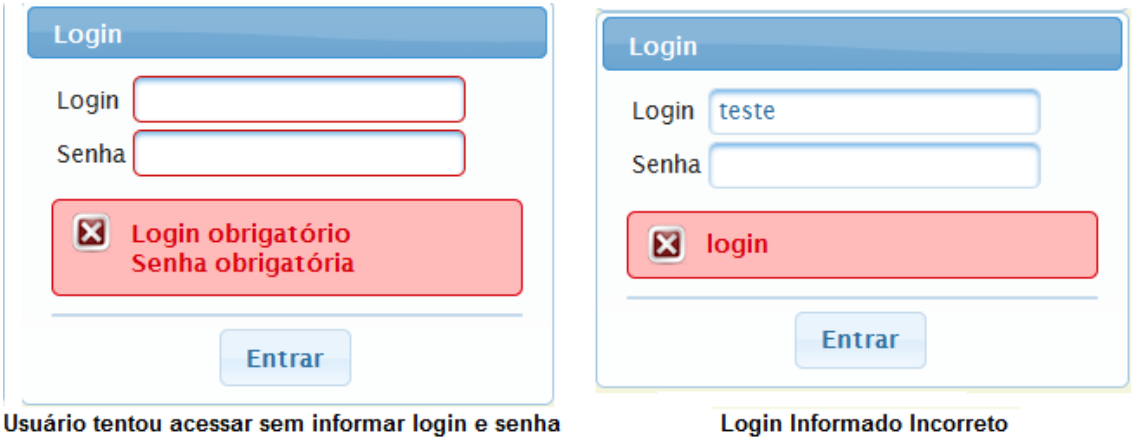

Figura 4.5 – Erros de Login

4.3.2 Funcionalidades do Aluno/Visitante

A aplicação quando acessada por alunos ou visitantes fornecerá acesso apenas algumas funcionalidades do sistema, porém essas serão as que eles realmente precisam. A tela irá exibir um campo de busca no qual deverá ser informado o assunto pesquisado, a interface ao receber o parâmetro da pesquisa irá se comunicar com o pacote *Bean* e chamar o método pesquisa do Lucene que por sua vez retornará aos documentos relacionados com o parâmetro pesquisado.

O aluno ainda contará com alguns menus na barra superior, os quais contém as normas para o uso das teses, a apresentação da biblioteca, qual o objetivo da biblioteca e um menu com os arquivos mais acessados no sistema. O menu é mostrado na Figura 4.6.

Figura 4.6 - Tela Principal do Sistema

Funcionalidades dos Alunos/Visitantes

O botão de busca avançada irá redirecionar o usuário para outra página, na qual se pode fazer a pesquisa por metadados, por exemplo, o usuário do sistema seleciona a opção título e informa o título da obra, o sistema irá retornar todas as obras que contém o parâmetro digitado. Os metadados são: Título, Autor, Orientador, Ano Publicação, Nível (Mestrado ou Graduação), Palavra Chave, Resumo, *Abstract* e Sub-Área. Os metadados utilizados no sistema foram incluidos por meio de levantamentos de requisitos realizado com a bibliotecária da UENP-CLM.

4.3.3 Funcionalidades do Responsável

O usuário do tipo responsável terá as funcionalidades do aluno/visitante e mais duas funções, que será cadastrar documentos na biblioteca e se caso precisar consultar o documento de autorização de um determinado documento Na Figura 4.7 são exibidas essas funcionalidades.

Figura 4.7 – Funcionalidade do Responsável.

Ao selecionar o menu Enviar Arquivo, o sistema é redirecionado uma nova página na qual o responsável poderá cadastrar as informações do documento. Esse processo é feito em 3 partes:

1. O responsável iniciará cadastrando os metadados do sistema, não podendo deixar nenhuma informação em branco. Na Figura 4.8 **é** demonstrada uma tentativa de cadastrar um documento sem fornecer o autor da tese.
2. Ao cadastrar todos os metadados o usuário irá carregar os arquivos (TCCs ou dissertações de Mestrado e a autorização do documento), para que se possa carregar os arquivos devem estar com a extensão .pdf, essa informação pode ser acessada no menu inicial em Instruções. A tela de cadastro de metadados é mostrada na Figura 4.9**.**

