UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

RODRIGO CESAR AREND

IMPLEMENTAÇÃO DE APLICAÇÃO WEB PARA CONTROLE DE GASTOS DA CENTRAL TELEFÔNICA DA UTFPR

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO
2013

RODRIGO CESAR AREND

IMPLEMENTAÇÃO DE APLICAÇÃO WEB PARA CONTROLE DE GASTOS DA CENTRAL TELEFÔNICA DA UTFPR

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Diplomação, do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. Dr. Fábio Favarim

PATO BRANCO
2013

ATA Nº: **213**

DEFESA PÚBLICA DO TRABALHO DE DIPLOMAÇÃO DO ALUNO **RODRIGO CESAR AREND**.

Às 18:30 hrs do dia 30 de abril de 2013, Bloco V da UTFPR, Câmpus Pato Branco, reuniu-se a banca avaliadora composta pelos professores Fábio Favarim (Orientador), Beatriz Terezinha Borsoi (Convidada) e Lucilia Yoshie Araki (Convidada), para avaliar o Trabalho de Diplomação do aluno Rodrigo Cesar Arend, matrícula 00.0846.058, sob o título **Implementação de Aplicação Web para Controle de Gastos da Central Telefônica da UTFPR**; como requisito final para a conclusão da disciplina Trabalho de Diplomação do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, COADS. Após a apresentação o candidato foi entrevistado pela banca examinadora, e a palavra foi aberta ao público. Em seguida, a banca reuniu-se para deliberar considerando o trabalho **APROVADO**. Às 19:20 hrs foi encerrada a sessão.

Prof. Fábio Favarim, Dr.
Orientador

Profa. Beatriz Terezinha Borsoi, Dr.
Convidada

Profa. Lucília Yoshie Araki, M.Sc.
Convidada

Prof. Omero Francisco Bertol, M.Sc.
Coordenador do Trabalho de Diplomação

Prof. Edilson Pontarolo, Dr.
Coordenador do Curso

RESUMO

AREND, Rodrigo Cesar. Implementação de aplicação *Web* para controle de gastos da central telefônica da UTFPR. 2013. 70f. Trabalho de conclusão de curso – Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná. Pato Branco, 2013.

Aplicações *Web* estão se tornando cada vez mais comuns devido ao fato de serem soluções versáteis e acessíveis. Essas aplicações não necessitam de instalação em máquinas locais, apenas um servidor e conexão com a Internet são necessários para que o sistema esteja disponível. Assim, esse trabalho tem por objetivo a implementação de um sistema *Web* para atender uma demanda da UTFPR – Câmpus Pato Branco que é a informatização do controle de gastos com ligações telefônicas que são realizadas pelos servidores da universidade. O sistema permitirá ao gestor um controle mais efetivo e facilitado das ligações realizadas e os servidores possuirão um mecanismo mais ágil para informar a finalidade das ligações que cada um realiza.

Palavras-chave: Aplicação *Web*. Sistema de controle de gastos de ligações telefônicas. Análise orientada a objetos.

ABSTRACT

AREND, Rodrigo Cesar. Implement of Web Application for telephone exchange spending control in UTFPR. 2013. 70f. Trabalho de conclusão de curso – Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná. Pato Branco, 2013.

Web applications are becoming increasingly common due to the fact that they are versatile and affordable solutions. These applications do not require installation on local machines, only one server and internet connection are required for the system is available. Thus, this study aims the implementation of a web system to meet a demand of UTFPR - Câmpus Pato Branco, who is the computerization of control spending on phone calls that are performed by employees of the university. The system will allow the manager to more effectively control and facilitated the calls made and servers possess a more expeditious to inform the purpose of the calls that each performs.

Keywords: *Web* application. Object-oriented analysis. Spending control for telephone exchange system

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

AOO	Análise Orientada a Objetos
ARPA	Agência de Projetos de Pesquisas Avançadas
CERN	Organização Européia para Pesquisa Nuclear
CRUD	Inserção, leitura, atualização e exclusão
GPL	*General Public License*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
IBM	*International Business Machine*
MDA	Arquitetura Dirigida ao Modelo
MYSQL	Sistema de Gerenciamento de Banco de Dados
OMG	Grupo de Gestão da Orientação a Objetos
OO	Orientação a Objetos
PHP	*Hypertext preprocessor*
PIM	Modelo Independente de Plataforma
POO	Programação Orientada a Objetos
PSM	Modelo Específico de Plataforma
SQL	*Structured Query Language*
SSH	*Secure Shell*
UML	*Unified Modeling Language*
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1 INTRODUÇÃO

Esta introdução tem por finalidade apresentar informações iniciais deste trabalho, que são constituídas dos objetivos, da justificativa e da estruturação do mesmo.

1.1 Considerações Iniciais

A rápida expansão da Internet tem aberto novas possibilidades para a implantação de serviços computacionais que substituam atividades realizadas de maneira manual e as tornem mais amplamente acessíveis. A *Web* é, atualmente, o principal veículo para a prestação desses serviços, permitindo atingir um número cada vez maior e mais diversificado de usuários.

Uma aplicação *Web* é, tecnicamente, um sistema desenvolvido para ser interpretado em um navegador de Internet, conectando-se através de um protocolo de comunicação a um servidor HTTP (*Hypertext Transfer Protocol*). Este servidor é responsável por tratar os pedidos e respostas entre cliente e servidor.

A ideia principal do sistema desenvolvido como resultado deste trabalho é a elaboração de um aplicativo *Web* que manterá o controle de custos de todas as ligações telefônicas realizadas na central telefônica da UTFPR – Câmpus Pato Branco. Com isso cada funcionário que utilize a central da Universidade possuirá um relatório on-line de suas ligações, podendo justificá-las e saber qual será seu gasto.

1.2 Objetivos

A seguir estão os objetivos definidos para este trabalho. O objetivo geral destaca o resultado principal obtido com o desenvolvimento deste trabalho. E os objetivos específicos complementam o objetivo geral no sentido de contribuições do uso desse sistema pela própria Universidade.

1.2.1 Objetivo Geral

Desenvolver uma aplicação para controle de gastos telefônicos da Universidade Tecnológica Federal do Paraná – Câmpus Pato Branco.

1.2.2 Objetivos Específicos

- Auxiliar a UTFPR, câmpus Pato Branco, no controle e na cobrança das ligações telefônicas realizadas por parte dos funcionários e setores;
- Desenvolver um aplicativo *Web*, visando facilitar o acesso, para a justificativa das ligações telefônicas;
- Facilitar o trabalho do setor responsável pelo controle das ligações telefônica que são realizadas a trabalho e particulares.

1.3 Justificativa

O desenvolvimento de uma aplicação *Web* para realização do controle das cobranças das ligações telefônicas da Universidade Tecnológica Federal do Paraná – Câmpus Pato Branco tem como justificativa a automatização de um processo que atualmente é realizado mensalmente pela equipe responsável do setor. Esse processo vem sendo realizado de maneira manual. E além de ser muito dispendioso é passível de erros, ocasionando cobranças indevidas.

O desenvolvimento de uma aplicação para plataforma *Web* torna todo o processo do controle das cobranças das ligações telefônicas mais simples e rápido e cada usuário terá um controle mais efetivo de gastos. O usuário poderá acessar o sistema a partir de qualquer computador com acesso à Internet.

Com auxílio do sistema desenvolvido como resultado deste trabalho o controle das ligações telefônicas realizadas será mais confiável, porque o próprio usuário registra as justificativas das ligações que são a trabalho e as particulares diretamente no aplicativo computacional. Atualmente o relatório das ligações

realizadas e as justificativas para o setor responsável são enviadas por email. Os dados são enviados para o usuário no corpo do texto dificultando a identificação das informações. Com o sistema desenvolvido o acesso e entendimento dessas informações será mais fácil, além da possibilidade de registrar justificativa padrão para números de telefone, como os costumeiramente utilizados pelo servidor.

1.4 Estrutura do Trabalho

Este texto está organizado em capítulos, dos quais este é o primeiro e apresenta a ideia do sistema, incluindo os objetivos e a justificativa.

O Capítulo 2 apresenta o referencial teórico, e está centrado em *Web*, aplicações cliente-servidor e descreve brevemente sobre análise de sistemas orientados a objetos.

No Capítulo 3 estão os materiais e o método utilizados. Os materiais se referem às ferramentas, linguagens e demais recursos utilizados na implementação do sistema. O método contém as etapas necessárias para realizar o trabalho.

O Capítulo 4 apresenta o resultado obtido com a realização deste trabalho que é o desenvolvimento de um sistema para gerenciamento de gastos com ligações telefônicas.

No Capítulo 5 apresenta a conclusão. E por fim estão as referências bibliográficas utilizadas na elaboração do trabalho.

2 APLICAÇÕES WEB COM ORIENTAÇÃO A OBJETOS

Este referencial teórico tem o objetivo de apresentar a distinção entre *Web* e Internet, a lógica de funcionamento de uma aplicação *Web*, a orientação a objetos em aplicações *Web*, e um comparativo entre as aplicações *desktop* e as desenvolvidas com programação estruturada.

2.1 Diferença entre Web e Internet

Com a expansão da Internet dois termos se tornaram muito comuns nesta área de pesquisa e em alguns momentos são tratados como sinônimos: a *Web* e a Internet. Mas na verdade *Web* e Internet são duas coisas distintas.

