

LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RN
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

UMA APLICAÇÃO WEB PARA UM SISTEMA DE HELPDESK

Wagner Nicacio de Castro Barbosa

NATAL/RN

2005

RB- 26.563

FARN
BIBLIOTECA
Nº Reg. <u>M000443</u>
<u>18</u> / <u>05</u> / <u>06</u>

M

004.4

B563a

2005

WAGNER NICACIO DE CASTRO BARBOSA

UMA APLICAÇÃO WEB PARA UM SISTEMA DE HELPDESK

Monografia apresentada ao Curso de Graduação em Bacharelado em Sistemas de Informação da Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador:

Aluizio Ferreira da Rocha Neto

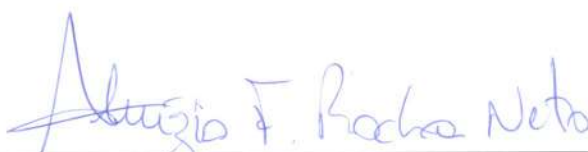
**NATAL/RN
2005**

**LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RN
BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

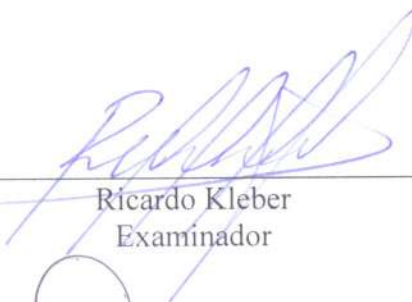
UMA APLICAÇÃO WEB PARA UM SISTEMA DE HELPDESK

Wagner Nicacio de Castro Barbosa

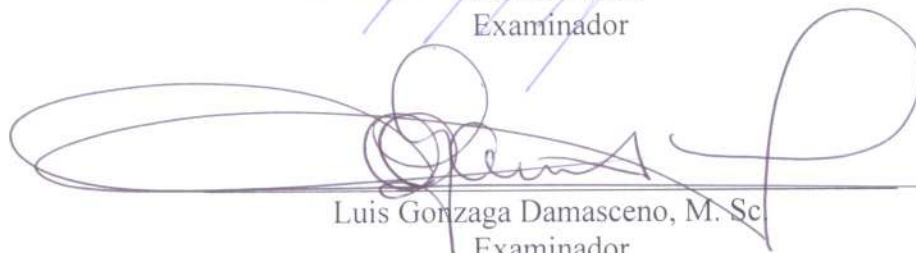
Monografia apresentada e aprovada em 17 de 12 de 2005 pela banca examinadora composta pelos seguintes membros:



Aluizio Ferreira da Rocha Neto, M. Sc.
Orientador



Ricardo Kleber
Examinador



Luis Gonzaga Damasceno, M. Sc.
Examinador

Natal 17 de DEZEMBRO de 2005.

DEDICATÓRIA

Dedico esta obra em memória da minha avó, Maria Martins de Castro, por sua grandeza, por suas palavras, por sua fé inabalável nas coisas inexplicáveis da vida.

AGRADECIMENTOS

Os meus sinceros agradecimentos,

Aos meus pais que acreditaram em mim e me ajudaram a tornar esse momento possível.

Ao professor e orientador Aluizio Rocha, por sua ajuda e por sua dedicação inestimável para a realização deste projeto.

A todos os professores que fazem parte desta instituição e que de alguma maneira me incentivaram a seguir em frente, mesmo diante dos muitos obstáculos que encontrei nessa jornada, e que por muitas vezes fizeram com que essa caminhada parecesse menos árdua.

A Gabrielle minha adorável sobrinha, que encheu a casa e meu coração de felicidade com sua chegada e sua alegria.

RESUMO

O desenvolvimento de aplicativos para web vem crescendo cada vez mais, e já se tornou comum. Há uma necessidade real de acesso a informações, e a possibilidade de acessar informações dos mais variados tipos em qualquer lugar já faz parte do dia a dia. Softwares são desenvolvidos para os mais variados tipos de serviços, e a web é um campo vasto para propagação destes. Esse projeto é um estudo sobre o desenvolvimento de um aplicativo para web, utilizando JSP, uma tecnologia para criação páginas dinâmicas, que utiliza a plataforma Java 2 Standard Edition, além de implementar um estudo de caso prático e funcional usando essa tecnologia.

Palavras-chave: Java. Java 2 Standard Edition. Aplicações web. JSP.

ABSTRACT

The development of applicatory for web, is growing more and more, became already common. There is a real need of access to information, and the possibility to access information of the most varied types anywhere is already day by day part of the. Softwares is developed for the most varied types services, and the web is a vast field for propagation of these. This project is a study on the development of web applications, using JSP, a technology for creation dynamic pages, that it uses the platform Java 2 Standard Edition, besides implementing a study of practical and functional case using this technology.

Word Key: Java. Java 2 Standard Edition. Applications WEB. JSP.

SUMÁRIO

1	INTRODUÇÃO.....	13
1.1	PROBLEMA DE PESQUISA.....	14
1.2	OBJETIVOS GERAIS	14
1.3	OBJETIVOS ESPECÍFICOS	15
1.4	JUSTIFICATIVA.....	15
1.5	METODOLOGIA	16
2	REFERENCIAL TEÓRICO.....	17
2.1	PLATAFORMA JAVA.....	17
2.1.1	Java independente de plataforma	19
2.1.2	Java e o paradigma de Programação Orientada a Objetos	19
2.1.3	Herança e Polimorfismo	20
2.1.4	Java Server Pages - JSP	21
2.1.5	Java Database Connectivity - JDBC.....	23
2.2	SISTEMA GERENCIADOR DE BANCO DE DADOS - SGBD.....	23
2.2.1	Modelo Entidade-Relacionamento – (E-R).....	24
2.2.2	A linguagem SQL – <i>Structured Query Language</i>	25
2.3	UTILIZANDO UML, PADRÕES E ANÁLISE ORIENTADA A OBJETOS	26
2.4	ARQUITETURA DE SISTEMAS.....	27
2.5	APLICATIVO WEB	29
2.6	PADRÕES DE PROJETO	31
2.7	FERRAMENTAS UTILIZADAS.....	31
3	ANÁLISE E PROJETO DO SISTEMA	33
3.1	VISÃO GERAL DO SISTEMA.....	33
3.2	DIAGRAMA DE CASO DE USO.....	33
3.2.1	Formatos de casos de usos textuais	34
3.2.2	Diagramas de Casos de Uso de alto nível	35
3.3	DIAGRAMAS DE CLASSES	42
3.4	MODELAGEM DO BANCO DE DADOS	44
3.5	TELAS DO SISTEMA.....	45

3.6	MAPA DO SITE	53
4	ANÁLISE DE RESULTADOS.....	54
4.1	ANÁLISE SOBRE O DESENVOLVIMENTO	55
5	CONCLUSÃO.....	56
5.1	CONSIDERAÇÕES FINAIS	56
	REFERÊNCIAS BIBLIOGRÁFICAS	58

Figura 3.10

Figura 3.11

Figura 3.12

Figura 3.13

Figura 3.14

Figura 3.15

Figura 3.16

Figura 3.17

Figura 3.18

Figura 3.19

Figura 3.20

LISTA DE FIGURAS

Figura 1: A divisão da plataforma Java 2	18
Figura 2: Modelo de camadas	27
Figura 3: Padrão MVC	30
Figura 4: Casos de Uso do Sistema	34
Figura 5: Diagrama de Classe da Aplicação	44
Figura 6: Modelo Operacional do Banco de Dados	44
Figura 7: Tela principal do Sistema	45
Figura 8: Tela de login do sistema	46
Figura 9: Tela de cadastro do usuário	47
Figura 10: Tela para registrar chamado.....	48
Figura 11: Tela de visualização de chamados já realizados pelo usuário	49
Figura 12: Tela de Artigos.....	50
Figura 13: Tela para mostrar artigo selecionado	51
Figura 14: Tela de busca de artigo técnico	52
Figura 15: Mapa do Site	53

LISTA DE TABELAS

Tabela 1: Cenário Caso de uso Cadastrar Usuário	35
Tabela 2: Cenário Caso de uso Efetuar Login.....	36
Tabela 3: Cenário Caso de Uso Lembrar Senha.....	37
Tabela 4: Cenário Caso de Uso Abrir Chamado	37
Tabela 5: Cenário Caso de Uso Listar Chamados (OS).....	38
Tabela 6: Cenário Caso de Uso Buscar Artigo.....	39
Tabela 7: Cenário Caso de Uso Ver Links	39
Tabela 8: Cenário Caso de Uso Efetuar logout	40
Tabela 9: Cenário Caso de Uso Cadastrar Artigo	41
Tabela 10: Cenário Caso de Uso Mudar status da OS	41
Tabela 11: Cenário Caso de Uso Inserir links	42

LISTA DE SIGLAS

TI	Tecnologia de Informação
JSP	Java Server Pages
UML	Unified Modeling Language
J2SE	Java Standard Edition
J2EE	Java Enterprise Edition
J2ME	Java Micro Edition
EJB	Enterprise Java Bean
PDAs	Personal Digital Assistants
POO	Programação Orientada a Objetos
HTML	Hyper Text Markup Language
ASP	Active Server Pages
JDBC	Java Database Connectivity
SGBD	Sistema Gerenciado de Banco de Dados
SQL	Structured Query Language
MVC	Model View Controller
DAO	Data Access Object
OS	Ordem de Serviço

1 INTRODUÇÃO

O uso de computadores atualmente vem crescendo e se popularizando cada vez mais em todo o mundo, hoje em dia é muito comum ouvirmos falar em comunicação entre computadores através de redes, comunicação esta feita principalmente através da Internet (a rede mundial de computadores). O uso de tecnologias corporativas baseadas na Internet, como a Intranet, melhora significativamente a eficiência e produtividade dos funcionários na empresa, possibilitando que esses possam executar serviços de forma mais otimizada, eficaz e transparente.

Com o surgimento da *World Wide Web* os usuários de computador podem localizar e visualizar documentos baseados em multimídia sobre quase qualquer assunto pela Internet, a Internet explodiu tornado-se, um dos principais meios de comunicação do mundo.

Segundo Deitel a Internet e a *World Wide Web* são as criações mais importantes e profundas que já ocorreram na área tecnológica. Antigamente, a maioria dos aplicativos de computador era executado em computadores que não se comunicavam entre si. Os aplicativos de hoje podem ser escritos para se comunicarem com centenas de milhões de computadores do mundo. A web une computação e tecnologias de comunicação. Ela torna o trabalho mais fácil e as informações acessíveis mundialmente de maneira instantânea e conveniente. Permite ainda que indivíduos e empresas de pequeno porte local obtenham exposição mundial.

Esse trabalho visa gerenciar chamadas ao setor de suporte em TI (Tecnologia da Informação) de uma empresa, através de um aplicativo web. Será apresentada nesta monografia a especificação e modelagem de um sistema *helpdesk* para web utilizando a tecnologia Java.

Esse sistema, denominado *Helpdesk*, tem como princípio coletar, manipular, disseminar dados e informações de chamados abertos pelos usuários da empresa RMNor do Brasil, gerenciando e controlando componentes inter-relacionados.

1.1 PROBLEMA DE PESQUISA

Diante do atual cenário de tecnologias de informação e da importância que ele representa para um bom desempenho de qualquer organização ou empresa, é necessário ter cautela na confecção de um software, desde sua modelagem até a sua implementação, para desenvolver um aplicativo para web é necessário levar em conta alguns fatores importantes, fatores esses como: qual público alvo se pretende atingir, se o software atinge os objetivos que se propôs a alcançar, se as soluções são viáveis, entre outros, a internet é um meio de comunicação de possibilidades ilimitadas.

Segundo Kurniawan (2002), os termos *Internet Application* (aplicativo de Internet) ou *web application* (aplicativo da web) foram cunhados quando foi introduzido o conteúdo dinâmico na Internet. Vagamente interpretado, um aplicativo web é um web site cujos conteúdos são gerados dinamicamente antes de serem enviados ao *browser*.

A sociedade contemporânea é muito dinâmica, e a cada dia novas tecnologias surgem com uma rapidez espantosa, e quem estiver na vanguarda, estará sempre um passo à frente no futuro.

Para Kurniawan (2002), o que isso significa, como alguns acreditam, é que quem controlar a Internet controlará o futuro da computação – ou mesmo o próprio futuro.

Foi desenvolvida com essa pesquisa uma solução viável para o problema de gerência e controle do setor de suporte em informática da empresa RMNor do Brasil Ltda, que atua na indústria têxtil, fazendo com que haja maior interação do usuário com a base de suporte através de uma aplicação em três camadas.

1.2 OBJETIVOS GERAIS

Este trabalho tem como objetivo principal desenvolver um “*web application*” (aplicativo web) em um paradigma de programação distribuída através da separação das aplicações entre servidores (aplicações que disponibilizam algum serviço) e clientes (aplicações que usam esses serviços).