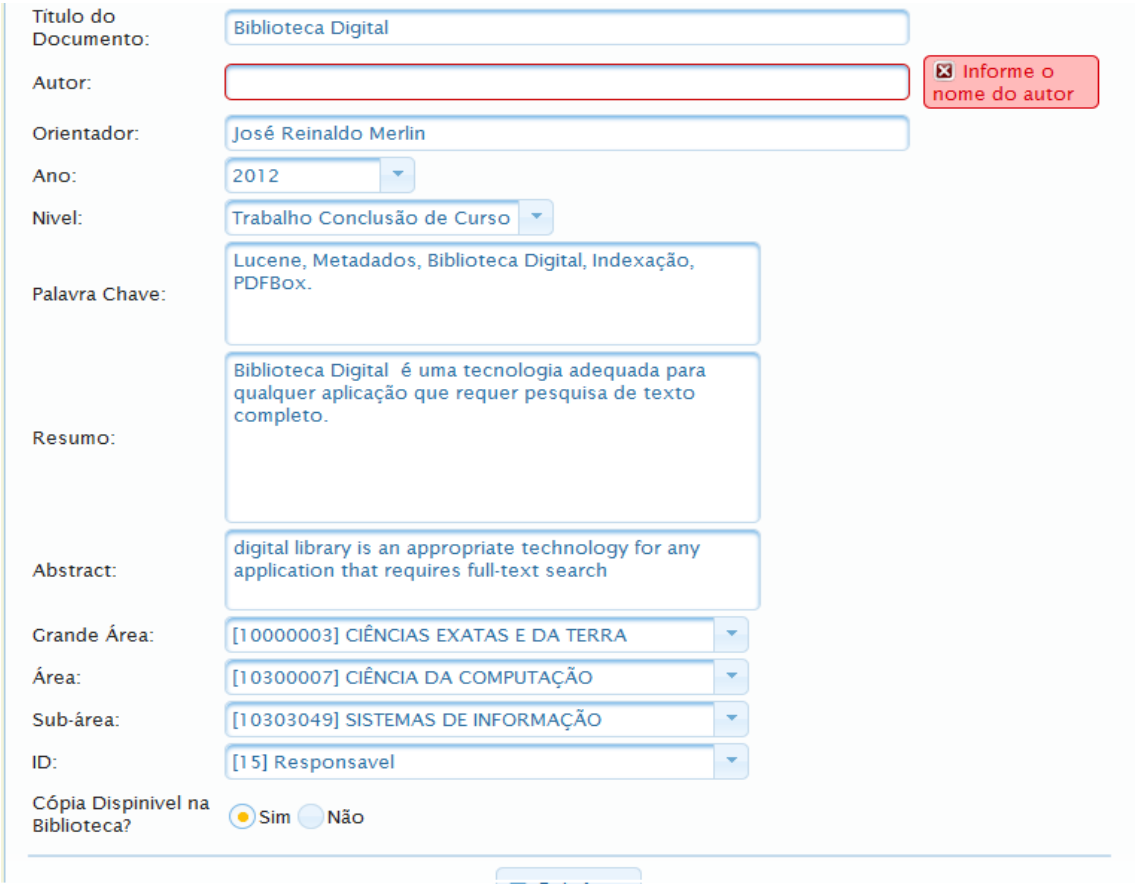

Figura 4.8 – Cadastro de metadados com erro de campo nulo.

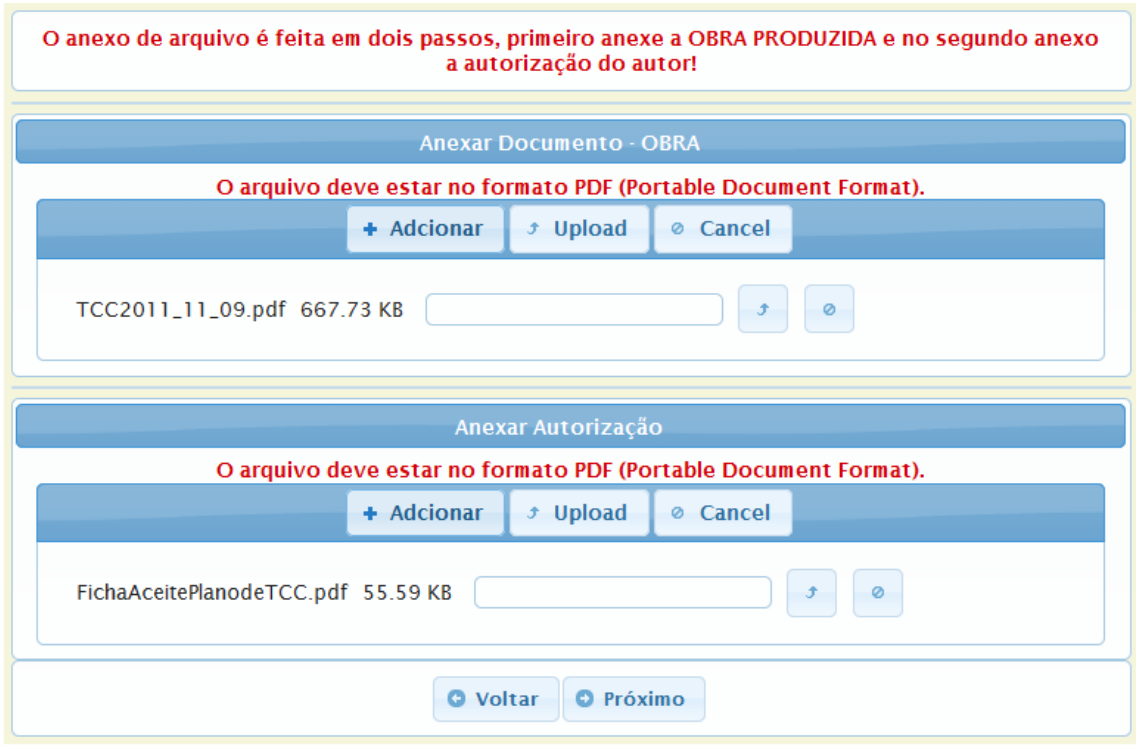

Figura 4.9 – Upload de Arquivos.

3. Na terceira e última etapa é confirmado os dados que foram cadastrados por meio de uma página *xhtml* e se os dados estiverem todos corretos basta gravar. Para evitar problemas com os arquivos que são anexados, os nomes dos arquivos são alterados, pois pode ocorrer do arquivo estar com um nome que não tem sentido algum com a aplicação, como foi o caso demonstrado na Figura 4.10**.** Isso é feito por meio de uma verificação no código de *upload*, alterando o nome do documento como os valores dos metadados: nome do autor e título, e a autorização somente o título da obra.