A Internet surgiu a partir de um projeto da Agência Norte-Americana de Projetos de Pesquisa Avançada (ARPA), com o objetivo de interligar computadores dos seus departamentos de pesquisa. A primeira estrutura montada que realizava essa comunicação entre computadores foi chamada de ARPANET, que no ano de 1969 interligava quatro instituições de ensino americanas: Universidade da Califórnia, a Universidade de Santa Bárbara; Instituto de Pesquisa de Stanford e Universidade de Utah.

A Internet pode ser definida como um sistema global de comunicação. É uma infraestrutura em rede que liga inúmeros computadores em todo o mundo, formando uma rede em que qualquer computador pode comunicar-se com qualquer outro computador desde que ambos estejam ligados à Internet. A informação que trafega pela Internet é feita por meio de uma variedade de linguagens conhecidas por protocolos. Para Kenneth e Jane Laudon “a Internet é atualmente uma rede global, de total integração de centenas de milhares de outras redes locais, regionais ou nacionais” (LAUDON, K.; LAUDON, J., 1998, p. 168).

A Internet é organizada na forma de uma teia. Para conectar um computador a outro as mensagens de comunicação passam por inúmeros computadores. O primeiro computador informa que deseja se conectar com a outra máquina, através de protocolos essa mensagem passa por vários servidores, que informam o próximo caminho a ser seguido até chegar à máquina destino. Carlos J. Costa define a Internet como:

> A Internet é uma rede constituída por várias redes que se encontram interligadas, criando uma comunidade virtual de utilizadores de grande dimensão. Uma das características da Internet é o fato de não possuir qualquer autoridade a controlar. (COSTA, 2007, p. 5)

No ano de 1980, Timothy Berners-Lee, físico e pesquisador do Instituto de Tecnologia de Massachusetts e da Organização Européia para a Investigação Nuclear (CERN) na Suíça, criou um sistema que armazenava e disponibilizava informações interconectadas e compartilhadas com os demais pesquisadores. Esse sistema ficou conhecido como ENQUIRE, e é considerado o precursor da *Web*.

Em 1989, Timothy Berners-Lee juntamente com Robert Cailliau, percebeu o grande desenvolvimento do sistema ENQUIRE, e elaborou a primeira definição formal em que se referia à *Web* como uma plataforma multimídia e interativa que tornou a Internet popular fora dos meios acadêmicos e de pesquisa.

Atualmente *Web* pode ser definida como um serviço que é aplicado sobre a Internet responsável pela difusão das informações interconectadas por *links* definidos como hipertextos. Para Carlos J. Costa a *Web* pode ser definida como:

> WWW é um sistema de informação cliente-servidor em hipertexto distribuído na Internet. Na Web, tudo (documentos, menus e índices) são objetos em hipertexto no formato HTML. O programa cliente é chamado de browser, sendo executado no programa do utilizador operações de navegação. (COSTA, 2007, p. 5)

A *Web* é um modelo de compartilhamento de informações que utiliza o protocolo HTTP (*Hipertext Transfer Protocol*) para transmitir informações, e que se serve de navegadores, como o Internet Explorer ou Mozilla Firefox, para acessar uma série de documentos, que contêm gráficos, sons, textos e vídeos, chamados páginas, e que possuem uma relação de interligação por meio de *hyperlinks*. Portanto, a *Web* pode ser definida como uma grande parte da Internet.

2.2 Aplicações Web

Uma aplicação *Web* é um programa implementado para ser executado por meio de um navegador, na Internet ou em redes privadas (Intranet). Esta aplicação tem a sua execução em um servidor com suporte a HTTP. Este modelo de aplicação

se caracteriza pela facilidade em ser atualizado e mantido, pois seus códigos e bancos de dados ficam localizados em apenas um servidor.

Para Aniceto (2009, p.2), aplicação *Web* pode ser definida como:

> sistema de informática projetado para a utilização através de um navegador, na internet ou em redes privadas, e o seu desenvolvimento tem muito haver com a necessidade de simplificar a utilização e manutenção, mantendo o código fonte em um mesmo local, de onde é acessado pelos diferentes usuários.

Muitas pessoas consideram que a *Web* serve apenas para a criação de páginas simples, com HTML básico, apresentando imagens, documentos, arquivos de maneira simples, se importando principalmente com a estética e em alguns casos ignorando a usabilidade. Entretanto uma aplicação *Web* pode ser tão ou mais usual e eficiente que uma aplicação *Desktop*. Para isso um projeto deve seguir alguns passos semelhantes aos das aplicações Desktop: planejamento, definição de arquitetura e padrões, teste, medida de desempenho, atualizações e manutenção.

Aplicações *WEB* também são conhecidas por serem aplicações que são processadas no modelo cliente/servidor. Isto é, um cliente acessa determinada página, com isso uma requisição é enviada a um servidor, que processará todas as regras e a estrutura da página requisitada e enviará ao cliente, com isso o cliente recebe apenas o resultado da requisição.

2.2.1 Categorias das Aplicações Web

Devido a grande diversidade de aplicações *Web* que existem atualmente, alguns autores acabam dividindo-as em grupos. Suh (2004) apresenta uma divisão baseando-se principalmente na funcionalidade que o sistema possui. Este modelo divide todas as aplicações em seis grupos distintos, porém existem aplicações que se enquadram em mais de uma categoria.

As categorias das aplicações *Web* são:

- Informacional: aplicações voltadas para apenas fornecer informações para os usuários.
- Interativa: também voltada a apresentar informações, porém possui pequena interação com os usuários por meio de formulários.

- Transacional: são aplicações que possuem maior interação com o usuário, sendo que esses além de fornecer informações podem alterar o conteúdo das aplicações.
- Orientadas a fluxo de trabalho: aplicações que auxiliam os trabalhos internos de uma empresa ou as movimentações entre empresas diferentes. Também auxiliam no planejamento e funcionamento da empresa.
- Ambiente de trabalho corporativo: aplicativos que realizam a cooperação de objetos em operações não estruturadas de empresa, auxiliando na comunicação entre usuários.
- Comunidades online: também são aplicações que realizam a comunicação entre os usuários, porém de maneira informal.

O Quadro 1 ilustra exemplos de aplicações de cada uma das categorias.

Categorias	**Exemplos**
Informacional	Jornais *on-line*, catalogo de produtos, revistas, classificados *on-line.*
Interativo	Formulários de registro, apresentação de informações personalizadas, jogos *on-line.*
Transacional	Shoppings *on-line* (encomendas de bens e serviços), banco on-line, reserva de passagens aéreas *on-line*, pagamento de contas *on-line.*
Fluxo de trabalho orientado	Planejamento e agendamento *on-line*, gestão de estoque, monitoramento de status, gestão de abastecimento de energia.
Ambientes colaborativo de trabalho	Sistema de autorização distribuída, ferramentas de design colaborativo.
Comunidades *on-line*, comercio eletrônico	Grupos de discussão, comercio eletrônico, sistemas de recomendação.

Quadro 1 – Categorias das Aplicações Web
Fonte: Traduzido de SUH (2004, p.5)

2.3 Arquitetura Cliente-Servidor

A arquitetura cliente-servidor é um modelo computacional que separa o processamento das informações entre o lado cliente (usuário do sistema) e servidor (unidade de processamento das informações), sendo estas duas estruturas interligadas por meio de rede de computadores.

Para Kenneth e Jane Laudon, “o servidor pode armazenar programas aplicativos e arquivos de dados e pode distribuir programas ou arquivos de dados

para outros computadores da rede à medida que estes os solicitam (LAUDON, LAUDON, 1998, p. 154).

As instâncias clientes realizam requisições de dado para o servidor da aplicação que está sendo executada e aguardam o retorno da resposta. O servidor disponível para a aplicação pode aceitar as requisições, processá-las e retornar o resultado para o cliente.

A estrutura atual da arquitetura cliente-servidor pode ser dividida em quatro tipos básicos, conforme ilustra a Figura 1. Esses tipos foram definidos por Ferrante e Rodriguez (2000, p. 294).

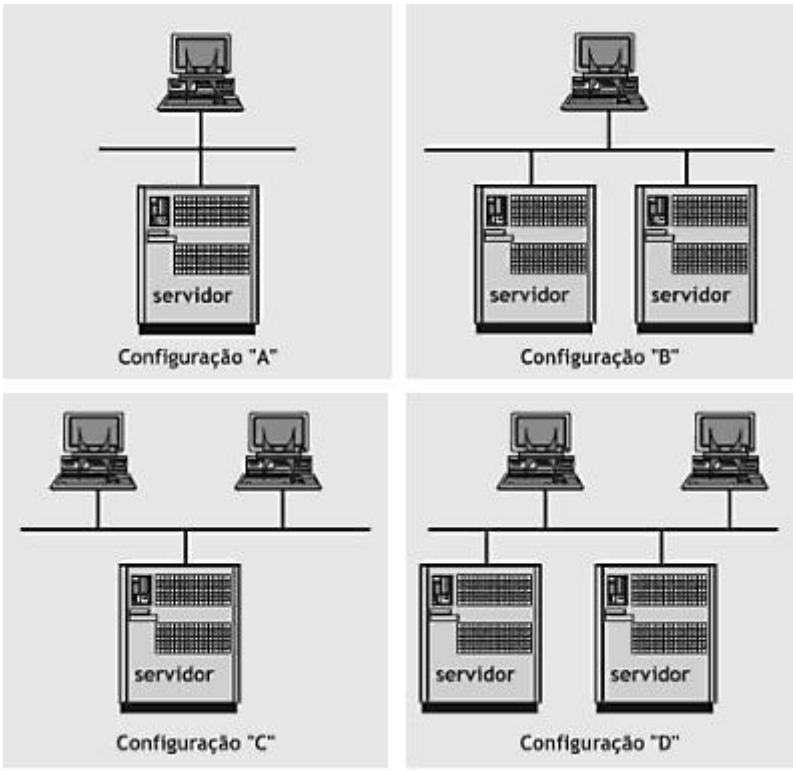

Figura 1 – Topologia da arquitetura cliente-servidor
Fonte: Ferrante e Rodriguez (2000, p. 294)

A configuração ‘A’ é conhecida como um servidor para um cliente. É uma tipologia de uso comum, em que dados e sistema estão localizados em apenas um servidor e fornecem acesso a apenas uma máquina cliente.