1.3 OBJETIVOS ESPECÍFICOS

Desenvolvemos esse aplicativo utilizando a linguagem de programação Java, JSP (*Java Server Pages*) que é uma tecnologia baseada em Java que simplifica o processo de desenvolvimento de páginas dinâmicas na web e utilizamos também um servidor de banco de dados para guardar dados relacionados a informações dos chamados realizados pelos usuários e ainda UML (*Unified Modeling Language*) Linguagem de Modelagem Unificada para fazer a análise e modelagem do sistema.

A pesquisa será realizada nas seguintes etapas:

1. Levantar as informações que precisam estar disponibilizadas no website da empresa, com o intuito de atender as diversas necessidades dos usuários.
2. Modelar o sistema, ilustrar os conceitos e etapas deste, utilizando diagramas em UML, uma notação diagramática para a modelagem e o desenvolvimento de sistemas de software usando conceitos de orientação a objetos, que dá suporte e visualização a todas as etapas do desenvolvimento do sistema.
3. Demonstrar as vantagens de desenvolver o sistema na Plataforma Java, implementando o sistema.

Esta pesquisa irá difundir um maior conhecimento a sociedade do uso desta Tecnologia de Informação, o *helpdesk* quanto à forma em que foi desenvolvido.

1.4 JUSTIFICATIVA

O desenvolvimento de aplicações utilizando a Internet como infra-estrutura para seu funcionamento é hoje o foco de muitas organizações que prestam consultoria em desenvolvimento de sistemas. Isto faz com que, dependendo do cenário de desenvolvimento, as linguagens que possuem um suporte natural para a programação web estejam sendo as mais utilizadas. O que podem levar estas organizações ou pessoas a terem um diferencial em relação aos concorrentes, podendo oferecer serviços e acesso facilitando as informações referentes a tal negócio.

1.5 METODOLOGIA

Essa pesquisa é do tipo exploratória e funcional, e tem como foco principal o aperfeiçoamento de idéias com base em materiais já publicados e um projeto já elaborado.

O método utilizado nesta pesquisa foi hipotético-dedutivo, pois foi possível efetivar este projeto de forma funcional, elucidando todas as etapas do projeto, por meio de conhecimentos adquiridos com o decorrer da pesquisa.

A técnica de pesquisa empregada foi leitura documentada e bibliográfica.

O cenário para desenvolvimento do trabalho foi uma empresa da indústria têxtil, situada em Parnamirim, que confecciona camisas para algumas empresas privadas na cidade do Natal.

A arquitetura do sistema segue o paradigma de desenvolvimento em três camadas, onde a primeira camada define como serão tratadas as solicitações do usuário, a segunda camada é responsável pela lógica comercial e a terceira camada é responsável pela persistência de dados.

2 REFERENCIAL TEÓRICO

Para que se possa entender melhor este projeto, será feita uma contextualização dos principais conceitos e ferramentas utilizadas.

2.1 PLATAFORMA JAVA

Java é uma linguagem de programação orientada a objetos, muito conveniente para o desenvolvimento de software que funcione em conjunto com a Internet, desenvolvida pela Sun Microsystems.

“Quando a linguagem de programação Java foi lançada publicamente pela primeira vez, em novembro de 1995, ela era muito semelhante a um pistoleiro de Clint Eastwood perambulando em uma inóspita cidade do oeste norte-americano” (LEMAY; CADENHEAD, 1999, p. 10).

Java é uma linguagem de programação relativamente nova, que foi lançada pouco depois da explosão da Internet, a Sun viu na propagação da *World Wide Web* a oportunidade de criar programas que funcionem com a Internet.

“A World Wide Web explodiu em popularidade em 1993 e as pessoas da Sun viram o imediato potencial de utilizar Java para criar páginas na Web com o chamado conteúdo dinâmico”. (H. M. DEITEL; P. J. DEITEL, 2001, p. 64).

Podemos observar que Java apresenta um grande potencial, em especial para desenvolvimento na *World Wide Web*, funcionando independente de *browser* e plataforma.

Segundo os autores Lemay; Cadenhead (1999), a linguagem Java é mais conhecida por sua capacidade de executar em páginas da World Wide Web. Os navegadores podem carregar por download um programa Java a partir de uma página Web e executá-lo localmente no sistema de um usuário.

A plataforma Java avançou de uma maneira fenomenal desde o seu lançamento e atende os requisitos do desenvolvimento de software desde do mainframe do lado servidor até os menores dispositivos existentes, como cartões inteligentes. A plataforma J2SE

fornece um conjunto completo de classes e bibliotecas prontas para agilizar o desenvolvimento de aplicativos web, como será visto nas seções seguintes.

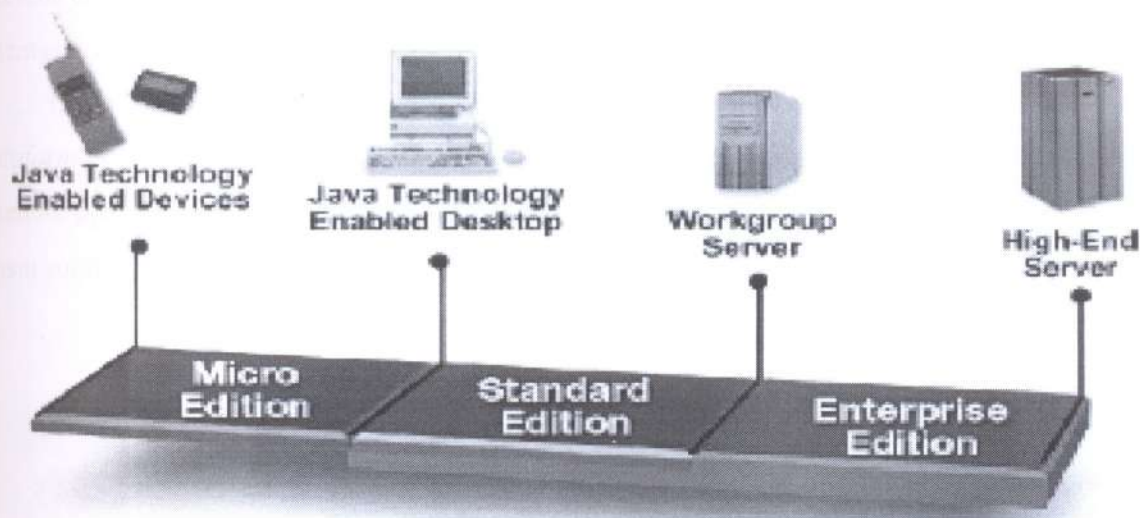


Figura 1: A divisão da plataforma Java 2

O Java atualmente encontra-se na sua segunda edição, sendo que a versão mais recente dentro dessa plataforma no momento da escrita desse trabalho é a 1.5, ou Java 5. Quando se diz "segunda edição" refere-se ao fato que quando a Sun lançou a versão 1.2 as modificações eram tão grandes que resolveu mudar o nome para Java 2, com isso houve também uma reestruturação dos pacotes (não confundir com *packages* da API), que agora contam com o prefixo J2, seguido do tipo do pacote (como em J2SE, por exemplo).

O Java conta com três diferentes tipos de pacotes, cada um com um foco específico:

- **J2SE (Java 2 Standard Edition):** É a versão básica (*core*), destinada ao desenvolvimento da maior parte das aplicações de *desktop* e estações de trabalho.
- **J2EE (Java 2 Enterprise Edition):** Versão destinada ao desenvolvimento de aplicações de grande porte, as quais fazem extenso uso de EJB (*Enterprise JavaBeans*) e Servidores de Aplicação, por exemplo. Outros grande personagem desse pacote são as *Servlets*.
- **J2ME (Java 2 Micro Edition):** Destinada ao desenvolvimento de programas para periféricos móveis ou de pequeno porte, como celulares e PDAs.

2.1.1 Java independente de plataforma

A linguagem de programação Java é independente de plataforma, ou seja, ela funciona em qualquer sistema operacional ou plataforma, desta forma ela foge destes outros sistemas prioritários enfadonhos, o que beneficia o usuário.

De acordo com os autores Lemay; Cadenhead (1999), a *independência de plataforma* – a capacidade de o mesmo programa ser executado em diferentes plataformas e sistemas operacionais – é uma das vantagens mais significativas que a linguagem Java possui sobre as outras linguagens de programação.

“Ela é linguagem multiplataforma, o que significa dizer que seus programas podem ser criados para executar do mesmo modo no Microsoft Windows, Apple Macintosh e na maioria das versões de UNIX, incluindo a Solaris”.

2.1.2 Java e o paradigma de Programação Orientada a Objetos

Uma das características mais interessantes de linguagens de programação orientadas a objeto é a capacidade de facilitar a reutilização de código – o aproveitamento de classes e seus métodos que já estejam escritos e que já tenham o seu funcionamento testado e comprovado. Reutilização de código diminui a necessidade de escrever novos métodos e classes, economizando o trabalho do programador e diminuindo a possibilidade de erros.

Vale lembrar que a forma de reusabilidade é bastante vasta, podendo se fazer a reutilização de código, reutilização de herança, reutilização de componentes e etc.

“A capacidade de reutilização de software economiza tempo no desenvolvimento de programas. Ela encoraja a reutilização de software de alta qualidade testado e depurado, reduzindo assim problemas depois que um sistema se torna operacional” (H. M. DEITEL; P. J. DEITEL, 2001, p. 385).

Para Santos (2003), *Programação Orientada a Objetos* ou, POO, é um paradigma de programação de computadores onde se usam classes e objetos, criados a partir dos modelos descritos anteriormente, para representar e processar dados usando programas de computadores.

De acordo com Lemay (1999), a programação orientada a objetos é, em sua raiz um modo de conceitualizar um programa de computador. Você pode imaginar um programa

como uma lista de instruções que dizem ao computador o que fazer, ou como um grupo de pequenos programas que respondem a eventos específicos, iniciados pela entrada do usuário.

2.1.3 Herança e Polimorfismo

As principais tecnologias de programação orientada a objetos são herança e polimorfismo. O que contribui para o desenvolvedor em uma enorme economia de tempo. Podendo reutilizar classes e métodos já projetados adicionando outros métodos para uma aplicação específica.

Existem dois mecanismos básicos de reutilização de classes em Java: delegação (ou composição) e herança. Com delegação, usamos uma instância da classe base como campo da nova classe, e com herança criamos a classe nova como uma extensão direta da classe base.

“Podemos criar novas classes que estendem uma outra classe base se incluirmos uma instância da classe base como um dos campos da nova classe, que serão então *composta* de campos específicos e de uma instância de uma classe base. Para que os métodos da classe base possam ser executados, escreveremos métodos correspondentes na classe nova que chamam os da classe base, dessa forma *delegando* a execução dos métodos”. (SANTOS, 2003, p.114).

Reutilização de classes visa o mecanismo de delegação, é útil quando consideramos que a classe que reutiliza instâncias de outras é *composta* das outras classes.

“Nem sempre o mecanismo de delegação é o mais natural para reutilização de classes já existentes, embora seja simples. Em especial, quando queremos usar uma classe para servir de base à criação de outra mais especializada, a relação de composição imposta pelo uso do mecanismo de delegação acaba por criar soluções pouco naturais”. (SANTOS, 2003, p.122).

Java oferece uma outra maneira de reutilizar classes, através do mecanismo de *herança*, que permite que criemos uma classe usando outra como base e descrevendo ou implementando as diferenças e adições da classe usada como base, reutilizando os campos e métodos não privados da classe base.

“O mecanismo de herança é o mais apropriado para criar relações *é-um-tipo-de* entre classes”. (SANTOS, 2003, p.123).

Segundo Lemay; Cadenhead (1999), herança é um mecanismo que permite a uma classe herdar de outra todo o comportamento e os atributos de outra classe.

Para M. Deitel; P. J. Deitel (2001), herança é uma forma de reutilização de software em que novas classes são criadas a partir de classes existentes, absorvendo seus atributos e comportamentos e sofisticando-os com capacidades que as novas classes exigem. A capacidade de reutilização de software economiza tempo no desenvolvimento de programas.

Podemos observar que o conceito de herança na visão dos autores citados é bastante semelhantes, que implicam nas mesmas ações.

A linguagem de programação orientada a objetos entre outros benefícios nos permite usar polimorfismo, entende-se como polimorfismo a capacidade de usar a mesma mensagem para invocar comportamento diferentes do objeto.

“A relação *é-um-tipo-de* entre classes permite a existência de outra característica fundamental de linguagens de programação orientadas a objetos: *polimorfismo*. Polimorfismo (“muitas formas”) permite a manipulação de instâncias de classes que herdam de uma mesma classe ancestral de forma unificada: podemos escrever métodos que recebam instâncias de uma classe C, e os mesmos métodos serão capazes de processar instâncias de qualquer classe que herde da classe C, já que qualquer classe herde de C *é-um-tipo-de* C. (SANTOS, 2003, p.140).