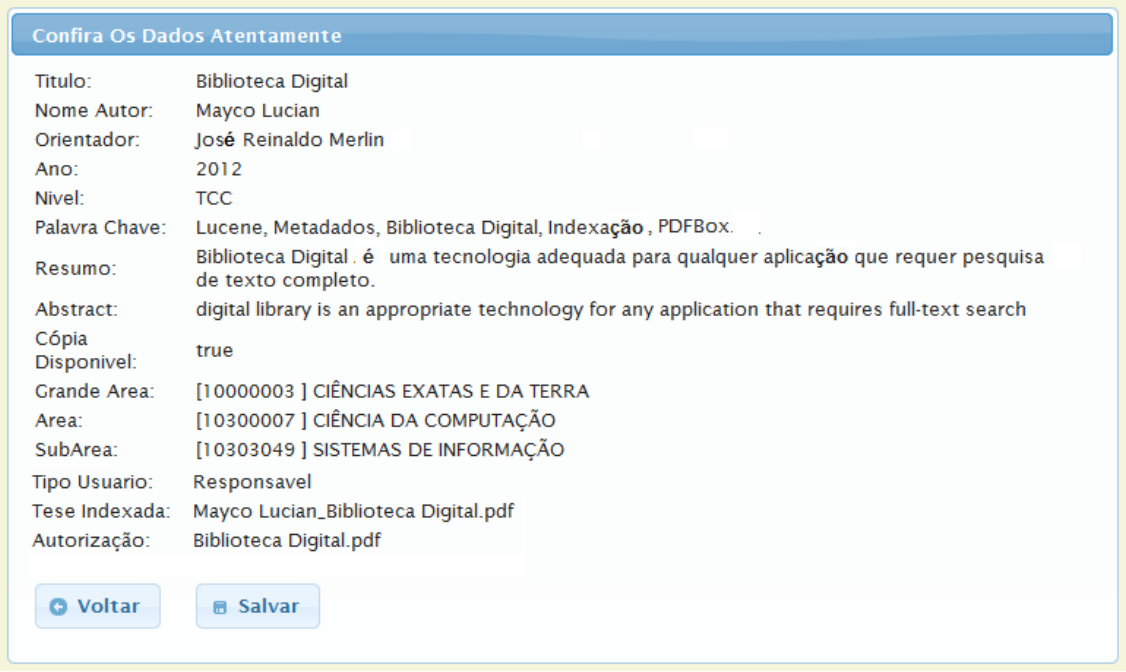

Figura 4.10 – Confirmação

O menu Consultar Autorizações exibe uma tela que tem por função retornar a autorização assinada pelo aluno autorizando a disponibilização da tese do mesmo na biblioteca digital.

Será solicitado que o usuário informe o título da obra defendida, no exemplo acima foi indexado a obra “Biblioteca Digital” então ao pesquisar sua autorização o usuário terá que informar este nome. O sistema usa o Lucene para realizar a busca do arquivo, para ser uma busca mais rápida e eficiente o diretório de autorizações será

indexado somente pelo metadado nome da obra, ao encontrar o documento o sistema retorna um link para *download* do mesmo. O processo é ilustrado na Figura 4.11**.**

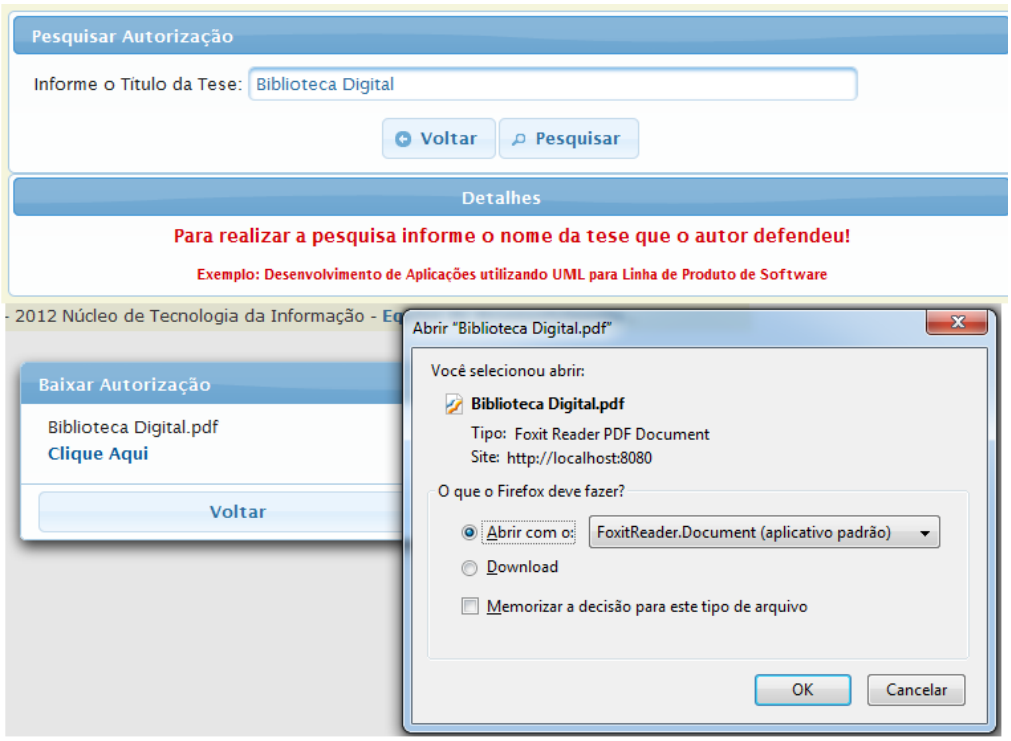

Figura 4.11 – Pesquisar Autorização.