A configuração ‘B’ é conhecida como dois ou mais servidores para um cliente, isso significa que os dados, as funções e operações de processamento ficam divididas em mais de um servidor, sendo que cada um tem a sua função específica. O cliente deve acessar o servidor adequado para a função que deseja resposta.

A configuração ‘C’ é definida como muitos clientes para um servidor. É a estrutura mais utilizada atualmente e pode ser considerada análoga às estruturas de *mainframes*. Apenas um servidor é responsável por responder todas as requisições de todos os clientes.

A configuração ‘D’ é definida como muitos clientes para muitos servidores. Representa a verdadeira computação distribuída, em que cada função é desempenhada por um servidor específico e os inúmeros clientes devem enviar a requisição para o servidor adequado.

Algumas vantagens e desvantagens da estrutura cliente-servidor são apresentadas por Ferrante e Rodriguez (2000):

Vantagens:

- o processamento é executado bem próximo ao usuário;
- a capacidade de processamento pode ser ajustada localmente, de forma modular;
- o processamento de imagens, sons e vídeos pode ser disponibilizado localmente de forma bem mais fácil do que no processamento centralizado.

Desvantagens:

- é necessária uma adequada distribuição do processamento entre o cliente e o servidor. De outra forma, a performance de todo o sistema pode se degradar;
- um alto tráfego de dados na rede poderá afetar negativamente a performance do sistema;
- a segurança dos dados e o controle de versão do software serão mais difíceis de se aplicar, por causa da distribuição dos sistemas pela rede.

Atualmente, o processamento que se utiliza de processamento cliente-servidor tornou-se atraente por parecer menos dispendioso embora mais flexível que usar um *mainframe* central. Os custos para a elaboração de uma rede com estrutura cliente-servidor acabam sendo bem mais baixos que a estrutura necessária para a utilização de um servidor central ou *mainframe*. A maior dificuldade na utilização do processamento cliente-servidor, é que as aplicações utilizadas têm que estar preparadas para esta estrutura e que as redes clientes-servidores ainda não são tão confiáveis.

2.4 Análise Orientada a Objetos

A Análise Orientada a Objetos (AOO) é uma técnica de análise de sistemas derivada do projeto orientado a objetos e da programação orientada a objetos. Basicamente a Análise Orientada a Objetos tem o objetivo de diminuir a distância entre o modelo conceitual e o modelo real, para isso identificando e definindo um conjunto de objetos que interagem e comportam-se conforme os requisitos do sistema.

Segundo Michael Blaha e James Rumbaugh a AOO pode ser considerada como “uma nova maneira de pensar os problemas utilizando conceitos do Mundo Real. O componente fundamental é o objeto que combina estrutura e comportamentos em uma única entidade.” (RUMBAUGH , BLAHA, 2006)

Esta proximidade entre o mundo real e o modelo computacional é a vantagem mais relevante da Análise Orientada a Objetos e com isso outros benefícios podem ser obtidos. Para Correia (2006) outros benefícios importantes são:

- A manutenção da modelagem do sistema e a automação do mesmo o mais próximo possível de uma visão conceitual do mundo real.
- Baseia a decomposição e a modelagem do sistema nos dados, que é o elemento mais estável de todos aqueles que compõem um sistema de informação.
- Das abordagens de análise a orientada a objetos é a que oferece maior transparência na passagem da análise (modelo essencial) para o projeto (modelo de implementação). Isso se deve ao fato de que se for seguida a técnica de projeto orientado a objeto a passagem de um modelo para o outro será feita através da introdução de detalhes, não requerendo uma reorganização do modelo, como é o caso na passagem de análise estruturada para o projeto estruturado.

2.4.1 Objeto

Na orientação a objetos um objeto pode ser definido como um agente responsável por realizar uma tarefa específica, e através da interação e troca de mensagens entre objetos distintos, determinada tarefa é realizada.

Em uma analogia entre o mundo real e o mundo computacional um objeto pode ser dito como uma “coisa, real ou abstrata, a respeito da qual armazenamos dados e os métodos que os manipulam”.

Todo objeto possui características próprias que os descrevem. Essas características são chamadas de atributos. Todos os objetos devem possuir esses atributos que podem ser diferentes entre um objeto e outro. Alguns exemplos de atributos podem ser vistos na Figura 2.

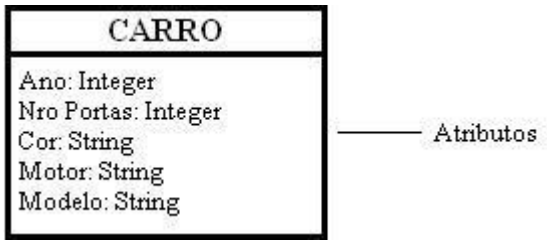

Figura 2 – Exemplo de atributos de um objeto
Fonte: Autoria própria.

Os métodos de um objeto são os procedimentos ou funções que manipulam os dados contidos no objeto para a realização de determinada tarefa. Esses métodos podem ser exclusivos do objeto ou interagir com outros objetos para a realização da tarefa. A Figura 3 apresenta alguns exemplos de métodos.

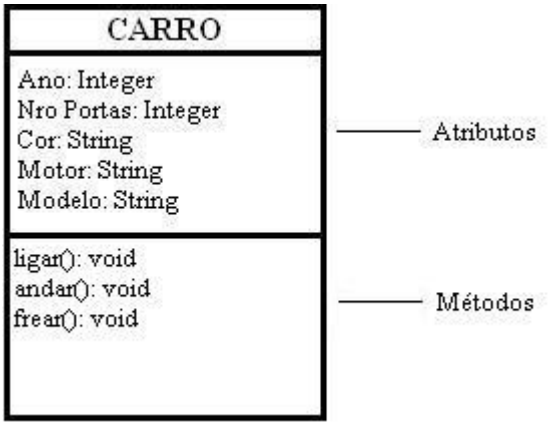

Figura 3 – Exemplo de métodos de um objeto
Fonte: Autoria própria.

2.4.2 Classes

Classe é uma estrutura responsável pela abstração de um conjunto de objetos com características semelhantes. Uma classe define o comportamento de seus objetos através de métodos e os estados possíveis destes objetos através de

atributos. Em outros termos, uma classe descreve os serviços providos por seus objetos e quais informações eles podem armazenar.

Uma classe define o estado e o comportamento de um objeto geralmente implementando métodos e atributos. Para Eduardo Bezerra, “uma classe é uma descrição dos atributos e serviços comuns a um grupo de objetos” (BEZERRA, 2003). Os atributos, também chamados de campos, indicam as possíveis informações armazenadas por um objeto de uma classe, representando o estado de cada objeto. Os métodos são procedimentos que formam os comportamentos e serviços oferecidos por objetos de uma classe.

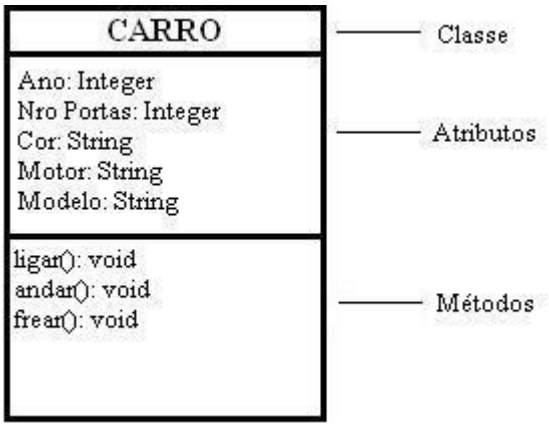

Figura 4 – Exemplo de classe
Fonte: Autoria própria.

2.4.3 Encapsulamento

Na orientação de objetos, encapsulamento é a propriedade que permite o empacotamento de atributos e métodos em uma mesma classe. Esta propriedade protege os dados contra acesso indevido, pois somente métodos da própria classe poderão alterar as estruturas de dados desta classe.

Para Serson (2009, p. 302), o “Encapsulamento é o mecanismo a partir do qual os detalhes da implantação dos métodos de uma classe são ocultados dos usuários da classe.”

2.4.4 Herança

Herança é um mecanismo que permite que características comuns a diversas classes, sejam decompostas em uma classe base, ou superclasse. A partir de uma classe base, outras classes podem ser especificadas. Cada classe derivada ou

subclasse apresenta as características (estrutura e métodos) da classe base e acrescenta a elas o que for definido de particularidade para a mesma.