Como definido pelos autores H. M. Deitel; P. J. Deitel (2001), o polimorfismo permite-nos escrever programas de uma forma geral para tratar uma ampla variedade de classes relacionadas existentes e ainda a serem especificadas. O polimorfismo torna fácil adicionar novos recursos a um sistema. Herança e polimorfismo são técnicas eficazes para lidar com a complexidade de software”.

2.1.4 Java Server Pages - JSP

Java ServerPages é a utilização de código Java dentro da página HTML, para tornar suas páginas dinâmicas. JSP compartilha dados entre objetos Java, permite o uso de

ferramentas como *Homesite*, *FrontPage*, separa melhor o trabalho do Webdesigner do desenvolver Java.

É uma tecnologia para desenvolvimento de aplicações Web semelhante ao *Microsoft Active Server Pages* (ASP), porém tem a vantagem da portabilidade da plataforma podendo ser executado em outros Sistemas Operacionais além dos da Microsoft. Ela permite ao desenvolvedor de sites produzir aplicações que permitam o acesso a banco de dados, o acesso a arquivos-texto, a captação de informações a partir de formulários, a captação de informações sobre o visitante e sobre o servidor, o uso de variáveis e *loops* entre outras coisas.

“JSP é uma tecnologia baseada em Java que simplifica o processo de desenvolvimento de sites da web dinâmicos”. (DUANE K. FIELDS; MARK A. KOLB, 2000, p.2).

Para Kurniawan (2002), *Java Server Pages* é uma outra tecnologia Java para desenvolver aplicativos web.

“Java Server Pages (JSP) é uma outra tecnologia Java para desenvolver aplicativos web. JSP foi lançada quando a tecnologia de servlet tinha atingido popularidade como uma das melhores tecnologias disponíveis. JSP, porém, não se destina a substituir servlet. Na verdade, JSP é uma extensão da tecnologia de servlet, e é prática comum usar ambas, servlet e páginas JSP nos mesmo aplicativos web”. (KURNIAWAN, 2002, p.217).

Além de JSP uma tecnologia que está intimamente ligada a ela é *Servlet*. *Servlet* é uma outra forma para desenvolver aplicativo Web usando Java. O autor ainda explica que se você planejar montar um aplicativo só com JSP, é interessante olhar com atenção a tecnologia *servlet*.

“Servlet, tecnologia que é a base do desenvolvimento de aplicativo web usando a linguagem de programação Java. Ela é uma das tecnologias Java mais importantes, e é a tecnologia subjacente para outra tecnologia Java popular para desenvolvimento de aplicativo: *JavaServer Pages* (JSP)”. (KURNIAWAN, 2002, p.3).

“Ainda que você planeje desenvolver o seu aplicativo web Java usando apenas páginas JSP, entender a tecnologia servlet ajuda a montar um aplicativo JSP mais eficiente e efetivo”. (KURNIAWAN, 2002, p.3).

2.1.5 Java Database Connectivity - JDBC

De modo geral, um banco de dados é uma coleção de objetos que mantêm e manipulam dados. Um sistema de banco de dados é muito importante para qualquer que seja o sistema na web ou não, pois será num sistema de banco de dados que serão guardado informações relacionados a aplicação, cadastro de clientes, cadastro de fornecedores, recuperação de dados etc.

“Há muito um baluarte de grandes empreendimentos, os banco de dados encontraram seu lugar em uma gama muito mais ampla de sites web em anos recentes”. (FIELDS; KOLB, 2000, p. 182).

Java permite o acesso e manipulação de dados através do JDBC. Usando JDBC o usuário será capaz de inserir, remover, alterar e consultar dados. Em Java o que nos permite acessar um banco de dados é a API de banco de dados de Java – JDBC. É necessário fazer com que aplicações se comuniquem com o banco de dados.

“Quando uma aplicação JSP precisa se comunicar com o banco de dados, ela o faz através de uma classe de driver fornecida pelo fornecedor gravada na API JDBC”. (FIELDS; KOLB, 2000, p. 182).

Acessar um banco de dados em JSP, não é tão complexo quanto se imagina; simplesmente se atenha à API JDBC da Sun.

2.2 SISTEMA GERENCIADOR DE BANCO DE DADOS - SGBD

Um sistema gerenciador de banco de dados consiste em uma coleção de dados inter-relacionados e uma coleção de programas para prover o acesso a estes dados, provendo um ambiente adequado e eficiente para uso no armazenamento e recuperação das informações.

Um SGBD visa acabar com a inconsistência e redundância de dados, um sistema de banco de dados mal projetado pode trazer sérios danos à uma empresa e instituições, uma vez que todas as informações de uma empresa é armazenada num banco de dados e perdê-los ou danificá-los pode significar um golpe fatal.

“Um *Sistema Gerenciador de Banco de Dados (SGBD)* é constituído por um conjunto de dados associados a um conjunto de programas para acesso a esses dados. O conjunto de dados, comumente chamado de banco de dados, contém informações sobre uma empresa em particular. O principal objetivo de um SGBD é proporcionar um ambiente tanto para *conveniente* quanto *eficiente* para a recuperação e armazenamento das informações do banco de dados”. (KORTH; SILBERSCHATZ; SUDARSHAN, 1999, p. 1).

Date (2000), argumenta que em essência, um sistema de banco de dados é apenas um *sistema computadorizado de armazenamento de registros*. O banco de dados pode, ele próprio, ser visto como o equivalente eletrônico de um armário de arquivamento. Em outras palavras, é um repositório ou recipiente para uma coleção de arquivos de dados computadorizados.

2.2.1 Modelo Entidade-Relacionamento – (E-R)

Modelos são representações simplificadas de objetos, pessoas, itens, tarefas, processos, conceitos, idéias, etc, usados comumente por pessoas no seu cotidiano, independente do uso de computadores.

O modelo inclui características que são consideradas importantes para o seu uso e omitem outras. Para modelar algo nós precisamos abstrair algo do mundo real e pensar isoladamente como um componente de um todo.

Abstração é a habilidade de esconder detalhes e concentrar-se no geral, nas propriedades a um conjunto de objetos. Na atividade de modelagem de dados, a habilidade de abstrair é essencial e deve ser exercida de maneira sistemática.

Quando pensamos, por exemplo, em uma pessoa, o que logo nos vem em mente é o nome, o estereótipo, etc., entretanto quando abstraímos a idéia de pessoa e pensamos em pessoa como um modelo de uma entidade, nós imaginamos pessoa de outra forma, quais são as informações que realmente interessam para que possamos identificar uma pessoa? o nome, identidade, CPF, endereço, etc.

Existem vários tipos de modelos quando se trabalha com banco de dados, mas vamos nos limitar a trabalhar com o Modelo de Entidade-Relacionamento, cujo é o modelo que vamos implementar em nossa pesquisa.

Uma entidade é um objeto que é distinguível de um outro objeto por um conjunto específico de atributos. Um relacionamento é uma associação de várias entidades.

“O modelo de dados entidade-relacionamento (E-R) tem por base a percepção do mundo real como um conjunto de objetos básicos, chamados entidades, e do relacionamento entre eles. (KORTH; SILBERSCHATZ; SUDARSHAN, 1999, p. 1).

Korth, Silberschatz e Sudarshan (1998), definiram que uma entidade é uma “coisa” ou um “objeto” do mundo real que pode ser identificado por outros objetos.

Para Date (2000), entidade é usado comumente no meio de banco de dados para indicar qualquer objeto distinguível que deva ser representado no banco de dados.

2.2.2 A linguagem SQL – *Structured Query Language*

A linguagem SQL tem sido tão utilizada que pode ser considerada um padrão para Sistemas de Gerência de Banco de Dados relacionais. Consiste de uma série de declarações, adotadas de comum acordo, que nos permitem realizar diversas operações.

Temos que usar a expressão “comum acordo” porque, embora uma SQL padrão tenha sido criada pelo Instituto de Padrões Nacionais Americanos (ANSI), todas as implementações particulares da SQL personalizam a linguagem de várias formas. Tais implementações complementam a linguagem padrão com novos tipos de declarações ou expressões e muitas vezes adaptam as declarações padronizadas às necessidades específicas.

“Certamente a SQL tem representado o padrão para linguagens de banco de dados relacionais. A versão original foi desenvolvida pela IBM no Laboratório de Pesquisa de San José (hoje o Centro de Pesquisa Almaden). Essa linguagem, originalmente chamada de Sequel, foi implementada como parte do projeto do Sistema R no início dos anos 70. Desde então, a linguagem Sequel foi evoluindo e seu nome foi mudado para SQL (Structured Query Language – Linguagem de Consulta Estruturada). (KORTH; SILBERSCHATZ; SUDARSHAN, 1999, p. 1).

Date (2000), defende que a SQL é a linguagem padrão para se lidar com banco de dados relacionais, e é aceita por quase todos os produtos existentes no mercado.

A linguagem SQL é usada de duas formas **embutida** em programas de aplicativos ou através de declarações SQL executadas diretamente a partir do teclado **interativa**. O enfoque do nosso projeto será sobre SQL usada na forma embutida uma vez que estamos interessados em fazer um aplicativo para web.

Julgamos oportuno chamar a atenção para o fato de que a linguagem SQL não é uma linguagem com características de uma linguagem de programação. Na realidade ela é uma linguagem de manipulação de dados (DML – Data Manipulation Language), uma linguagem de definição de dados (DDL – Data Definition Language) e uma linguagem de controle de dados (DCL), concedendo e retirando autorizações sobre os objetos do banco.

2.3 UTILIZANDO UML, PADRÕES E ANÁLISE ORIENTADA A OBJETOS

Todo software bem projetado antes de ser implementado, precisa ser analisado rigorosamente, para não haja surpresas desagradáveis no futuro. Para criar o software de uma aplicação, é necessária uma descrição do problema e dos seus requisitos – o que é o problema e o que o sistema deve fazer.

Para Larman (2000), a análise enfatiza uma *investigação* do problema, de como uma solução é definida.

“UML” é a abreviação de Linguagem de Modelagem Unificada (Unified Modeling Language), uma notação (principalmente diagramática) para modelagem de sistemas, usando conceitos orientados a objetos”. (LARMAN, 2000, p. 28).

“A essência da análise orientada a objetos, é enfatizar a consideração de um domínio de problema e uma solução lógica, segundo a perspectiva de objetos (coisas, conceitos ou entidades). (LARMAN, 2000, p. 30).

A análise orientada a objeto vai nos ajudar atribuir funções, como devem ser alocadas responsabilidades às classes de objetos, como objetos devem interagir, quais classes devem fazer o quê. Estas são questões críticas no projeto de um sistema.

De acordo com Larman (2000), a habilidade individual mais importante na análise e no projeto orientados a objetos é a atribuição habilidosa de responsabilidade aos componentes de software.

Porque esta é a atividade que deve ser executada (inevitavelmente) e que tem o efeito mais profundo sobre a robustez, a facilidade de manutenção e a re-usabilidade dos componentes de software.

2.4 ARQUITETURA DE SISTEMAS

Um aplicativo de software bem projetado é dividido em partes lógicas separadas, chamadas camadas. Cada camada tem uma responsabilidade diferente na arquitetura geral. Essas camadas são simplesmente abstrações e não correspondem à distribuição física.

O que torna essa abordagem diferente é o fato de que cada uma de suas camadas trabalha de forma separada e inteligente, levando em consideração a estrutura e a semântica de cada função a ser exercida. Isso é um imenso salto de qualidade.

Segundo Kurniawan (2002), essas são as camadas típicas em um software de sistema:

- **Camada de apresentação.** Nessa camada estão as partes que gerenciam a interface e a interação de usuário.
- **Camada de lógica comercial.** Essa camada contém componentes que gerenciam a lógica da programação do aplicativo.
- **Camada de dados.** Essa camada é usada pela camada comercial lógica para manter dados permanentemente. Normalmente, essa camada consiste em um ou mais banco de dados, onde os dados são armazenados. Entretanto, outros tipos de armazenagem de dados poderiam ser usados. Por exemplo, agora é muito comum usar documentos XML como armazenagem para manter dados.



Figura 2: Modelo de camadas

Traduzindo esse modelo abstrato para um modelo de código J2ME, obtém-se:

Interface do usuário

- HTML / JSP

Negócios

- Classes Java

Persistência

- Classes Java
- Padrão de projeto DAO

Antes de ser criado o modelo de a arquitetura de três camadas faziam aplicativos em uma arquitetura de duas camadas, o que não era muito interessante, pois o processamento do lado cliente ficava cheio.

“Um aplicativo de duas camadas é um simples aplicativo cliente-servidor, onde entra o processamento de carga de trabalho nos ombros do computador cliente, e o servidor simplesmente atua como um controlador de tráfego entre o cliente e os dados. O termo “fat client” (cliente cheio) para esse tipo de arquitetura é devido ao volume de exigências de processamento do lado do cliente. Nessa arquitetura, a camada de apresentação e a camada de lógica comercial são hospedadas em uma camada, e a camada de dados”. (KURNIAWAN, 2002, XVIII).