4.3.4 Funcionalidades do Administrador

Ao logar-se como administrador o usuário pode fazer tudo que os demais usuários fazem e criar *logins* e senhas para os responsáveis do sistema. Para criar um novo usuário será necessário preencher os seguintes dados da tela de cadastro: nome, sobrenome, email, campus, cpf, *login*, senha e tipo do usuário (Figura 4.12), após preencher os dados o sistema exibe todos os usuários cadastrados no sistema (Figura 4.13).

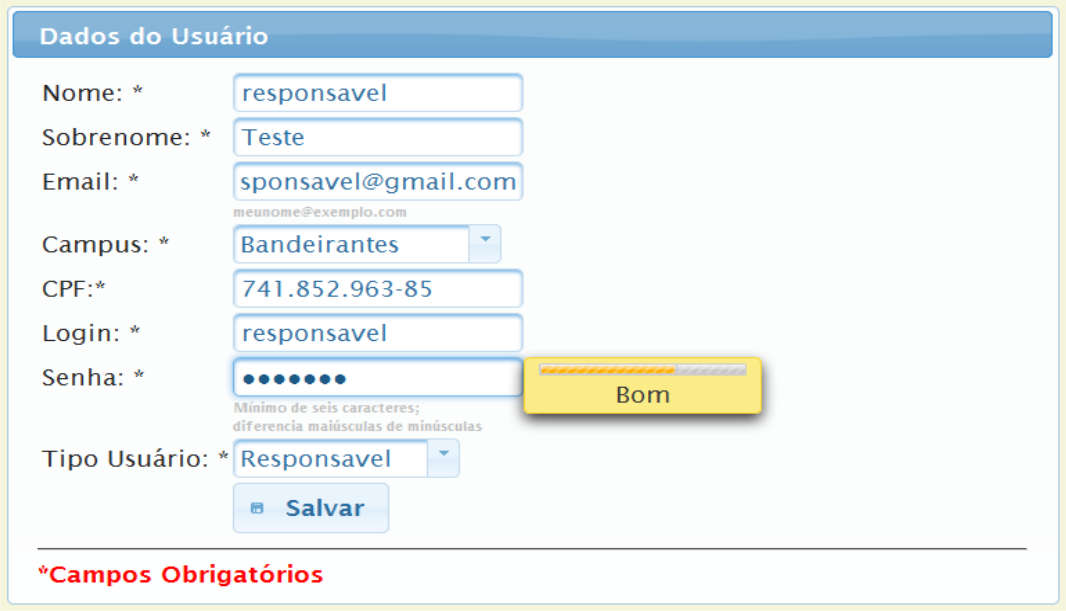

Figura 4.12 – Tela de Cadastro de Usuário.

Figura 4.13 – Exibição dos Usuário Cadastrados.

4.3.5 A Indexação do Sistema

Como já citado no desenvolvimento do projeto, para o processo de armazenar e buscar a informação será utilizando Lucene. Os dados que serão anexados estão em formato pdf, porém os dados serão extraídos para informações textuais para que as buscas se tornem mais rápidas.

O primeiro passo para a indexação é converter os dados em texto puro, os dados serão armazenados em uma estrutura de dados chamada de índice, para ser salvo no índice os dados são analisados. Durante o processo de análise, o texto passa por diversas operações: extração das palavras, remoção das *stopswords*, entre outras.

Por meio dessas operações os dados são convertidos para *tokens*, é com os *tokens* que se consegue obter os filtros de pesquisas.

Existem alguns analisadores integrados ao Lucene, *StopAnalyzer, SimpleAnalizer, WhitespaceAnalyzer* e o *StandardAnalyzer*. O analisador usado no sistema Pegasus é o *StandardAnalyzer*, a escolha por este analisador deve-se pelo fato dele ser mais completo que os demais, o *StandardAnalyzer* além de remover as *stopswords* e dividir o texto em *tokens,* possui uma gramática sofisticada, como endereços de e-mail e caracteres alfanuméricos.

Assim que o documento é analisado e filtrado o arquivo é indexado no índice do Lucene. Na Figura 4.14 é ilustrando o processo descrito.