Essa propriedade da orientação a objetos pode ser utilizada na intenção de reaproveitar código, comportamento generalizado, especializar operações ou atributos. Segundo Serson (2009, p. 298), “as duas razões para usar herança são: promover o reuso de código e proporcionar o polimorfismo”.

A herança pode ser considerada a principal característica da orientação a objetos, pois, se utilizada de maneira adequada apresenta diversas vantagens como reaproveitamento de código, velocidade na programação, menor número de linhas de códigos, maior organização e tornam o uso de conceitos da orientação a objetos muito mais efetivo.

2.5 Programação Orientada a Objetos

A Programação Orientada ao Objeto (POO) é considerada uma evolução natural da Programação Estruturada, porém esta mudança está acontecendo de maneira progressiva. Em geral a POO utilizou-se das melhores ideias da programação estruturada e adicionou a ela novos conceitos que fossem capazes de subdividir as tarefas a serem realizadas e as tornassem mais simples.

A primeira linguagem a apresentar conceitos de orientação a objetos foi a Simula (*Simula Language*), concebida na Noruega no ano de 1966, sendo utilizada para a programação de simulações de sistemas que possibilitem a criação de modelados pela interação de um grande número de objetos distintos (LEITE, 2006).

Mas a primeira linguagem a tornar a programação orientada a objetos mais conhecida, foi Smalltalk, desenvolvida no Centro de Pesquisas da Xerox durante a década de 70. Sendo uma linguagem OO que implementa todos os conceitos de OO. O que não acontece com as chamadas linguagens OO híbridas que implementam apenas alguns conceitos de orientação ao objeto, como por exemplo o C++ e Java.

Basicamente, o princípio da POO assemelha-se com a da engenharia de *hardware* que projeta novos equipamentos, usando os mesmos componentes básicos, como transistores, resistores e outros. Os objetos básicos já existentes são utilizados para produzir novos objetos.

Seguindo a ideia de Martius Rodrigues e Agustín Juan Ferrante, “a Programação Orientada a Objetos elimina a divisão clássica entre programação de aplicação e programação de estrutura de dados, já que as funções são agora atributos dos objetos” (FERRANTE; RODRIGUES, 1999, p. 290). Estes objetos são responsáveis pela principal vantagem que a Programação Orientada a Objetos tem sobre a Programação Estruturada que é a possibilidade de reutilização de código e modularidade de escrita. Com isto determinado código é escrito apenas uma vez e estará acessível sempre que o objeto for invocado, além deste código estar em apenas um arquivo sem que se repita por mais de um local.

Em uma implementação de códigos orientados a objetos, alguns conceitos devem estar presentes: abstração, encapsulamento, herança e polimorfismo.

2.5.1 Abstração

O programador deve se concentrar nos aspectos mais relevantes de um contexto geral, deixando aspectos menos importantes em segundo plano. Uma classe que determina as características de um objeto deve ser é uma abstração de entidades existentes no sistema, contendo tudo o que for determinante para a existência e manutenção deste objeto.

2.5.2 Encapsulamento

A ideia central da POO consiste em separar os aspectos externos de um objeto, os quais são acessíveis a outros objetos, dos detalhes internos de implementação do objeto, os quais permanecem escondidos dos outros objetos. Desta maneira, pode-se dizer que um dado está encapsulado quando envolvido por código, de forma que só é visível na rotina onde foi criado; o mesmo acontece com uma rotina encapsulada em que suas operações internas são invisíveis às outras rotinas. Outra vantagem do encapsulamento, é que ele possibilita que os códigos evoluam sem interferir em instâncias do objeto. Se determinada rotina sofrer melhorias, todos os objetos criados por essa classe terão acesso às melhorias, sem a necessidade de alterações. Além disso, com o encapsulamento pode-se definir permissões de acesso para os atributos e métodos de um objeto, permitindo o acesso direto ou não. Essas permissões são:

- Pública: os atributos ou métodos podem ser acessados diretamente por subclasses ou por instâncias do objeto.

- Privado: define que os atributos ou métodos só poderão ser acessados apenas pela própria classe.

- Protegido: subclasses poderão acessar os atributos e métodos, mas outras classes não possuem esta permissão.

2.5.3 Herança

Permite que atributos comuns a diversas classes sejam divididos em uma classe base, ou superclasse. A partir desta classe base, outras classes, conhecidas como classes derivadas, podem ser especificadas, definindo características (estrutura e métodos) da classe base e acrescentando a elas o que for definido de particularidade. Com isso, as novas classes herdam todos os membros (propriedades e métodos) da classe-mãe; isto torna o mecanismo de herança uma técnica muito eficiente para construir, organizar e reutilizar código. Em linguagens que não suportam herança, cada classe é uma unidade independente, sendo todos os seus membros concebidos do zero (sem vínculo direto com outras classes), resultando em processos mais lentos e em redundância de código.

2.5.4 Polimorfismo

Etimologicamente significa “várias formas" e é definido como o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação, mas comportamentos distintos, especializados para cada classe derivada, usando para isto uma referência a um objeto do tipo da superclasse.

A idéia de Polimorfismo está estritamente relacionada com a existência de interfaces (característica de uma Linguagem OO). A partir destas interfaces é possível classificar grupos de objetos que têm comportamentos em comum, entretanto a sua implementação é diferente. Para executar a mesma ação de um determinado objeto, qualquer que seja o seu tipo, podem-se invocar funções diferentes, as quais possuem a mesma assinatura e o mesmo tipo de retorno, porém realizam essas funções de maneira distinta. O Quadro 2 apresenta um exemplo de

polimorfismo. Como pode ser visualizado o interface iFigura obrigatoriamente deve possuir uma função do tipo pública nominada de getArea. As classes Quadrado e Retângulo implementam a interface iFigura, sendo que obrigatoriamente devem possuir a função getArea, porem a implementação dessas funções são feitas de maneiras distintas.

```
<?php
/**  Interface responsável por definir a assinatura do método */
interface IFigura {
            /** Assinatura do método */
            public function getArea();
}
?>
<?php
/** Classe Quadrado */
class Quadrado implements IFigura {
        /** Definição de um atributo privado */
        private $lado;
        /** Construtor da classe */
        function Quadrado( $intValue ) {
             $this->lado = (double) $intValue;
         }

        /** Área do Quadrado */
         public function getArea() {
              return $this->lado * $this->lado;
          }
}
?>
<?php
/**  Classe Retângulo */
class Retangulo implements IFigura {
    private $base;
    private $alt;
    function Retangulo( $intBase,  $intAlt){
           $this->base = (double) $intBase;
           $this->alt = (double) $intAlt;
    }

    /** Função getArea Polimórfica  */
    public function getArea(){
            return ($this->base * $this->alt);
    }
}
?>
```

Quadro 2 - Codificação exemplo de polimorfismo
Fonte: Alexandrino (2013).

3 MATERIAIS E MÉTODO

Este capítulo de materiais e método relaciona todas as ferramentas, tecnologias e a metodologia utilizada na elaboração deste trabalho.

3.1 Materiais

Na realização deste trabalho foram utilizadas as seguintes tecnologias e ferramentas.

- DBDesigner 4: modelagem do sistema;
- StarUML: modelagem do sistema;
- MySQL: Sistema de gerenciamento de banco de dados que utiliza linguagem SQL;
- MySQL-Front: ferramenta para administrar e manipular o banco de dados MySQL;
- PHP: linguagem de programação;
- Adobe Dreamweaver: IDE para criação de páginas HTML, com interface visual e para manipulação de códigos PHP;
- Easy-PHP: servidor *Web* e da Linguagem PHP.

3.1.1 DBDesigner 4

O DBDesigner (FABFORCE, 2013) é um software de código aberto, desenvolvido pela empresa Fabulous Force Database, que possibilita a elaboração de modelos lógicos de banco de dados, com todos os seus componentes e relacionamentos. Além desta funcionalidade, a ferramenta permite a criação de *scripts* automatizados para SQL (*Structured Query Language*).

Esta ferramenta foi desenvolvida voltada para usuários de banco de dados MySQL, pois adota a mesma metodologia deste sistema de gerenciamento de banco de dados. Outra vantagem de sua utilização combinado com o MySQL é a possibilidade de conexão entre o DBDesigner e o MySQL, sendo possível automatizar os processos, pois todas as alterações realizadas no modelo podem ser exportadas automaticamente para o banco de dados. Outra vantagem desta

ferramenta quando utilizada juntamente com MySQL, é a possibilidade de Engenharia Reversa, sendo com isso possível a criação de modelo completo a partir de um banco de dados pré-existente.

O programa possui dois modos de interface. A primeira é a interface de *Design Mode*, mais simples e intuitiva, sendo esta o modo de trabalho padrão do programa. Neste modo a criação dos modelos de banco de dados é feita basicamente arrastando componentes para a área de modelagem. A segunda interface é a *Query Mode*, que possibilita a criação de SQL para alteração de tabelas e manipulação de dados.

A Figura 5 apresenta a tela do DBDesigner 4, em modo *Design* com um modelo de banco de dados.