A desvantagem desse tipo de arquitetura é que ela começa a apresentar problemas quando o número de clientes aumenta. O primeiro problema é devido ao fato que todo o processamento ocorre no lado cliente. Há um tráfego de rede aumentado, pois cada cliente precisa fazer múltiplas solicitações da dados a partir do servidor – mesmo antes de apresentar qualquer coisa ao usuário.

Um outro problema é o custo, pois cada cliente precisa de uma máquina com suficiente potência de processamento. À medida que o número de clientes aumenta, o custo só para oferecer máquinas cliente pode ser astronômico.

Para Kurniawan 92002), o problema mais sério, que esse tipo de arquitetura pode causar, provavelmente, é um problema de manutenção. Mesmo uma pequena mudança na lógica de processamento pode exigir uma alteração completa de toda a organização. Ainda que o processamento possa ser automatizado, sempre existem problemas em organizações maiores, porque alguns usuários podem não estar prontos para um *upgrade*, enquanto outros insistem que ele seja feito imediatamente.

Para controlar os problemas em muitos aplicativos de duas camadas, um aplicativo é dividido em três camadas separadas, ao invés de duas. A primeira contém a camada de apresentação, a segunda, ou camada central, consiste na camada de lógica comercial, e a terceira contém dados.

Segundo Kurniawan (2002), para conseguir mais flexibilidade, as três camadas no aplicativo de três camadas podem ser ainda mais segregadas. Um aplicativo com esse tipo de arquitetura é chamado de *n-tiered* (n-camadas). Nessa arquitetura, a camada comercial lógica é dividida pela função, em vez de ser fisicamente fragmentada. Ela se divide da seguinte maneira:

- **Uma interface de usuário.** Gerencia a interação de usuário com o aplicativo. Em um aplicativo de internet, normalmente isso é um browser de web usado para apresentar tags (guias) HTML.
- **Lógica de apresentação.** Define o que a interface de usuário define e como são tratadas solicitações de usuário.
- **Lógica comercial.** Modela a lógica comercial de aplicativo.
- **Serviços de infra-estrutura.** Oferecem a funcionalidade adicional exigida pelos componentes de aplicativo.
- **A camada de dados.** Hospeda os dados de aplicativo.

Podemos observar que para desenvolver qualquer aplicativo na web é de vital importância saber organizar, dividir responsabilidades e a funcionalidade de cada parte do projeto. Dividindo tarefas de forma coerente, é muito mais fácil a manutenção, evitando futuros transtornos para o desenvolvedor e para o cliente, fazendo com que seja uma forma de melhor atender as necessidades do mesmo.

2.5 APLICATIVO WEB

O aplicativo Web foi construído usando o padrão MVC – *Model View Controller* (Modelo, Visualização e Controlador). Esse padrão prega a separação de cada uma dessas camadas, permitindo assim uma manutenção mais fácil bem como reusabilidade do software.

Na arquitetura MVC o modelo representa os dados da aplicação e as regras do negócio que governam o acesso e a modificação dos dados. O modelo mantém o estado persistente do negócio e fornece ao controlador a capacidade de acessar as funcionalidades da aplicação encapsuladas pelo próprio modelo.

Um componente de visualização renderiza o conteúdo de uma parte particular do modelo e encaminha para o controlador as ações do usuário; acessa também os dados do modelo via controlador e define como esses dados devem ser apresentados.

Um controlador define o comportamento da aplicação, é ele que interpreta as ações do usuário e as mapeia para chamadas do modelo. Em um cliente de aplicações Web essas ações do usuário poderiam ser cliques de botões ou seleções de menus. As ações realizadas pelo modelo incluem ativar processos de negócio ou alterar o estado do modelo. Com base na ação do usuário e no resultado do processamento do modelo, o controlador seleciona uma visualização a ser exibida como parte da resposta a solicitação do usuário.

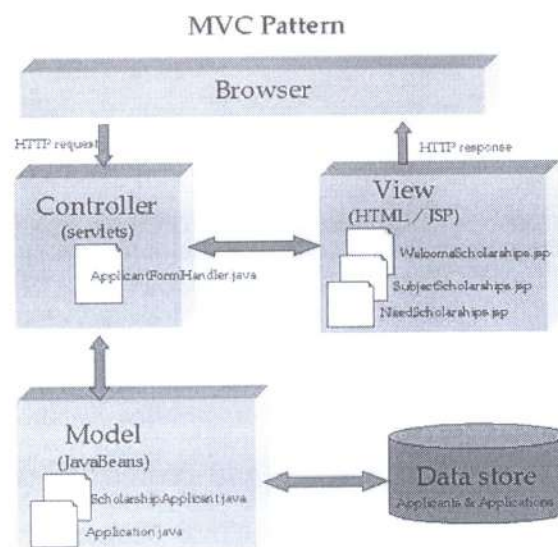


Figura 3: Padrão MVC
Fonte: Java One 2004 – TS-0226

2.6 PADRÕES DE PROJETO

Os padrões de projeto foram originalmente criados na arquitetura, pelo arquiteto Christopher Alexander. Os padrões de projeto tornam mais fácil reutilizar projetos e arquiteturas bem-sucedidas. Os padrões de projeto ajudam a escolher alternativas de projeto que tornam um sistema reutilizável. Em suma, ajudam um projetista a obter um projeto “certo” mais rápido. [Gamma, Helm, Johnson, Vlissides].

Os seguintes padrões foram utilizados no projeto:

- **DAO** (*Data Access Object*): Permite abstrair o acesso direto aos dados. O acesso aos dados fica em um nível mais alto, passa-se a utilizar objetos e enviar os mesmos para a camada de persistência, que a mesma trata de persisti-los com algum mecanismo.
- **Factory** (Fábrica): Delega a responsabilidade para classes específicas a criação de objetos.
- **Singleton** (Única instância): Define que só existirá na memória uma única instância de determinada classe.

2.7 FERRAMENTAS UTILIZADAS

A seguir, relataremos as ferramentas de software utilizadas para o desenvolvimento do aplicativo web.

- **JDK 1.5 (Java Development Kit)**: Trata-se da plataforma padrão do Java. Fundamental para o desenvolvimento de aplicativos Java.
- **NetBeans 5.0 Beta**: IDE livre da Sun para criação de projetos Java.
- **Bundled Tomcat**: Integrado a IDE NetBeans 5.0. É um servidor capaz de suportar o desenvolvimento e a execução em ambiente de produção de aplicações web criadas segundo os padrões da plataforma Java.
- **JSP 2.0**: Utilizado para construir o aplicativo Web seguindo o padrão MVC.
- **Eclipse versão 3.1**: É um ambiente de desenvolvimento Java *open source* mais usado pelos desenvolvedores.
- **MySQL 5.0**: É um banco de dados relacional livre e muito veloz. Utilizado por inúmeras empresas de médio e até grande porte.

- **Power Design 9:** Ferramenta para modelagem de Sistema Orientado a Objetos.
- **Rational Rose 2000 Enterprise Edition:** Utilizado para modelagem do Sistema em UML.
- **DBDesigner 4:** Utilizado para modelagem do Banco de Dados.
- **Macromedia Dreamweaver 8:** Ferramenta de desenho de páginas da Web dentro de um contexto de site.

3 ANÁLISE E PROJETO DO SISTEMA

Neste capítulo, apresentaremos a modelagem do sistema objeto desta monografia, na linguagem UML. Serão apresentados os diagramas UML de Casos de Uso, e de Classes para especificação, documentação, visualização e desenvolvimento do sistema.

3.1 VISÃO GERAL DO SISTEMA

Esta fase é destinada à construção do modelo conceitual do sistema, onde o foco é voltado para o domínio do problema.

De acordo com Jones (2000), esta fase de análise está preocupada com as primeiras abstrações e mecanismos que estarão presentes no domínio do problema. Na análise, só serão modeladas classes que pertençam ao domínio principal do problema do software, ou seja, classes técnicas que gerenciem banco de dados, interface, comunicação, concorrência e outras que estarão presentes nos diagramas UML.

No projeto do sistema Helpdesk para a RMNor do Brasil, foi identificada uma série de procedimentos que fizeram parte do desenvolvimento deste sistema, e que foi levado em consideração para a construção do modelo conceitual, como será exibida abaixo:

- Um *Usuário* (funcionário) comum poderá se cadastrar no sistema, obtendo uma senha poderá ter acesso a informações de uso geral da aplicação, como visualizar chamados já abertos, abrir novo chamado, consultar artigos, ver links;
- Um *Administrador* pode cadastrar artigo, mudar status da ordem de serviço, por exemplo, o administrador pode declarar como encerrada, pendente, ou em espera a ordem de serviço;

Diante destas considerações, se pode à primeira vista, considerar o Administrador, e o funcionário como usuários internos do sistema.

3.2 DIAGRAMA DE CASO DE USO

Casos de uso (do inglês *Use Case*) são utilizados para identificar as regras de negócio e são uma excelente forma de entender o ponto de vista do usuário simplesmente pelo fato de que modela o que ele precisa executar.

Internamente, um caso de uso é uma sequência de ações que permitem a execução completa de um comportamento esperado para o sistema.

De acordo com Alexandre Veloso (2002), um caso de uso não deve se preocupar com o “como” das funções, mesmo que este “como” seja apenas conhecer o fluxo de entrada e saída de dados. É importante encapsular ações que em conjunto sejam relevantes e que de alguma forma influenciam algo ou alguém no sistema.

A figura 4 mostra o diagrama de casos de uso para o sistema helpdesk.



Figura 4: Casos de Uso do Sistema

3.2.1 Formatos de casos de usos textuais

Um caso de uso pode ser especificado de diversas formas, pois não há um padrão estabelecido para essa atividade. Sendo assim, os casos de usos textuais podem ser divididos em duas categorias:

- **Caso de uso de alto nível** – É uma forma de expressar o comportamento pretendido para um sistema, sem se aprofundar em seu detalhamento, ou seja, sob uma perspectiva macro.

- **Caso de uso detalhado** - Descreve os processos com mais detalhes, com finalidade de se obter uma compreensão dos requisitos. São utilizados em estilo "convencional" entre os atores e o sistema. O objetivo desse detalhamento é descrever a seqüência de passos que deve ocorrer para o respectivo caso de uso sem a omissão de qualquer informação.

Segundo Larman (2000), embora a estrutura de casos de usos textuais sejam típicos, a UML não especifica um formato rígido, ou seja, ele pode ser alterado para satisfazer a clareza de comunicação.

3.2.2 Diagramas de Casos de Uso de alto nível

A seguir será descrito cada caso de uso com uma especificação de alto nível para uma melhor compreensão do processo.

- **Cadastrar usuário** – O usuário entra no site da RMNor a partir de um *browser*, o sistema exibe a página inicial, e a partir do momento que o usuário queira executar qualquer função da aplicação ele terá que se cadastrar para isso.
 - Caso de Uso: **Cadastrar usuário**
 - Atores: Usuário
 - Finalidade: Apresentar ao visitante a tela de cadastro no site, com os campos pré definidos que o usuário terá que preencher para efetuar o cadastro.
 - Visão Geral : O usuário interessado em se cadastrar no site, terá que informar a empresa, o setor, seu e-mail, digitar e confirmar sua senha. O sistema então exibirá uma mensagem confirmando o cadastro do usuário.

Tabela 1: Cenário Caso de uso Cadastrar Usuário

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário com interesse em registrar um chamado, tem que se logar no sistema para fazê-lo, uma vez em que o usuário ainda não está cadastrado no sistema terá que informar dados pré definidos para efetivar seu cadastro.	2. Apresenta a página de cadastro para se registrar no sistema. 3. Exibe uma mensagem confirmando o cadastro do usuário.

SEQUÊNCIAS ALTERNATIVAS	
Item 1 – o usuário não preenche todos os campos necessários para efetivar o cadastro	

- **Efetuar login** – O usuário entra na pagina de login, informa seu e-mail e sua senha, para acessar o sistema.
 - Caso de Uso: **Efetuar login**
 - Atores : Usuário
 - Finalidade: Apresentar ao visitante a tela de login, o usuário informará seu e-mail e sua senha a partir deste momento poderá utilizar o sistema.
 - Visão Geral: O usuário se identifica no sistema, então poderá utilizar os serviços do sistema, como abrir chamado, visualizar chamados já abertos, buscar textos com informações técnicas e ver links.

Tabela 2: Cenário Caso de uso Efetuar Login

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário tem interesse em acessar o sistema, a tela de login é exibida para o usuário, o usuário informa seu e-mail e sua senha definida na hora do cadastro.	2. O sistema permite que o usuário utilize o sistema, com as permissões definidas para usuário comum.
SEQUÊNCIAS ALTERNATIVAS	
Item 1 – o usuário informa e-mail ou senha errada.	