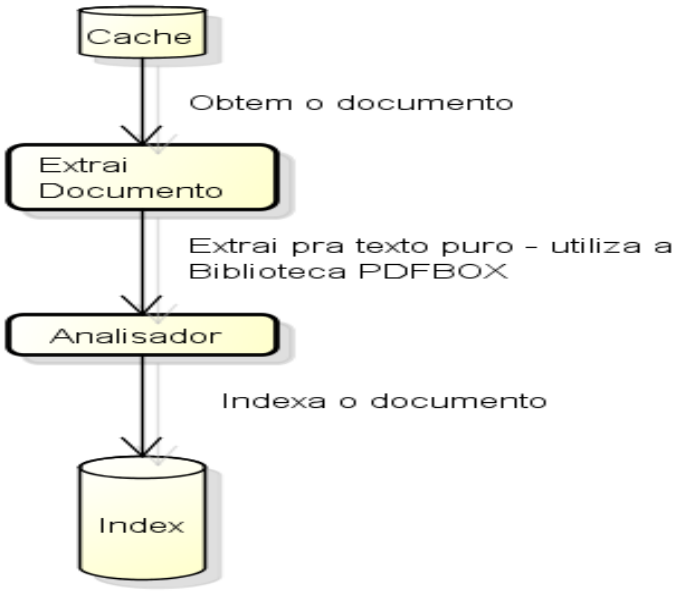

Figura 4.14 – Processo Indexação

O processo de busca no índice utiliza a classe *Search* (Procurador), é uma classe que possui métodos de procura sobrecarregados. Para recuperar a informação encontrada pela classe *Search* as classes *ScoreDoc* e *TopDocs* são utilizadas para realizar este serviço.

O *ScoreDoc* é responsável por obter a posição do documento no índice, retorna a posição exata do documento. O *TopDocs* engloba o total de documentos encontrados na pesquisa e uma matriz de *ScoreDoc*, se a palavra pesquisada estiver contida em mais de um documento o *ScoreDoc* retornará uma matriz com todas as posições em que se encontra as palavras pesquisadas.

O processo de busca pode ser acompanhado na Figura 4.15**.**

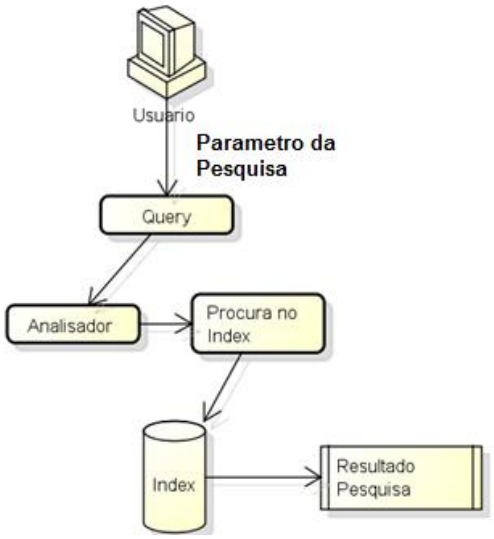

Figura 4.15 – Busca

O sistema Pegasus esta utilizando o termo *QueryParser* (analisar cadeias de consultas inseridas pelo usuário) para fazer a procura no *IndexSearch*. O *IndexSearch* é uma classe que realiza a procura direto no índice do Lucene.

Foi implementado com o *QueryParser* pois o mesmo não é *case sensitive* por tanto não tem importância se o usuário digitar tudo em minúsculo e a informação do documento estiver em maiúsculo. Como mostrado na Figura 4.12 o usuário informa a pesquisa que irá ser tratada pela *Query*. A *Query* por sua vez é analisada (*Analyser)* pois o Lucene considera somente as palavras de importância (desconsiderando as *stopwords*) e será feito busca diretamente no índice do Lucene pelo *IndexSearch,* retornando os resultados na tela para o usuário.

4.3.6 Testes para verificar a Indexação

Para confirmar se os documentos estão sendo indexados corretamente foi usada uma ferramenta do próprio Lucene conhecida como Luke. Luke é uma ferramenta de diagnóstico, já que acessa índices Lucene existentes e permite exibir e modificar seu conteúdo. Neste tópico será demonstrado o processo de indexação utilizando esta ferramenta.

No exemplo demonstrado no tópico 1.3.3 foi indexado o documento Mayco Lucian_Biblioteca Digital. Será usada esta mesma obra para demonstrar o funcionamento do Luke, mostrado na Figura 4.16**.**