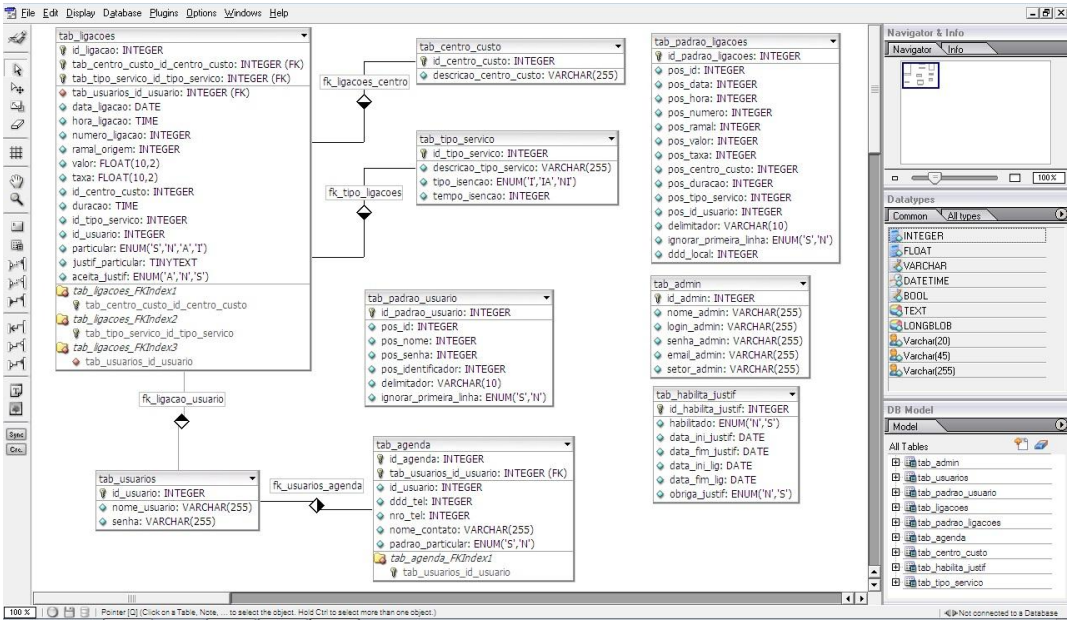

Figura 5 – Tela do software DBDesigner 4 com interface design
Fonte: Autoria própria.

3.1.2 StarUML

StarUML (StarUML, 2013) é um software que adota a política de código aberto e distribuído de maneira gratuita com o objetivo de propiciar o desenvolvimento de diagramas no padrão UML de maneira simplificada e flexível, sendo disponível na plataforma Microsoft Windows. StarUML é uma ferramenta de modelagem de software, com todos os diagramas e componentes da UML, sendo capaz de substituir ferramentas comerciais de UML.

O software possui as seguintes características:

- **Padrão UML 2.0:** Devido aos trabalhos realizados pelo OMG (Object Management Group) a UML está continuamente se expandindo. O padrão mais recente e mais utilizado é a versão 2.0, sendo este o utilizado pelo software.
- **MDA (Model Driven Architecture):** MDA é um método de desenvolvimento de software elaborado pela OMG que torna a modelagem o centro de todo o processo. Todos os modelos devem ser criados a partir de 3 (três) etapas;
    - A primeira etapa é a construção de um modelo com um alto nível de abstração, independente de qualquer tecnologia. Esse modelo é chamado de Modelo Independente de Plataforma (PIM).
    - A segunda etapa é a transformação do PIM em um ou mais Modelos Específicos de Plataforma (PSM). Um PSM é mais específico para o sistema em termos de tecnologia de implementação.
    - A terceira etapa é transformar um PSM em código propriamente dito.

  Com a implementação desses três passos, o software se torna personalizável e independente de linguagem. O programa StarUML suporta MDA e fornece variáveis de personalização e códigos no padrão MDA.
- **Arquitetura em Plug-ins:** O programa StarUml adota a utilização de *plug-ins*, isso o torna expansível e moldável a cada necessidade individual. Em um repositório *on-line* várias novas funcionalidades estão disponíveis ou o próprio usuário pode desenvolver em módulos compatíveis com C++, Delphi, VB ou C#.
- **Usabilidade:** A usabilidade é uma questão muito relevante no programa StarUML. Sua interface segue o padrão da maioria dos programas de modelagem. À esquerda existe uma caixa de ferramentas com todos os elementos que podem ser arrastados para a área central, ou área de designer. Dentre a esses componentes estão: objetos, conectores, classes, tipos de associações. À direita está o menu de edição dos componentes. Selecionando um componente é possível editar todas as suas características.

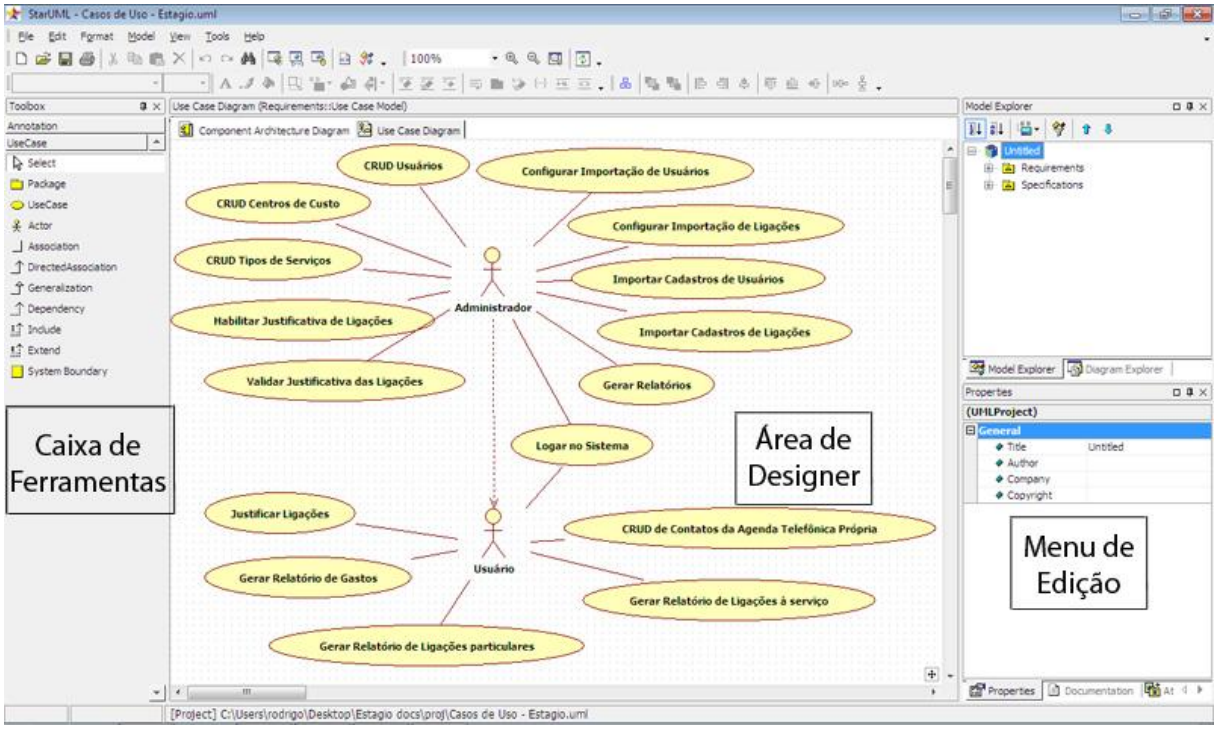

Figura 6 – Tela do software StarUML
Fonte: Autoria própria.

3.1.3 MySQL

O MySQL é um sistema de gerenciamento de banco de dados, que utiliza a linguagem SQL (Linguagem de Consulta Estruturada) como interface, sendo atualmente o banco de dados mais utilizado no mundo (Mysql, 2013).

O sistema de gerenciamento de banco de dados foi criado pelos suecos David Axmark e Allan Larsson e pelo finlandês Michael Widenius no início da década de 80.

Entre as principais características estão:

- Portabilidade: pode ser utilizado em quase todas as plataformas da atualidade.

- Compatibilidade: grande número de *drivers* ODBC, JDBC e .NET e módulos de interface para diversas linguagens de programação, como Delphi, Java, C/C++, C#, Visual Basic, Python, Perl, PHP, ASP e Ruby.

- Desempenho: motor de pesquisa avançado o que permite alta capacidade de desempenho com baixa exigência de hardware.

- Software livre: é um sistema de gerenciamento de banco de dados livre e de código aberto seguindo a licença GPL (*General Public License*)

- Suporte a controle transacional, *triggers,* procedimentos e funções.

3.1.4 MySQL-Front

O MySQL-Front é uma ferramenta para auxiliar administradores de bancos de dados que lidam com MySQL (Mysql-Front, 2013). Facilita a execução de diversas tarefas relacionadas a manipulação do próprio banco de dados, com interface gráfica completa, evitando a necessidade de elaboração de códigos SQL complexos. Possui também uma interface de codificação de comandos SQL. A interface do programa pode ser visualizada na figura 7.