- **Lembrar senha** – O usuário entra na pagina de login, não lembra a senha e pede para que lhe seja enviado um e-mail com essa informação.
 - Caso de Uso: **Lembrar senha**
 - Atores : Usuário/Administrador
 - Finalidade: Enviar um e-mail para o usuário com sua senha.
 - Visão Geral: O usuário não consegue se logar no sistema, pois esqueceu sua senha, o usuário clica no link **lembrar senha**. O sistema pede para que o usuário informe o e-mail com o qual ele se cadastrou. Usuário informará, enviará um e-mail para o Administrador do sistema.

Tabela 3: Cenário Caso de Uso Lembrar Senha

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário tenta se logar no sistema e não consegue, pois esqueceu sua senha. 2. O usuário clica no campo lembrar senha, informará com qual e-mail se cadastrou.	3. Será enviada para o administrador do sistema a solicitação do usuário.
SEQUÊNCIAS ALTERNATIVAS	
Item 2 – o usuário informa erroneamente e-mail com o qual se cadastrou.	

- **Abrir chamado (OS)** –O usuário previamente cadastrado no sistema, poderá abrir um chamado, entende-se como chamado, um problema cujo usuário solicita ao setor de suporte para que seja solucionado.
 - Caso de Uso: **Abrir chamado**
 - Atores: Usuário/Administrador
 - Finalidade: Registrar um chamado, informando categoria, assunto e descrição.
 - Visão Geral: O usuário abrirá a página de suporte, informará a categoria do problema, essas categorias estão pré-definidos em um CHECKBOX, entre as categorias pré-definidas encontram-se: SO (Sistema Operacional), Linx (Sistema ERP utilizado pela empresa), impressora, redes e etc. informará o assunto, e a descrição do problema.

Tabela 4: Cenário Caso de Uso Abrir Chamado

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário abre a página de suporte do sistema. 2. O usuário informa a categoria o assunto e descrição do problema e clica em enviar.	3. Exibe uma mensagem informando que o registro do chamado foi efetivado com êxito.
SEQUÊNCIAS ALTERNATIVAS	
Item 2 – o usuário não seleciona a categoria dentro de um Checkbox.	

- **Listar chamados realizados** – O usuário previamente cadastrado no sistema, poderá listar a os chamados já realizados por ele, assim poderá ver o andamento de cada um, ou seja, seu status..
 - Caso de Uso: **Listar chamados**
 - Atores: Usuário
 - Finalidade: Visualizar os chamados já abertos pelo próprio usuário.
 - Visão Geral: Na tela de menu para o usuário, o usuário poderá ver os chamados abertos por ele mesmo.

Tabela 5: Cenário Caso de Uso Listar Chamados (OS)

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário abre a página de menu do usuário, e deseja visualizar os chamados já abertos por ele.	2. Exibe uma lista com os chamados já realizados por aquele usuário.
SEQUÊNCIAS ALTERNATIVAS	
Item 2 – O usuário nunca registrou um chamado.	

- **Buscar artigo** – O usuário previamente cadastrado no sistema, poderá procurar por artigos técnicos, estes artigos são publicados pelo administrador do sistema.
 - Caso de Uso: **Buscar artigo**
 - Atores: Usuário
 - Finalidade: Procurar artigos técnicos que possam ajudar o usuário a resolver sozinho algum tipo de problema.
 - Visão Geral: Incentivar o usuário a resolver determinados problemas sem a interferência do setor de TI, esses problemas são em grande maioria de simples solução, por exemplo, como gravar arquivos em um servidor na rede, como adicionar um impressora, como enviar um e-mail para uma lista.

Tabela 6: Cenário Caso de Uso Buscar Artigo

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário abre a página inicial do sistema e deseja procurar artigos técnicos, esses artigos são adicionados pelo Administrador do sistema.	2. Exibe uma tela com uma caixa de texto com o título a ser pesquisado.
SEQUÊNCIAS ALTERNATIVAS	
Item 2 – Palavra chave não encontrada	

- **Ver links** – O usuário poderá acessar links que o farão navegar por outros sites interessantes, por exemplo, o site do webmail da empresa, que fica em um servidor localizado na empresa Matriz em São Paulo.
 - Caso de Uso: *Ver links*
 - Atores: Usuário
 - Finalidade: Acessar links que o farão percorrer por outros sites, estes relativos a atividades da empresa.
 - Visão Geral: Na página inicial da empresa existe um campo chamado links, nele o usuário encontrará links relativo ao ramo de negocio da empresa, fornecedores, parceiros, clientes, por exemplo, Renner, C&A, Stalker, Zoomp, Overend, Fórum, Colombo, Vila Romano.

Tabela 7: Cenário Caso de Uso Ver Links

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário abre a página inicial do sistema e deseja procurar links relacionados a atividades da empresa.	2. Exibe um determinado site quando o usuário clica no link correspondente.
SEQUÊNCIAS ALTERNATIVAS	

- **Efetuar logout** – Quando o usuário desejar sair do sistema e encerrar suas atividades no aplicativo.
 - Caso de Uso: *Efetuar logout*
 - Atores: Usuário
 - Finalidade: Encerrar conexão com o sistema.
 - Visão Geral: Assim que desejar o usuário poderá encerrar sua atividades no aplicativo. E se desejar acessar alguma funcionalidade do sistema terá de se logar novamente. Vemos a ação de efetuar logout, como um princípio de segurança, uma vez que o usuário não efetiva o logout, outra pessoa não autorizada poderá acessar a página de suporte, e solicitar algo se passando por outra pessoa, o que poderia gerar confusão na administração do sistema.

Tabela 8: Cenário Caso de Uso Efetuar logout

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do usuário	Resposta do sistema
1. Começa quando o usuário deseja encerrar suas atividades momentâneas no sistema. Ele efetuará o encerramento da sua sessão naquele momento.	2. Encerra sessão do usuário e volta a tela inicial.
SEQUÊNCIAS ALTERNATIVAS	

- **Cadastrar artigo** – Quando o administrador quiser cadastrar algum artigo no site.
 - Caso de Uso: *Cadastrar artigo*
 - Atores: Administrador
 - Finalidade: Quando o administrador julgar importante ou necessário cadastrar algum artigo técnico que agregue valor ao sistema, e ajude o usuário a manter-se informado sobre as futuras realizações e mudanças no setor de TI, que afetará de maneira direta ou indireta o usuário, o administrador poderá cadastrar artigos.

- Visão Geral: O Administrador irá abrir o menu de administrador e cadastrará os artigos que achar interessante e que de alguma maneira influencie na rotina normal de trabalho, tem como princípio melhorar as atividades do setor de TI, isso é um processo de melhoria contínuo.

Tabela 9: Cenário Caso de Uso Cadastrar Artigo

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do Administrador	Resposta do sistema
1. Começa quando o administrador deseja cadastrar algum artigo, ele entrará no menu destinado a ele, e assim cadastrar o artigo.	2. Confirma inclusão do artigo.
SEQUÊNCIAS ALTERNATIVAS	

- **Mudar status da OS** – Após ser informado da solicitação do usuário, o Administrador poderá mudar o status da OS (Ordem de Serviço) ou chamado.
 - Caso de Uso: *Mudar status da OS*
 - Atores: Administrador
 - Finalidade: O administrador poderá mudar o Status da OS, isso significa dizer que o administrador poderá informar o andamento do chamado, ou seja, se o chamado está pendente, ou se foi resolvido e encerrado.
 - Visão Geral: O Administrador irá abrir o menu destinado a ele, visualizará os chamados abertos pelos usuários, e após resolver o problema, deixar uma mensagem da posição do status.

Tabela 10: Cenário Caso de Uso Mudar status da OS

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do Administrado	Resposta do sistema
1. Começa quando o Administrador abre o menu destinado a ele, e visualiza os chamados abertos pelos usuários. 2. Diante das solicitações dos usuários, o Administrador designa algum funcionário, ou ele próprio para solucionar o problema registrado. Depois o Administrador deixará registrado o status do chamado, como	2. Confirma registro.

resolvido/encerrado ou pendente.	
SEQUÊNCIAS ALTERNATIVAS	

- **Inserir links** –Administrador insere links de parceiros da empresa e empresas públicas.
 - Caso de Uso: ***Inserir links***
 - Atores: Administrador
 - Finalidade: O Administrador insere links de empresas parceiras e públicas, fazendo com que o usuário ganhe tempo, evitando assim que o usuário digite endereços inexistentes.
 - Visão Geral: O Administrador irá abrir o menu destinado a ele, e insere links para que o usuário acesse outros sites de empresas que tenham alguma ligação com a empresa, por exemplo, o setor de Contabilidade/Financeiro precisa estar sempre acessando o site do Governo do Estado, para ver impostos, tributação, o site do Banco do Brasil para pagamento da folha e etc.

Tabela 11: Cenário Caso de Uso Inserir links

SEQUÊNCIAS TÍPICA DO EVENTO	
Ação do Administrador	Resposta do sistema
1. Começa quando o Administrador abre o menu destinado a ele, e deseja inserir links que darão acesso a outros sites parceiros da empresa.	
SEQUÊNCIAS ALTERNATIVAS	

3.3 DIAGRAMAS DE CLASSES

No centro do processo de modelagem de softwares orientados a objeto temos o diagrama de classes. Ele permite modelar as definições de recursos essenciais à operação correta do sistema. As definições envolvem a modelagem da estrutura, relacionamentos e comportamentos de uma entidade.

Para Alexandre Veloso (2002), as classes são importantes por quatro motivos:

- Na verdade, são a real estrutura de dados, ou seja, são o objetivo final para o início da etapa de programação;
- Dar ao programador a noção do domínio do problema
- Projetar uma solução de implementação ao problema;
- Traçar perspectiva de escalabilidade ao sistema (onde e como o sistema pode crescer em termos funcionais).

Os diagramas de classes costumam conter os seguintes itens:

- Classes
- Interfaces
- Colaborações
- Relacionamentos de dependências, generalização e associação.

Assim como os demais diagramas, os diagramas de classes podem conter notas e restrições, podem conter também pacotes ou subsistemas, utilizados para agrupar elementos do seu modelo em um conjunto maior.

Os diagramas de classes são utilizados para fazer a modelagem da visão estática de um sistema. Essa perspectiva oferece principalmente suporte para os requisitos funcionais de um sistema. Na Figura 5 é mostrado o diagrama de classes do sistema.

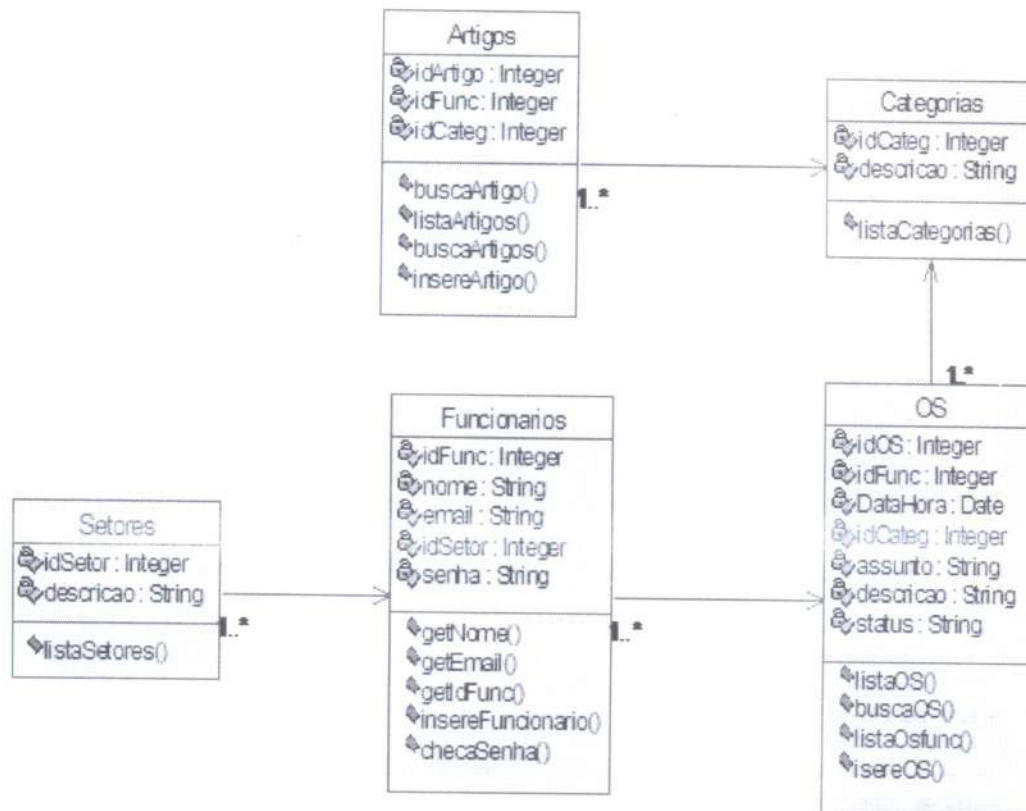


Figura 5: Diagrama de Classe da Aplicação

3.4 MODELAGEM DO BANCO DE DADOS

A Figura 6 exibe o modelo relacional de tabelas construído a partir da modelagem do sistema e do diagrama de classes.