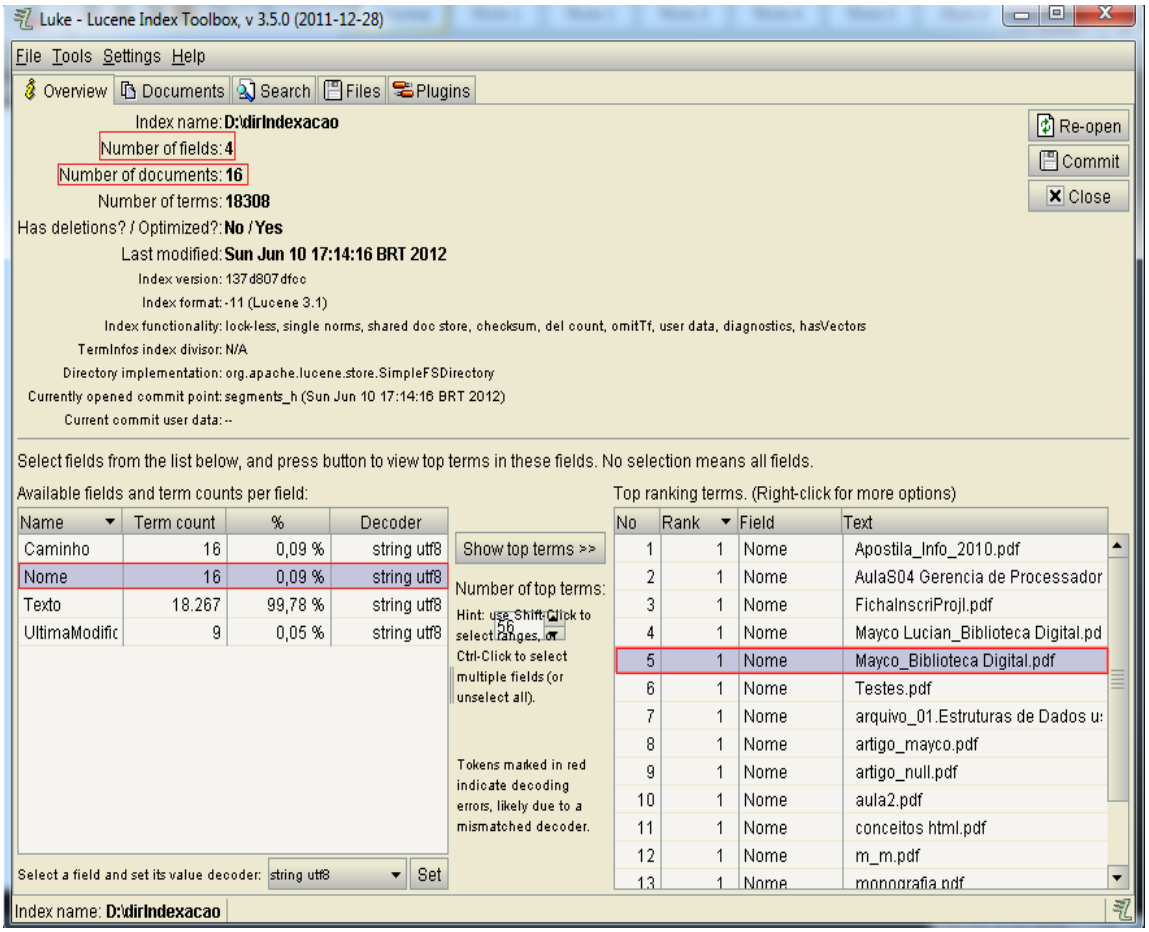

Figura 4.16 – Luke

Assim que se escolhe o índex do Lucene, o Luke exibe todos os arquivos e algumas outras informações. Pode-se observar que o diretório é D:\dirIndexacao e que foram indexados 4 campos sendo eles: Caminho; Nome; Texto e Ultima Modificação.

O número total de arquivos no índex também é exibido, no caso da Figura 4.16 16 arquivos indexados. Agora que já selecionado o arquivo desejado pode-se verificar o conteúdo do texto indexado podendo também ser feitas alterações no conteúdo, como mostrado na Figura 4.17**.**

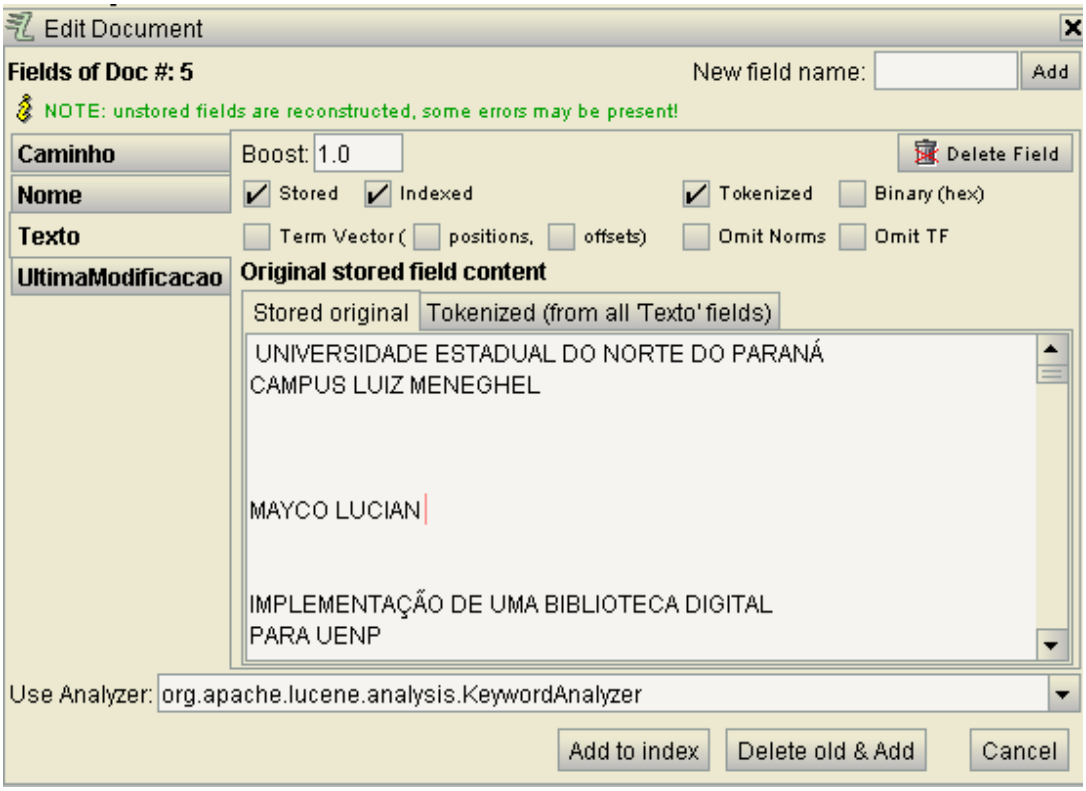

Figura 4.17 – Documento

Como se pode observar na Figura 4.17 é possível fazer alterações no documento e adicioná-lo novamente ao índex e até excluir o documento anterior salvando apenas o que foi alterado.