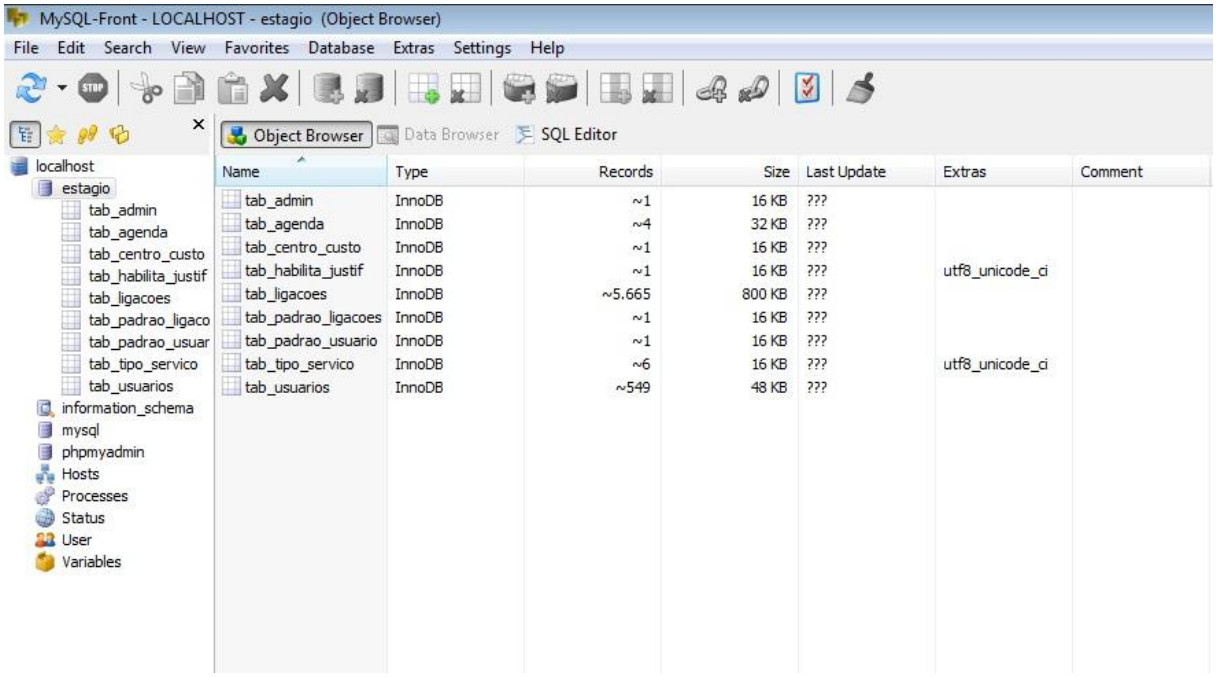

Figura 7 – Tela do software Mysql-Front
Fonte: Autoria própria.

3.1.5 PHP

PHP (H*ypertext Preprocessor*) é uma linguagem de inserção HTML, isto é, os códigos PHP são inseridos e executados no meio de uma pagina HTML normal. Criada por Rasmus Lerdorf no ano de 1995 e atualmente é mantida por um grupo chamado de *The PHP Group* sendo um software livre que possui licença própria. É uma linguagem interpretada, pois necessita de um programa interpretador para sua execução. Possui estrutura cliente-servidor, sendo que todos os códigos PHP são

interpretados no servidor e apenas a pagina resultante é encaminhada para o cliente. A figura 8 apresenta um trecho de código PHP em meio a uma pagina HTML.

```
</div>
<?php
    if($_GET['erro'] == ''){
        $disp = 'none';
    }else{
        $disp = '';
    }
?>
<div id="divErro" style="display:<?php echo $disp; ?>"><b>Usuario ou senha inválido.</b></div>
```

Figura 8 – Codificação PHP
Fonte: Autoria própria.

A sintaxe da linguagem PHP assemelha-se com a do **C** e do **Perl** e é uma linguagem com baixo grau de aprendizado, sendo atualmente a linguagem mais utilizada na internet. (Castela, 2013)

3.1.6 Adobe Dreamweaver

Dreamweaver é um software desenvolvido pela Adobe Systems desenvolvido para auxiliar o desenvolvimento de sistemas Web (Adobe, 2013). Inicialmente o programa era desenvolvido pela Macromedia, porem possui inúmeras limitações, não passando de um simples editor de códigos PHP. Atualmente a ferramenta esta muito mais desenvolvida, possuindo a opção de codificação e um modo de designe, que permite a criação de paginas HTML de maneira gráfica, apenas selecionando e arrastando componentes. Esse modo pode ser visualizado na figura 9.

A figura 10 apresenta o modo de codificação, o qual possui uma ferramenta que auxilia na inserção de funções nativas do PHP, alem de exibir as codificações com coloração distinta, facilitando a leitura do código por parte do programador.

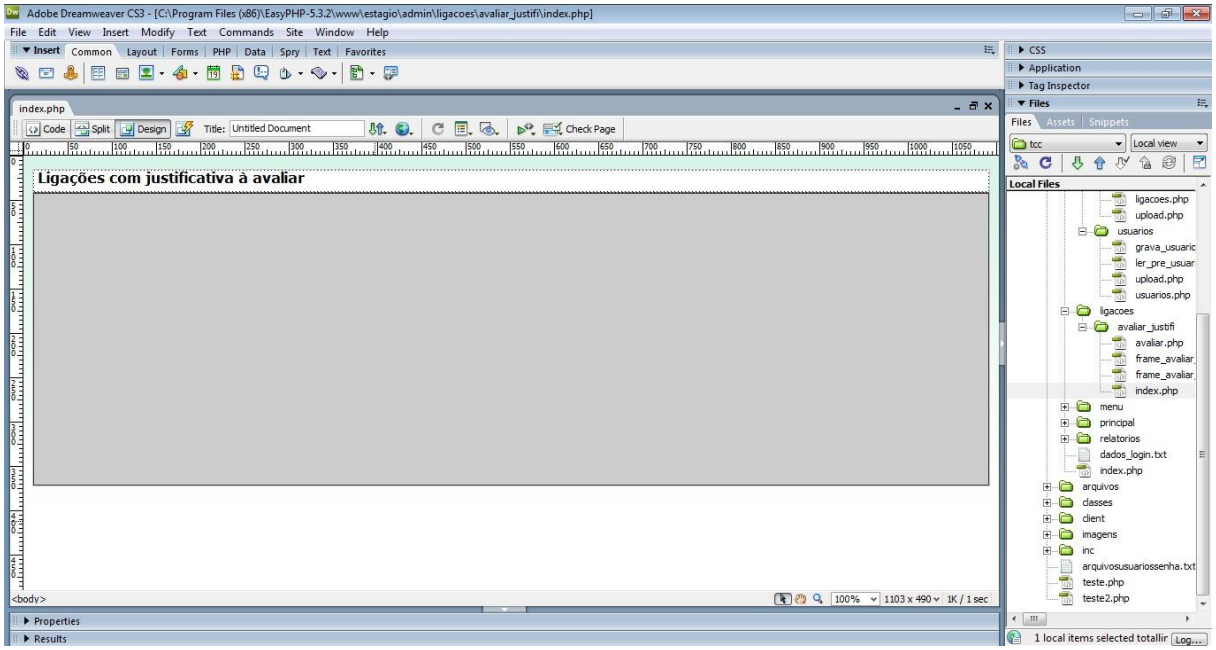

Figura 9 – Dreamweaver em modo design
Fonte: Autoria própria.

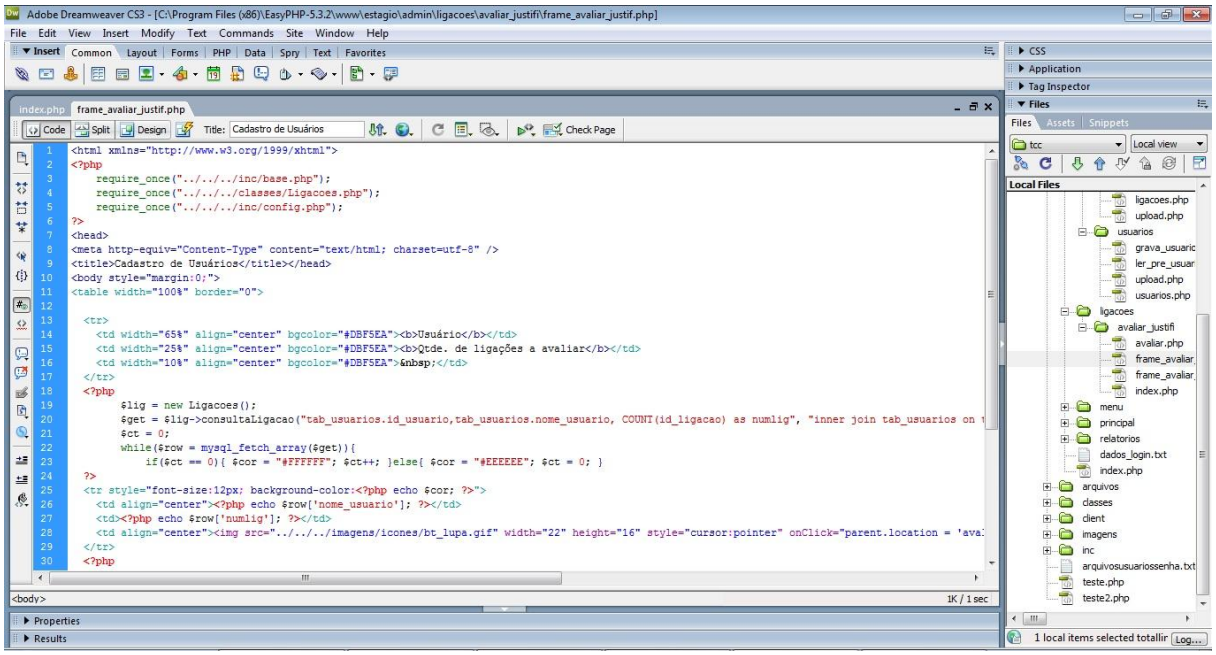

Figura 10 – Dreamweaver no modo de codificação
Fonte: Autoria própria.