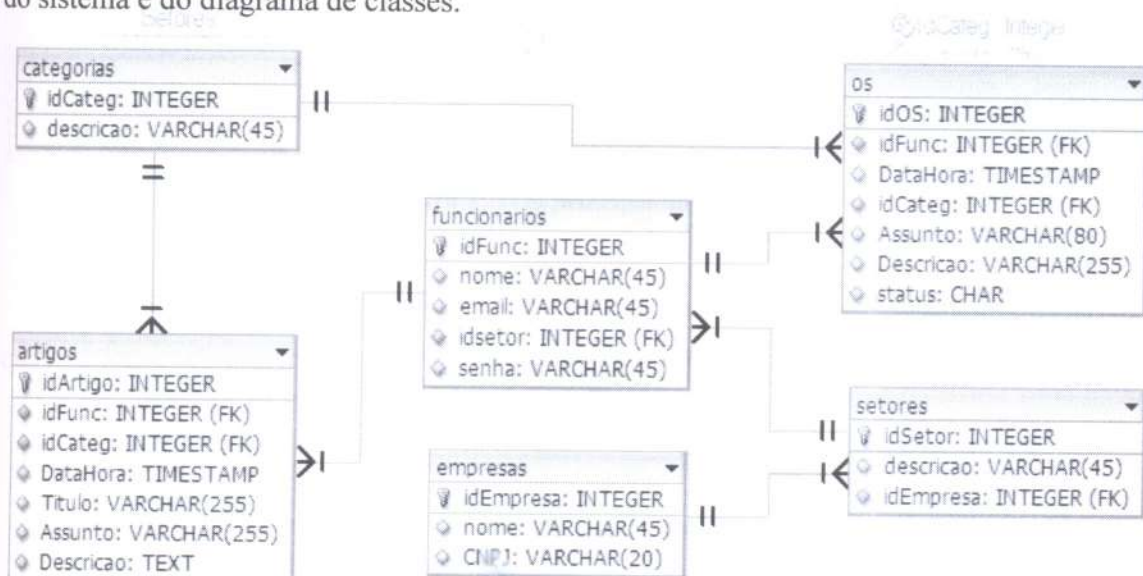


Figura 6: Modelo Operacional do Banco de Dados

3.5 TELAS DO SISTEMA

A seguir, serão mostradas as telas (páginas) do sistema de acordo com os casos de usos levantados. O desenho de todo o site foi feito utilizando *templates* pré-configurados pelo Dreamweaver 8 da Macromedia.

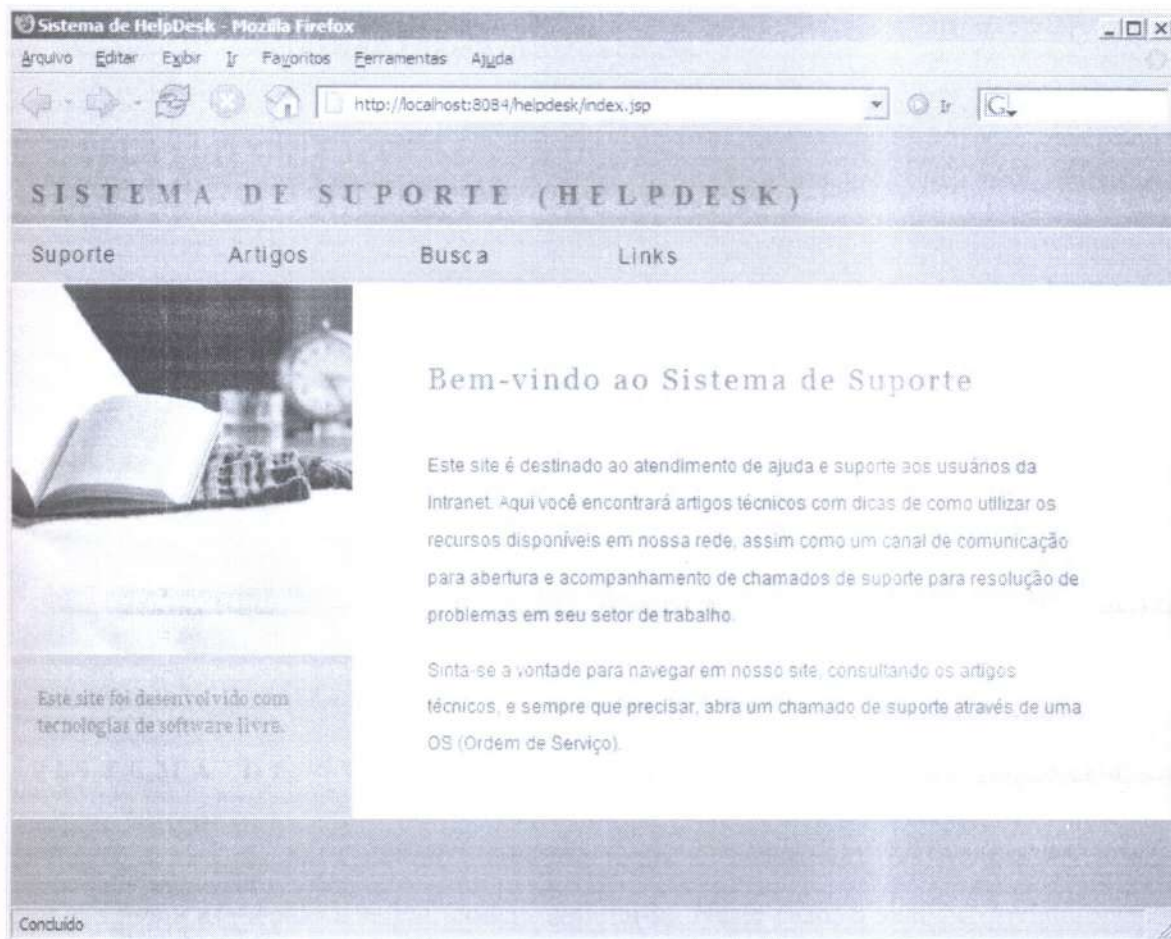


Figura 7: Tela principal do Sistema

Na tela apresentada na figura 7, o usuário poderá ter uma idéia inicial dos serviços que o sistema oferece.

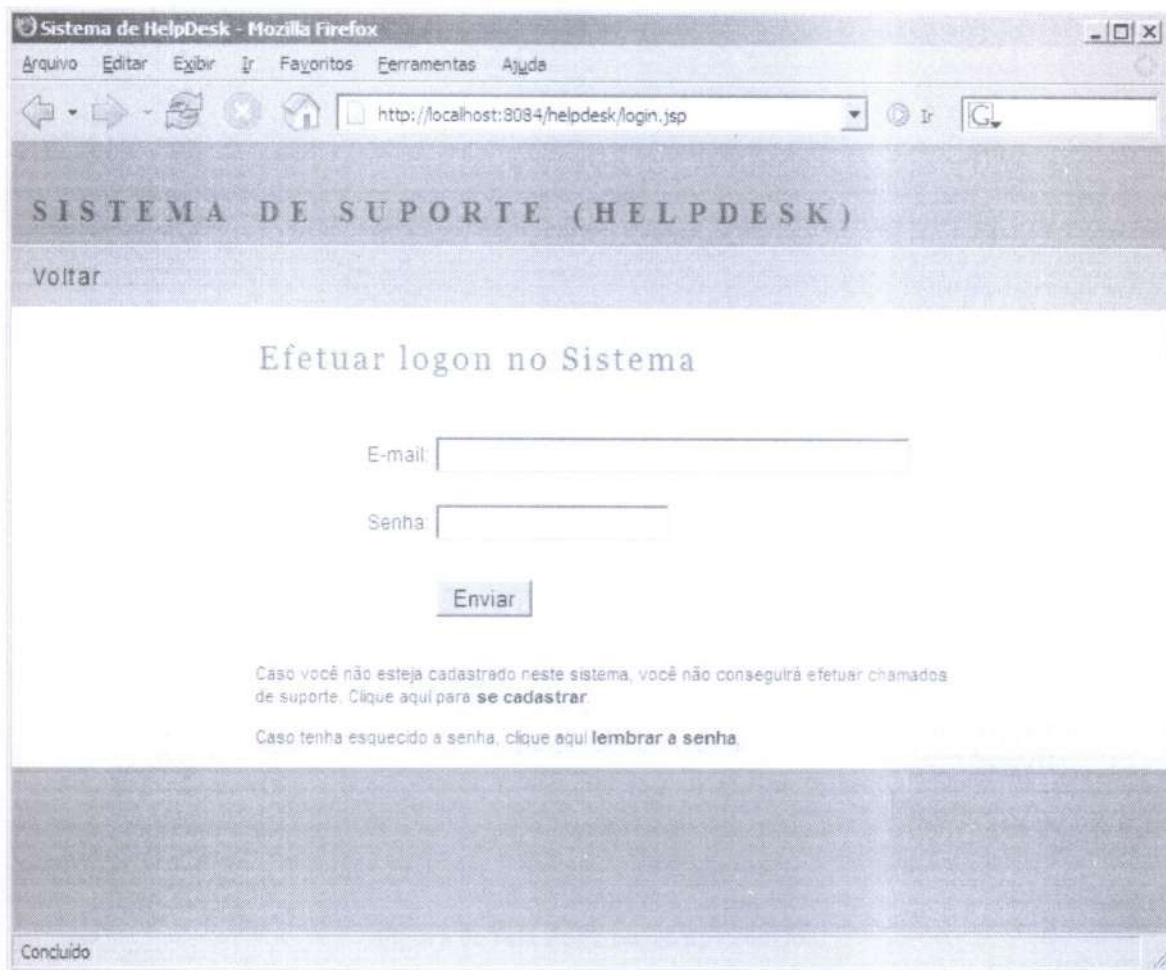


Figura 8: Tela de login do sistema

Na tela apresentada na figura 8, o usuário se identificará no sistema, para ter acesso as suas funcionalidades. Caso ele ainda não tenha se cadastrado, poderá fazê-lo. Ainda, caso ele tenha esquecido a senha, poderá solicitá-la. Neste caso, o sistema enviará a senha para o e-mail do usuário.

The screenshot shows a Mozilla Firefox browser window with the title "Sistema de HelpDesk". The address bar displays "http://localhost:8084/helpdesk/cadastro.jsp". The page has a header with the text "SISTEMA DE SUPORTE (HELPDESK)" and a link "Voltar". The main content area is titled "Cadastrar-se no Sistema" and contains a registration form. The form includes a dropdown menu for "Empresa/Setor" with the text "Selecione setor...", followed by text input fields for "Nome:", "E-mail:", "Senha:", and "Confirmar senha:". Below these fields is an "Enviar" button. At the bottom of the browser window, the status bar shows "Concluido".

Figura 9: Tela de cadastro do usuário

A tela apresentada na figura 9, é destinada aos usuários ainda não cadastrados no sistema. No *menu pull-down* o usuário informará a Empresa/Setor. Em seguida, deverá informar seu nome, e-mail, senha e a confirmação da mesma para evitar digitação errada da senha.

The screenshot shows a Mozilla Firefox browser window titled "Sistema de HelpDesk - Mozilla Firefox". The address bar displays "http://localhost:8084/helpdesk/abrir_os.jsp". The page content is titled "SISTEMA DE SUPORTE (HELPDESK)" and includes a "Voltar" button. The main heading is "Abrir Nova OS (Ordem de Serviço)". The form fields are as follows:

- Solicitante: MARIA DO SOCORRO BARRETO
- Categoria: Sistema Corporativo (dropdown menu)
- Assunto: Obter senha do sistema de almoxarifado
- Descrição: Gostaria de saber como devo proceder para obter uma senha no sistema do almoxarifado central...

Below the description field is an "Enviar" button. At the bottom of the page, there is a status bar that says "Concluído".

Figura 10: Tela para registrar chamado

Na tela apresentada na figura 10, o usuário poderá cadastrar um chamado. Ao efetuar o login, o sistema já informará qual é o solicitante. Cabe ao usuário escolher uma categoria, digitar o assunto, a descrição do problema e enviar.