Na aba *Tokenized* (Figura 4.19) são listadas as palavras que mais se repetem no texto e por meio delas são retiradas as *stopwords*. A implementação do código Lucene (Figura 4.18) irá ter acesso a estas palavras e conferir com as palavras existentes da classe *StopWords,* se a palavra que mais se repete for igual a da classe ela será descartada nas pesquisas**.**

```
public final class BrazilianAnalyzer extends Analyzer {

  // TODO make this private in 3.1
        public final static String[] BRAZILIAN_STOP_WORDS = {
      "a","ainda","alem","ambas","ambos","antes",
      "ao","aonde","aos","apos","aquele","aqueles",
      "as","assim","com","como","contra","contudo",
      "cuja","cujas","cujo","cujos","da","das","de",
      "dela","dele","deles","demais","depois","desde",
      "desta","deste","dispoe","dispoem","diversa",
      "diversas","diversos","do","dos","durante","e",
      "ela","elas","ele","eles","em","entao","entre",
      "essa","essas","esse","esses","esta","estas",
      "este","estes","ha","isso","isto","logo","mais",
      "mas","mediante","menos","mesma","mesmas","mesmo",
      "mesmos","na","nas","nao","nas","nem","nesse","neste",
      "nos","o","os","ou","outra","outras","outro","outros",
      "pelas","pelas","pelo","pelos","perante","pois","por",
      "porque","portanto","proprio","propios","quais","qual",
      "qualquer","quando","quanto","que","quem","quer","se",
      "seja","sem","sendo","seu","seus","sob","sobre","sua",
      "suas","tal","tambem","teu","teus","toda","todas","todo",
      "todos","tua","tuas","tudo","um","uma","umas","uns"};
```

Figura 4.18 – Classe que elimina as StopWords

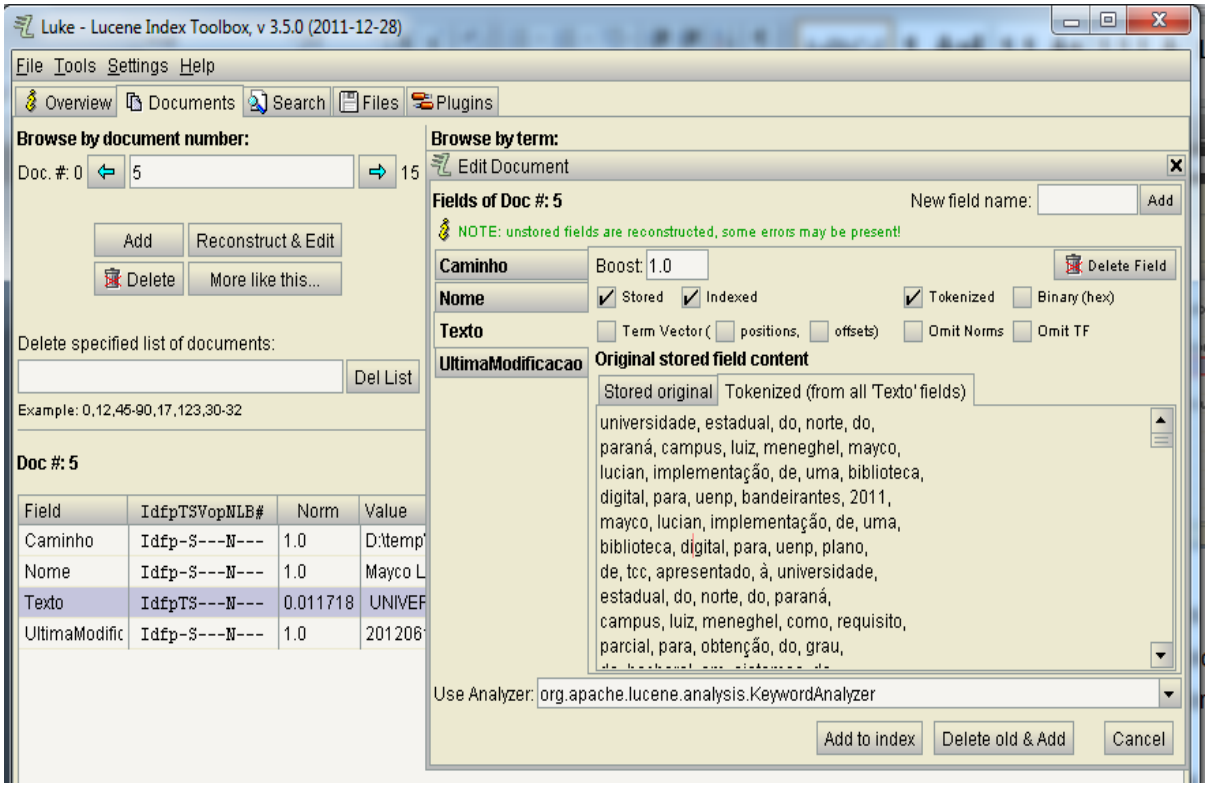

Figura 4.19 – Aba Tokenized.

5 CONSIDERAÇÕES FINAIS

Este trabalho apresentou as principais tecnologias que foram utilizadas na implementação do sistema Pegasus, que é uma Biblioteca Digital desenvolvida para UENP. O foco inicial da Biblioteca era pra acadêmicos da UENP, entretanto, a partir dos estudos realizados o foco foi ampliado, atingindo também o público em geral.