3.1.7 Easy-PHP

Easy-PHP é um pacote de softwares desenvolvido para a plataforma Windows que instala automaticamente o interpretador de códigos PHP, o servidor Apache que funciona como servidor Web e o banco de dados MySql (Easy-PHP,

2013). Por padrão o programa instala o servidor Apache na porta 8080 e servidor Mysql na porta 3306, porem a partir dos arquivos de configuração é possível configurar os servidores de inúmeras maneiras diferentes.

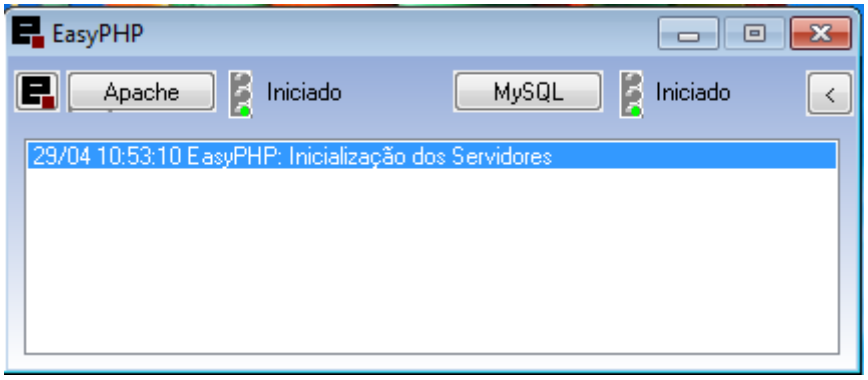

Figura 11 – Easy-php
Fonte: Autoria própria.

3.2 Métodos

Este trabalho de conclusão de curso é uma sequência do estágio realizado pelo autor deste trabalho. A análise e o projeto do sistema foram realizados como estágio e consistiram das etapas a) a d) apresentadas a seguir. As etapas realizadas neste trabalhos foram as descritas nos itens e) a g) apresentadoss a seguir:

a) Planejamento do sistema

Esta primeira etapa foi realizada por meio de uma entrevista com a funcionária responsável pelo controle da central telefônica da UTFPR – Câmpus Pato Branco, para levantar quais as necessidades que o sistema deveria atender. Inicialmente ficou definido que para o relatório de estágio seria realizado apenas a análise e projeto do sistema proposto, e que sua implementação e implantação seria realizada posteriormente.

b) Levantamento de requisitos

A partir da conversa inicial foi possível definir os requisitos que o sistema deve possuir, isto é, as condições e capacidades que o sistema deverá realizar para atender todas as demandas e necessidades dos usuários e em qual sequência as funcionalidades devem ser realizadas.

c) Tecnologias e ferramentas

Nesta etapa foi realizado um estudo para definição de qual tecnologia se enquadraria melhor para a resolução do problema apresentado. A partir disso foi definido quais ferramentas seriam utilizadas para a realização da análise do sistema e para, posteriormente, o desenvolvimento do mesmo. Inicialmente ficou definido que seriam utilizadas apenas ferramentas gratuitas.

d) Modelagem do sistema

Na etapa de análise do sistema foi realizada a detecção dos requisitos necessários para o sistema e, a partir disso, foi elaborada toda a documentação necessária para a análise completa do sistema, como Diagrama de Entidades e Relacionamentos, Diagrama de Casos de Uso e Diagrama de Classes.

e) Implementação

Codificação do sistema seguindo a modelagem que foi realizada no estágio.

f) Testes

Paralelamente ao desenvolvimento, foram feitos testes informais com o objetivo de garantir que o sistema estava funcional, atendendo as regras de negócios levantadas nos requisitos do sistema. Ao final do desenvolvimento do software o orientador do trabalho testou as funcionalidades e as correções foram realizadas pelo aluno.

g) Implantação

O sistema foi implantando em um servidor virtual alocado na UTFPR – Câmpus Pato Branco. O servidor virtual foi disponibilizado pela Coordenadoria de Tecnologia de Informação do câmpus.

4 RESULTADOS

Este capítulo apresenta o resultado da realização deste trabalho que é a implementação do sistema de controle de gastos da central telefônica da UTFPR. Inicialmente é apresentada a descrição do sistema como um todo e das suas funcionalidades essenciais. Na sequência são apresentados os requisitos levantados para o sistema e a partir destes são apresentados a documentação da análise e projeto do sistema, incluindo diagramas de casos de uso e a sua descrição detalhada, as regras de negócio, o diagrama de entidade e relacionamento, o diagrama de classes, salientando que essa documentação foi gerada durante o estágio do aluno. Por fim o resultado principal deste trabalho, que foi a implementação, testes e implantação do sistema são apresentados.

4.1 Descrição do Sistema

O sistema a ser analisado neste trabalho tem como finalidade o gerenciamento dos gastos com ligações telefônicas realizadas na Universidade Tecnológica Federal do Paraná – Câmpus Pato Branco. Atualmente este controle é realizado de maneira manual, isto é, cada colaborador da universidade possui um usuário e senha para poder realizar ligações externas. Mensalmente a funcionária responsável pela central telefônica e administradora do sistema, imprime um relatório com as ligações externas de cada usuário. Posteriormente, os usuários devem identificar manualmente quais ligações foram particulares e quais foram a trabalho, sendo que essas últimas devem ser justificadas.

Com a implantação do sistema, todas as ligações podem ser importadas para um banco de dados local. Essa importação é personalizável, devendo o responsável pela central telefônica definir qual é o caractere delimitador dos campos e em que posição cada campo estará na linha. Isso torna o sistema independente de modelo de central telefônica, sendo o único requisito a capacidade da central telefônica de exportar as ligações e os usuários em um arquivo com extensão .txt.

O módulo de importação exige alguns campos para o funcionamento do sistema. São eles: data, hora, ramal de origem, número de destino, valor, duração da ligação, tipo de ligação, centro de custo e código de usuário.

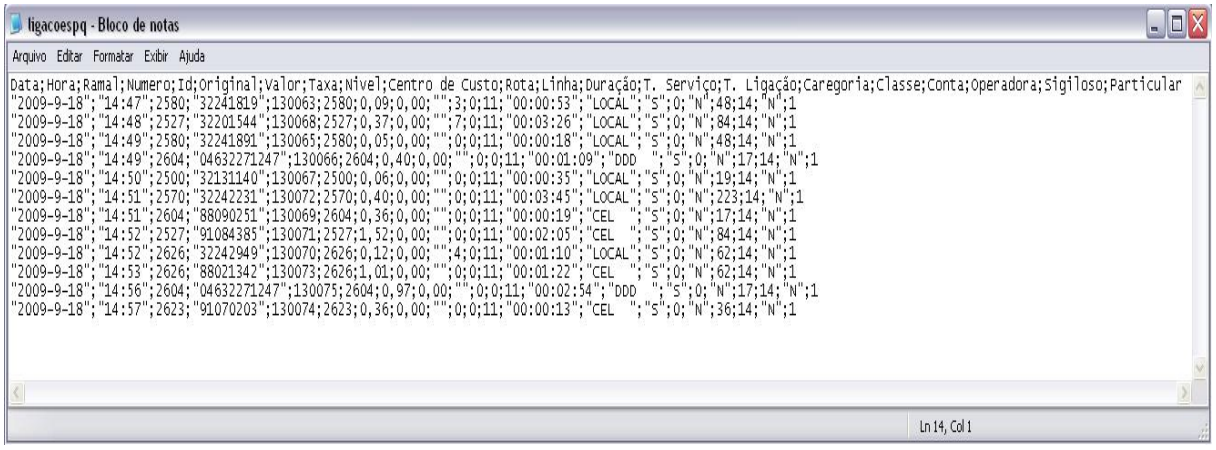

Figura 12 – Exemplo Arquivo de TXT para importação
Fonte: Autoria própria.

Após a importação das ligações, estas ficam disponíveis *on-line* durante um intervalo de tempo, determinado pelo administrador do sistema, para que cada usuário possa identificar suas ligações como particulares ou a trabalho, justificando cada uma delas se necessário. Ao término desse intervalo de tempo, o administrador do sistema deverá avaliar se aceita ou não as justificativas, e, a partir disso definir quais ligações serão cobradas do usuário e o seu valor total.

O administrador do sistema também poderá importar os usuários que estão cadastrados na central telefônica, utilizando-se do mesmo método realizado para as ligações.

Outra funcionalidade que o usuário poderá utilizar é uma agenda telefônica, com isso poderá cadastrar os números telefônicos que desejar e definir um valor padrão que identificará automaticamente se determinado telefone é de ligações particulares ou a trabalho.

4.2 Análise de Requisitos

A análise de requisitos é considerada a primeira etapa do desenvolvimento de um software sendo responsável por definir quais as funcionalidades e restrições que

o sistema deverá possuir, ou seja, definirá o que o sistema deve fazer sem se preocupar em como isto será feito.

Para Bezerra (2003) os requisitos estão divididos em funcionais, não-funcionais e suplementares. Requisitos funcionais representam funcionalidades que o sistema deve realizar atendendo as regras de negócio, requisitos não-funcionais são especificações ou qualidades relacionadas aos requisitos funcionais e requisitos suplementares são demais qualidades que não se enquadram nos requisitos anteriores e pertencem ao sistema como um todo.