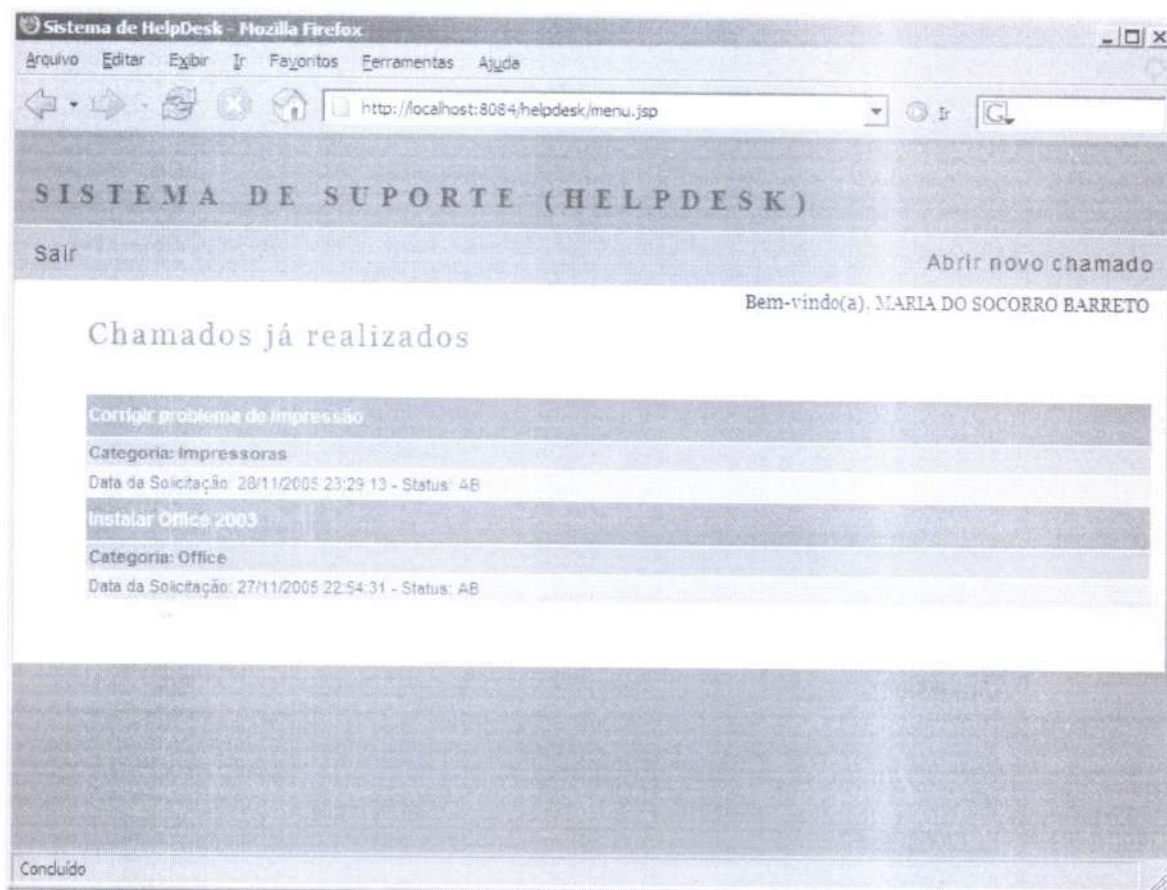


Figura 11: Tela de visualização de chamados já realizados pelo usuário

Na tela apresentada na figura 11, o usuário poderá ver os chamados já realizados por ele. Assim, poderá acompanhar e cobrar do setor de suporte os chamados não atendidos.

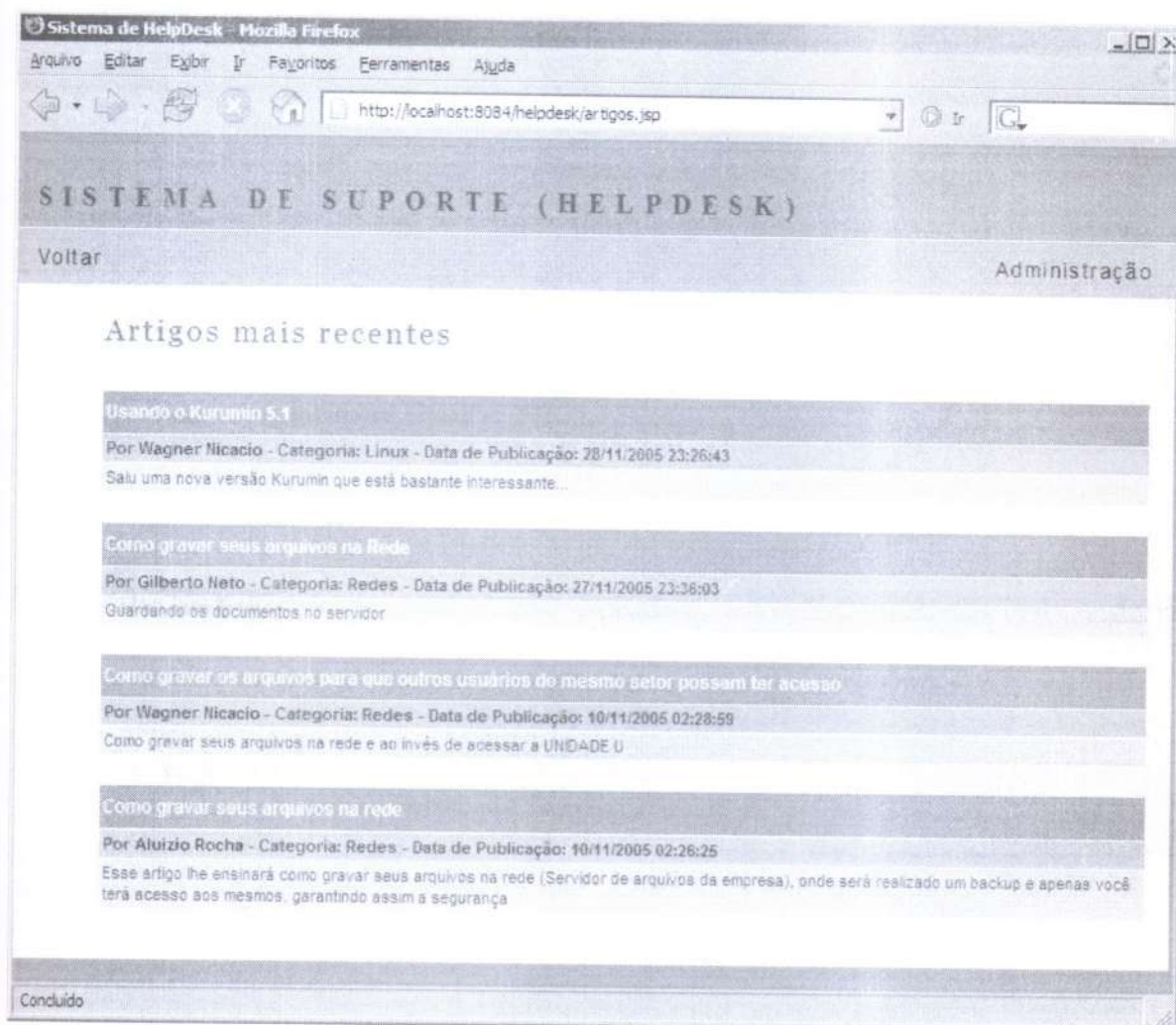


Figura 12: Tela de Artigos

Na tela apresentada na figura 12, estará disponível para que o usuário veja artigos recentes publicados no sistema, e assim incentivá-lo a resolver problemas simples sozinho.

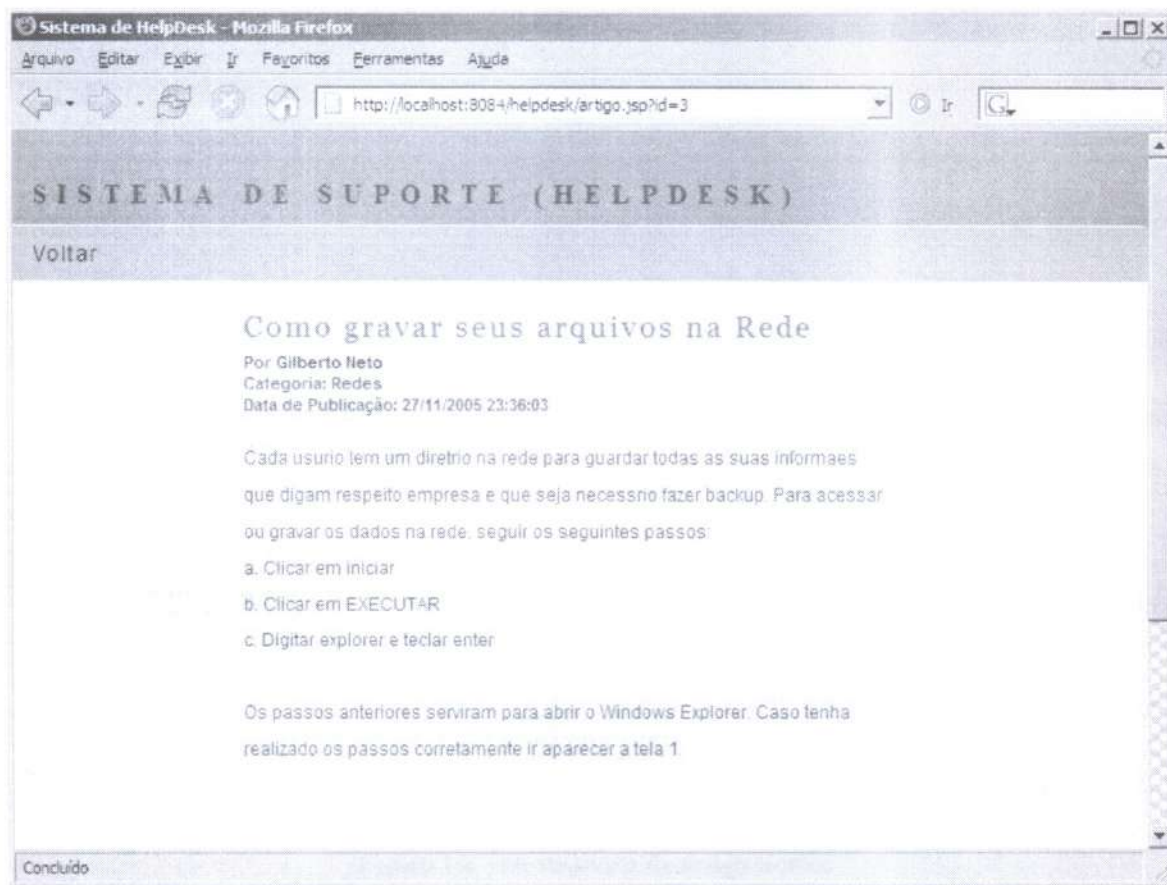


Figura 13: Tela para mostrar artigo selecionado

Na tela apresentada na figura 13, a página disponibilizará para o usuário o artigo escolhido na tela anterior.

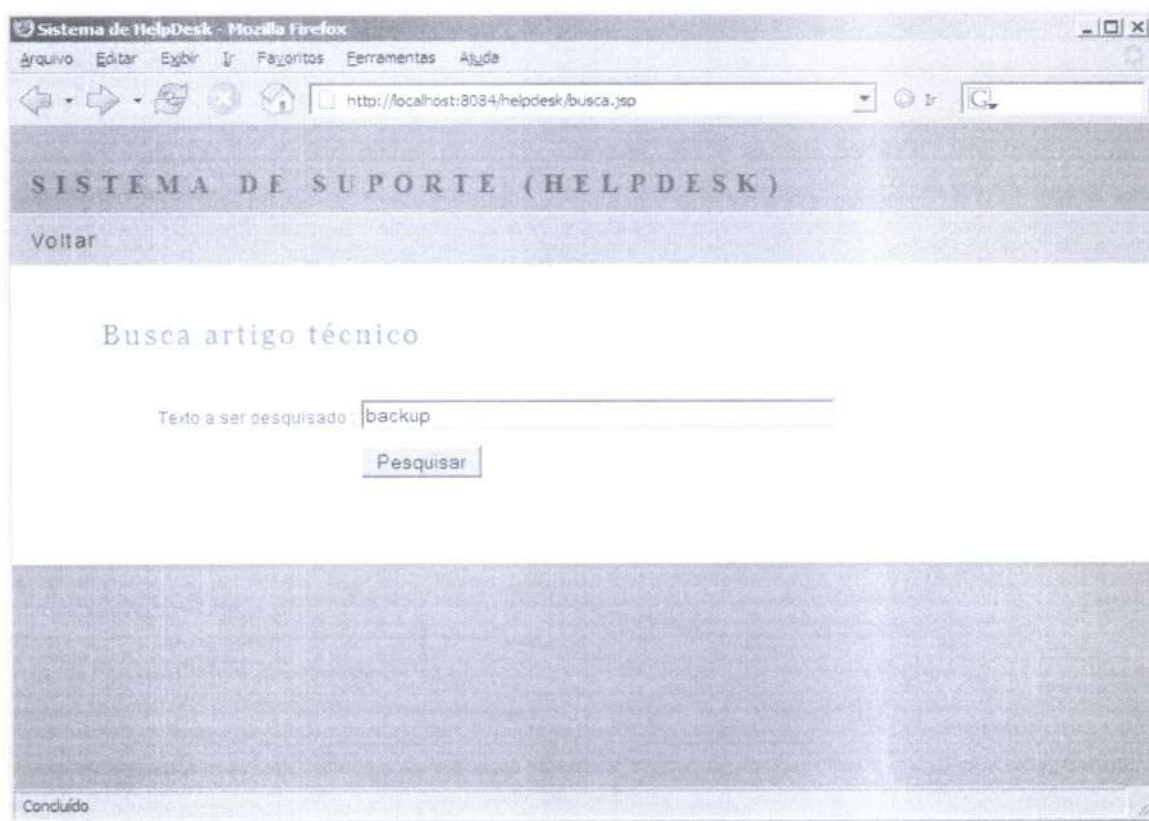


Figura 14: Tela de busca de artigo técnico

Na tela apresentada na figura 14, o usuário poderá buscar artigos técnicos através de alguma palavra-chave. Após digitar a palavra desejada, o usuário clica em pesquisar, e será disponibilizado para ele os artigos técnicos que contém a palavra-chave digitada.