A biblioteca digital irá possibilitar inúmeros benefícios para a instituição, dentre eles a organização da informação, por ser sistema web os alunos poderão acessar a qualquer hora e de qualquer lugar desde que tenha acesso à internet. Não precisa ser exigida nenhuma configuração de hardware por parte do usuário. Com o tempo estima-se que a biblioteca ajudará na divulgação da universidade, considerando que alunos de outras instituições usaram o sistema de busca, lembrando que em nosso índice só existiram publicação de acadêmicos e professores da UENP.

Existem muitos sistemas de biblioteca digital no mercado, porém usam tecnologias antigas e código fechado. Dentre as vantagens de a instituição ter seu próprio sistema de biblioteca digital é a oportunidade de poder realizar a integração com outros sistemas da universidade, como por exemplo, interligar o Pegasus com o sistema da biblioteca (física) da UENP, além do mais, o sistema implementado para a UENP conta com tecnologias novas que facilitam o uso e otimizam as buscas de informação.

Como continuação deste trabalho, sugere-se a implementação da funcionalidade *SpellChecker* ao sistema Pegasus. *SpellChecker* é uma classe do Lucene que funciona como um corretor ortográfico, que sugere uma lista de palavras semelhantes a uma palavra digitada incorretamente, por exemplo: O usuário acidentalmente digita “bibioteca”, o sistema irá dar a sugestão de “você quis dizer: biblioteca”. Outra possível continuação é introduzir os conceitos de preservação digital junto ao Pegasus.

Ainda como projeto futuro, fazer uma ligação com o sistema da biblioteca física com a Biblioteca Digital dessa forma o aluno poderá consultar quais livros a instituição possui e quais estão disponíveis.

REFERÊNCIAS

ALVES, Maria das Dores Rosa; SOUZA, Marcia Izabel Fugisawa. **Estudo De Correspondência De Elementos Metadados: Dublin Core e Marc 21.** Revista Digital de Biblioteconomia e Ciência da Informação, Campinas, v. 4, n. 2, p. 20-38, jan./jun. 2007 – ISSN: 1678-765X.

ASSOREIRA, Paulo; MOURÃO ,Cecília. **Bibliotecas Digitais**. Fevereiro de 2001.

BRAND AMY, Daly Frank, MEYERS Barbara. **Understanding Metadada.** NISO Press - National Information Standards Organization, julho 2003.

FERREIRA, Luciana Santana. **Indexação e Pesquisa Utilizando o Apache Lucene e o Hibernate Search: Uma Abordagem Comparativa com Ferramentas Existentes.** Universidade Católica do Salvador Curso de Bacharelado em Informática, Salvador 2010.

HATCHER, Erik; GOSPODNETIC, Otis; MCCANDLESS, Michael. **Lucene In Action**. Copyright 2009 Manning Publications.

Lucene. Acesso: <http://lucene.apache.org/> , em 10/04/2012.

MARANHÃO, Ana Maria Neves; MENDONÇA, Maria de Lourdes dos Santos. **MARC 21 - Formato Bibliográfico**. Divisão de Bibliotecas e Documentação - PUC-Rio Fevereiro 2010.

MONTEZ , Carlos; PISTORI, Jeferson; WILLRICH, Roberto.**Experiências na Implementação da Biblioteca Digital Multimídia RMAV/Florianópolis.** Universidade Fernando Pessoa, 2000.

NUNES, Rodrigo Irineu. **BIBLIOTECA DIGITAL APOIANDO O ENSINO.** Universidade do Vale do Itajaí – Centro de Ciências Tecnológicas da Terra do Mar – Itajaí Junho 2005.

BIBLIOGRAFIA CONSULTADA

AMOL, Sonawane. **Usando o Apache Lucene para Procura de Texto.** Acesso: < http://www.ibm.com/developerworks/br/java/library/os-apache-lucenesearch/#author1> em 15/04/2012.

DIAS, Uirá. **Indexando Documentos com Apache Lucene.** Acesso: <http://uiradias.wordpress.com/2011/03/25/indexando-documentos-com-apache-lucene/> em 15/04/2012.

Editco Comercial Ltda. **Construindo uma Biblioteca Digital.** Edições Inteligentes, São Paulo 2005.

ODEON, José Moraes Rosa. **Desenvolvimento de Aplicação WEB 2.0 Usando o Framework GWT Para Controle de Atividades Complementares.** Universidade Estadual do Norte do Paraná. Bandeirantes 2008.

PEREIRA, Maria de Lurdes Carvalhais. **Tecnologias de Informação Documental - Como Organizar uma Pequena Biblioteca Digital**. Porto, Junho de 2003.

RAABEL, André Luís Alice; PREBIANCAL, Giovana; da Silva, MARQUES Júlia; OLIVEIRA, Renate. **Avaliação de Usabilidade de uma Biblioteca Digital**. Ciência da Computação - CTTMar – UNIVALI, Pós-Graduação em Engenharia de Produção, 2003.

ROSETTO, Marcia; NOGUEIRA, Adriana Hypólito. **Aplicação de Elementos Metadados Dublin Core Para Descrição de Dados Bibliográficos On-Line da Biblioteca Digital de Teses da Usp.** Universidade de São Paulo - Departamento Técnico – São Paulo 2002.