3.6 MAPA DO SITE

A Figura 15 exibe todas as telas do sistema e como poderão ser realizadas as seqüências de navegação.

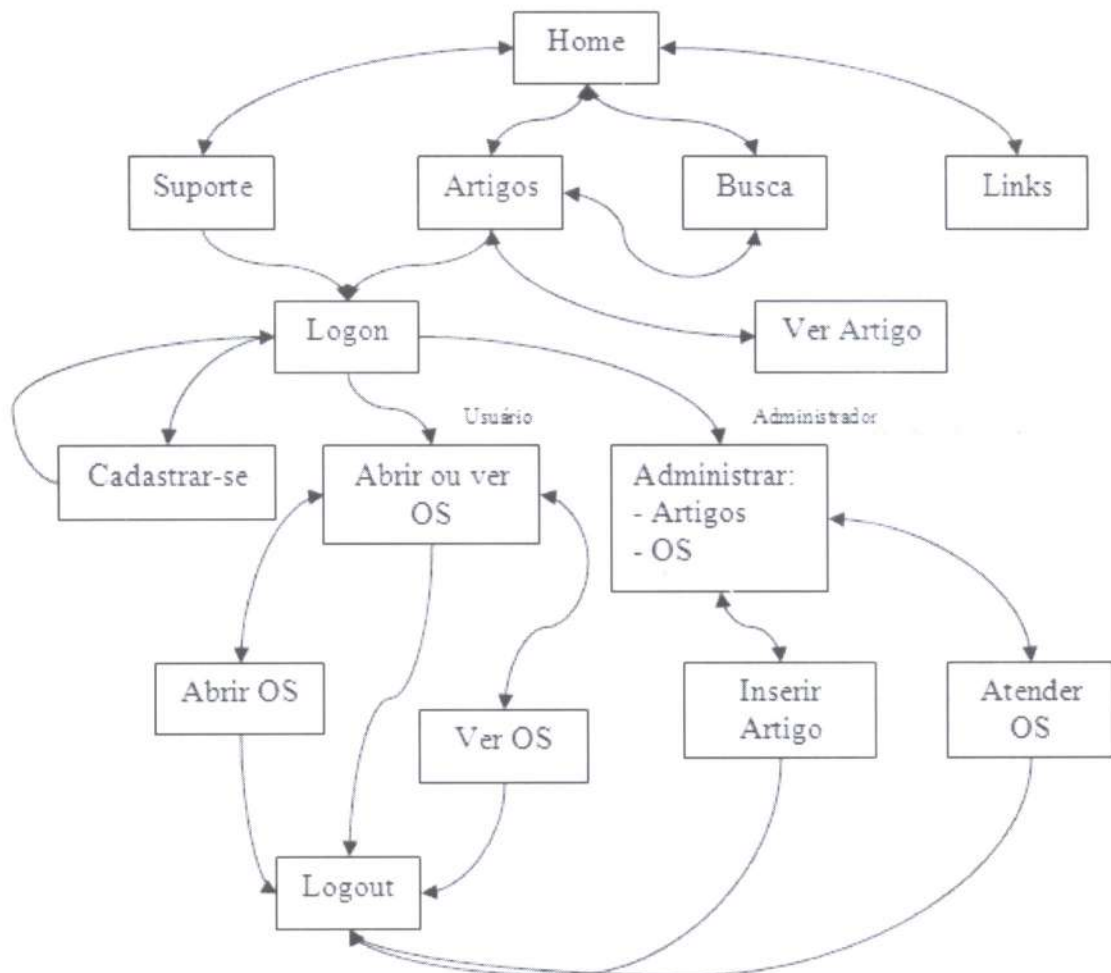


Figura 15: Mapa do Site

4 ANÁLISE DE RESULTADOS

A implantação desse sistema na empresa RMNor do Brasil, foi bem aceita pelos usuários, e sua implantação foi bem sucedida, embora no início houvesse resistência por parte da maioria dos usuários. Para o usuário era mais cômodo ligar para o setor de suporte em TI da empresa, do que acessar o site, se registrar no sistema e informar o suposto problema. Essa resistência acontece devido a uma mudança de hábito de certa maneira imposta pelo setor de TI. Foi necessário educar o usuário a usar o sistema, e mostrar ao usuário que ele é capaz de resolver problemas simples sozinhos. Mostrar ao usuário que ele precisa adquirir a cultura de que a tecnologia trabalha a seu favor e não contra ele. E a partir do momento que o usuário passa a compreender isso, ele se torna mais produtivo.

Esse sistema também serviu de segurança tanto para o administrador quanto para o usuário. Por exemplo, se o usuário liga para o setor de suporte de TI e faz uma solicitação para a resolução de algum problema, ninguém pode garantir que realmente foi feita uma solicitação. Porém, a partir do momento que está registrado no sistema, o usuário pode “exigir” uma posição em relação a sua solicitação. O mesmo acontece para o administrador. Caso o usuário reclame que não foi atendido, o administrador do sistema tem todos os dados da solicitação guardados no BD, ele pode verificar se realmente houve uma solicitação, e se realmente houve, por que não foi solucionada ainda.

Esses são alguns benefícios que o sistema trouxe ao setor de TI e principalmente ao usuário:

- Minimizou o uso de telefone para efetuar solicitações ao setor de TI.
- O usuário interno pode acompanhar toda a “vida” de um chamado, desde sua abertura até a sua conclusão pelo suporte, podendo reclamar caso haja uma demora para solução do mesmo.
- O administrador tem o controle total de todas as solicitações abertas, podendo gerenciá-las da melhor maneira possível. Visualizando os principais problemas que ocorrem dentro da empresa e com que frequência eles ocorrem, e qual o setor e usuário com maior número de ocorrências, e tentar ajudá-lo da melhor maneira possível.

- O usuário passou a se sentir mais confortável com o uso do sistema, as chamadas ao setor de TI diminuí de uma maneira notável, o usuário passou a se interessar mais pelo sistema, o que significa que ele começou a tomar iniciativa para resolução de questões simples.
- O tempo gasto para a implementação do sistema foi de 6 meses

4.1 ANÁLISE SOBRE O DESENVOLVIMENTO

Durante a elaboração do projeto foi feito um estudo sobre as ferramentas, e a própria tecnologia utilizada para o desenvolvimento do sistema, isso agregou valor intelectual ao projeto.

Algumas dificuldades ocorreram no decorrer do projeto, quanto a que ferramentas utilizar, qual SGBD utilizar para guardar e gerenciar informações? Qual a linguagem de programação e por quê? Qual ferramenta utilizar para modelagem do sistema? Qual plugin usar em determinada IDE? Essas questões foram sendo elucidadas aos poucos, diante de tantas ferramentas existentes atualmente no mercado foram testados dois SGBD, o PostgreSQL, e o MySQL, duas IDE's, o Eclipse e o NetBeans, quatro ferramentas para modelagem do sistema, Rational Rose 2000 Enterprise Edition, Power Design 9, JUDE, VP Suíte. Durante o desenvolvimento desse sistema foram lançadas as versões 5.0 do MySQL Server, a versão 3.1 do Eclipse, e a versão beta do NetBeans 5.0, sem contar com os vários plugins lançados para o Eclipse 3.1.

Um dos fatos interessantes do desenvolvimento desse projeto foi a utilização de alguns softwares livres, e descobrir o quão eficiente eles são, e nada deixa a desejar em relação a softwares proprietários. Descobrir ainda que há milhares de pessoas no mundo que usam esses softwares e compartilham conhecimento através de fóruns, tutoriais publicados, matérias publicadas em revistas, e o usuário ou desenvolvedor não precisa pagar nada por isso.

5 CONCLUSÃO

É preciso inovar, pois a cada dia que passa o cenário das tecnologias mudam, e cada vez mais os clientes, exigem um nível de qualidade cada vez maior fazendo com que as empresas melhorem a qualidade de seus produtos e serviços.

Nas empresas que usam a informática para atividades cotidianas problemas sempre acontecem, sejam eles problemas de hardware, problemas de software, configuração de sistemas, etc. Independentemente do papel que a informática representa na empresa, seja ela usada em um cenário de reviravolta, ou em um cenário de suporte, tais problemas sempre acontecerão. Por isso é importante criar um mecanismo em que o usuário possa recorrer quando tiver problemas. Por essa razão fica claro que é necessário criar um sistema que gerencie esses problemas, um sistema de suporte ao usuário.

Com o conhecimento adquirido ao desenvolver esse trabalho, surgirá a necessidade intelectual de aperfeiçoar o sistema e desenvolver outros, surgirão novas tecnologias e é importante estar atento a elas, isso não quer dizer que a cada tecnologia que surgir o desenvolvedor deva migrar seu sistema para elas, e sim estar antenado ao que está acontecendo ao seu redor.

Desenvolver esse trabalho foi uma experiência produtiva. A plataforma de desenvolvimento na qual esse projeto foi implementado, apresenta inúmeras possibilidades de desenvolvimento de software, desde soluções para *palmtops* a mainframes. É possível obter maior conhecimento e domínio desta tecnologia para projetos futuros.

5.1 CONSIDERAÇÕES FINAIS

Sistemas para o web vêm se destacando cada vez mais no cenário de desenvolvimento de software, é importante estar atento e apto para entrar nesse mercado. Essa aplicação para web mostrou uma ampla possibilidade para desenvolver softwares com mais qualidade e de maneira mais simples e objetiva.

Esse projeto mostra que é possível desenvolver qualquer coisa desde que exista motivação, quando se está motivado as coisas fluem de maneira equilibrada. O projeto fez perceber que por mais que pareça simples, um problema, é um pesadelo para um usuário, isso leva a refletir sobre como o desenvolvedor deve pensar ao projetar o software.

Um sistema de HelpDesk é de grande importância para empresas que trabalham com informática, ou seja, quase todas as empresas do mundo, esse sistema tornou a vida do setor de suporte em TI da RMNor mais fácil e menos estressante, isso não quer dizer que não ocorram mais problemas, no entanto eles passaram a ser controlados com mais eficiência, e soluções mais rápidas. Para o usuário esse sistema se mostrou também eficiente, pois o mesmo não precisa ficar ligando a todo momento para o setor de TI, basta que ele acesse de qualquer terminal da empresa o sistema, e registre seu problema.

REFERÊNCIAS BIBLIOGRÁFICAS

- FIELDS, Duane K; KOLB, Mark A. Desenvolvendo na Web com JavaServer Pages. Rio de Janeiro: Ciência Moderna, 2000.
- LEMAY, Laura; CADENHEAD, Rogers. Aprenda em 21 dias Java 2. Rio de Janeiro: Campus, 1999.
- DEITEL. H; DEITEL P. J. Java Como Programar. 3. ed. Porto Alegre: Bookman, 2001.
- KURNIAWAN, Budi. Java para a Web com Servlets, JSP e EJB. Rio de Janeiro: Ciência Moderna, 2002.
- SILBERSCHATZ, Abraham; KORTH, Henry; SUDARSHAN, S. Sistema de Banco de Dados. 3. ed. São Paulo: MAKROM Books, 1999.
- DATE, C. J. Introdução a Sistemas de Banco de Dados. 7. ed. Rio de Janeiro: Campus, 2000.
- LARMAN, Craig. Utilizando UML e Padrões: uma introdução à análise e ao projeto orientados a objetos. Porto Alegre: Bookman, 2000.
- SANTOS, Rafael. Introdução a Programação Orientada a Objetos Usando Java. Rio de Janeiro: Campus, 2003.
- ANSELMO, Fernando. Aplicando Lógica Orientada a Objetos em Java. Florianópolis: VisualBooks, 2003.
- ANSELMO, Fernando. Tudo o que você queria saber sobre a JDBC. Florianópolis: VisualBooks, 2001.

- ANSELMO, Fernando. Tudo o que você queria saber sobre o JSP quando utiliza o servidor TomCat com o Banco MySQL . Florianópolis: VisualBooks, 2002.