O Quadro 3 apresenta os requisitos funcionais do sistema com seus respectivos requisitos não-funcionais.

(continua)

F1 - Logar no Sistema		**Usuário:** Todos		
Descrição: O sistema deve possuir sistema de login para a identificação dos usuários.				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF1.1	Apenas usuários cadastrados podem acessar o sistema	Segurança	( )	(X)
NF1.2	O *login* do usuário deve espirar após 5 minutos de inatividade	Segurança	( )	(X)
NF1.3	Após *login* o usuário deve ser direcionado para seu módulo do sistema, ou seja, administrador para módulo administrativo e usuário para módulo usuário	Interface	( )	(X)
F2 - Configurar sistema de importação de ligações		**Usuário:** Administrador		
Descrição: O sistema deve permitir o administrador especificar quais são os campos a serem importados e qual será o caracter delimitador destes campos para a importação das ligações a partir da central telefônica.				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF2.1	Apenas usuário administrador pode ter acesso a essa função	Segurança	( )	(X)
NF2.2	O sistema deve verificar se todos os campos obrigatórios estão preenchidos	Segurança	( )	(X)
F3 - Configurar sistema de importação de usuários		**Usuário:** Administrador		
Descrição: O sistema deve permitir o administrador especificar quais são os campos a serem importados e qual será o caractere delimitador destes campos para a importação dos usuários a partir da central telefônica.				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF3.1	Apenas usuário administrador pode ter acesso a essa função	Segurança	( )	(X)
NF3.2	O sistema deve verificar se todos os campos obrigatórios estão preenchidos	Segurança	( )	(X)
F4 - Importar ligações		**Usuário:** Administrador		
Descrição: O sistema deve possuir uma funcionalidade que permita o administrador especificar um arquivo .txt para a importação das ligações telefônicas				
Requisitos não-funcionais				

(continua)

Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF4.1	Apenas usuário administrador pode ter acesso a essa função	Segurança	( )	(X)
NF4.2	A importação deve ser feita a partir de um arquivo .txt exportado da central telefônica	Segurança	( )	(X)
NF4.3	A importação deve seguir os parâmetros especificados pelo administrador	Segurança	( )	(X)
NF4.4	A importação não deve demorar mais do que 60 segundos	Performance	(X)	( )
NF4.5	O sistema deve retornar a evolução da importação por meio de barra de progresso	Interface	( )	(X)
NF4.6	O administrador deve poder escolher se deseja sobrescrever ou ignorar ligações já cadastradas	Segurança	( )	(X)
F5 - Importar usuários		**Usuário:** Administrador		
Descrição: O sistema deve possuir uma funcionalidade que permita o administrador especificar um arquivo .txt para a importação dos usuários da central telefônica				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF5.1	Apenas usuário administrador pode ter acesso a essa função	Segurança	( )	(X)
NF5.2	A importação deve ser feita a partir de um arquivo .txt exportado da central telefônica	Segurança	( )	(X)
NF5.3	Os usuários e senhas cadastrados na central telefônica serão os mesmos utilizados no sistema	Segurança	( )	(X)
NF5.4	A importação deve seguir os parâmetros especificados pelo administrador	Segurança	( )	(X)
NF5.5	A importação não deve demorar mais do que 60 segundos	Performance	(X)	( )
NF5.6	O sistema deve retornar a evolução da importação por meio de barra de progresso	Interface	( )	(X)
NF5.7	O administrador deve ter possibilidade de escolher se deseja sobrescrever ou ignorar usuários já cadastrados	Segurança	( )	(X)
F6 - Cadastro de Usuários		**Usuário:** Administrador		
Descrição: O sistema deve possuir ferramenta para inserção, alteração dados dos usuários ou excluir o usuário permanentemente.				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF6.1	Apenas usuário administrador pode ter acesso a essa função	Segurança	( )	(X)
NF6.2	O *login* identificador do usuário deverá ser importado diretamente da central telefônica	Segurança	( )	(X)
F7 - Cadastro de Centrais de Custo		**Usuário:** Administrador		
Descrição: O sistema deve possuir ferramenta para inserção, alteração ou exclusão de centrais de custos.				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF7.1	Apenas usuário administrador pode ter acesso a essa função	Segurança	( )	(X)
NF7.2	O identificador das centrais de custos deverá ser importado diretamente da central telefônica	Segurança	( )	(X)
F8 - Cadastro de Tipos de Serviços		**Usuário:** Administrador		

(conclusão)

Descrição: O sistema deve possuir ferramenta para inserção, alteração ou exclusão de tipos de serviços.				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF8.1	Apenas usuário administrador pode ter acesso a essa função	Segurança	( )	(X)
NF8.2	O identificador dos tipos de serviços deverá ser importado diretamente da central telefônica	Segurança	( )	(X)
F9 - Justificativa de Ligações		**Usuário:** Administrador e usuários		
Descrição: Todas as ligações devem ser identificadas como particulares ou a trabalho pelo usuários no sistema.				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF9.1	Administrador e usuários podem ter acesso a essa função	Segurança	( )	(X)
NF9.2	O administrador deve habilitar o módulo "justificativa de ligações" durante um intervalo de tempo	Especificação	( )	(X)
NF9.3	O administrador deve determinar quais ligações podem ser justificadas	Especificação	( )	(X)
NF9.4	Os usuários de identificar cada ligação registrada em seu *login* como particular ou a trabalho	Especificação	(X)	( )
NF9.5	As ligações a trabalho devem ser justificadas	Especificação	(X)	( )
NF9.6	Após o intervalo de tempo, o administrador deverá validar as justificativas	Segurança	( )	(X)
F10 - Agenda Telefônica		**Usuário:** Usuários		
Descrição: Cada usuário possuirá uma agenda telefônica, em que poderá identificar automaticamente os destinatários das ligações e definir um valor padrão (particular ou a trabalho) para cada um				
Requisitos não-funcionais				
Nome	**Descrição**	**Categoria**	**Desejável**	**Permanente**
NF10.1	Apenas usuários podem ter acesso a essa função	Segurança	( )	(X)

Quadro 3 – Requisitos Funcionais e Não-Funcionais
Fonte: Autoria própria.

Os requisitos suplementares estão especificados no Quadro 4. (continua)

Nome	Descrição	Categoria	Desejável	Permanente
S1 - Independência de Sistema Operacional	O sistema deverá ser executado em qualquer sistema operacional que possua navegador *Web*	Portabilidade	( )	(X)
S2 - Independência de Navegador Web	O sistema deverá ser implementado para executar nos principais navegadores *Web*, preferencialmente os gratuitos.	Portabilidade	( )	(X)
S3 – Suporte a concorrência	O sistema deverá suportar inúmeros acessos simultâneos	Concorrência	( )	(X)
S4 - Perfis dos Usuários	O sistema deve possuir os seguintes perfis de usuário: - Administrador: responsável por manter o sistema atualizado, com os dados referentes a todas as ligações e suas respectivas justificativas; - Usuário: deverá ter acesso apenas as suas próprias ligações, podendo justificá-las.	Segurança	( )	(X)

(conclusão)

S5 – Login Administrador por Arquivo	Os dados de *login* do Administrador do sistema devem estar em um arquivo .txt.	Segurança	( )	(X)

Quadro 4 – Requisitos Suplementares
Fonte: Autoria própria.

4.3 Regras de Negócios

Segundo IBM (2013) regras de negócio são declarações de políticas, condições ou diretrizes que devem ser cumpridas pelo sistema com o objetivo de atender as demandas do negócio. Regras de negócio são restrições impostas que regulamentam o comportamento de um procedimento operacional do sistema e essas políticas devem definidas pela administração da empresa.

No Quadro 5 são apresentadas as regras de negócio referentes ao sistema a ser desenvolvido.

(continua)

Padrões de Usuários (RN01)	
Descrição	O sistema deve possuir dois tipos de usuários: Administrador e usuários padrões. Cada tipo de usuário deverá possuir um módulo próprio.
Identificação (RN02)	
Descrição	O usuário deve se identificar através de um “usuário” e “senha” previamente importados ou cadastrados para obter acesso ao seu módulo do sistema.
Importação de Usuários (RN03)	
Descrição	No módulo administrador deverá existir uma funcionalidade que possibilite a importação dos usuários que se encontram cadastrados na central telefônica existente na universidade a partir de um arquivo “.txt”. Cada usuário pode ser importado apenas uma vez.
Importação de Ligações (RN04)	
Descrição	No módulo administrador deverá existir uma funcionalidade que possibilite a importação das ligações feitas na central telefônica da universidade a partir de um arquivo “.txt”. Cada ligação pode ser importada apenas uma vez.
Padronização dos Campos de Importação (RN05)	
Descrição	O administrador do sistema poderá definir quais campos podem ser importados da central telefônica, tanto para ligações quanto para usuários. Também poderá definir qual será o caractere delimitador dos campos a serem importados.
Justificativa de Ligações (RN06)	
Descrição	O administrador do sistema poderá definir um período do mês para que os usuários identifiquem quais ligações são particulares e quais são a serviço.
Validação de Justificativas (RN07)	
Descrição	Após o período de justificativa de ligações o administrador poderá definir quais justificativas são aceitas e quais não.

(conclusão)

Cobrança de Ligações (RN08)	
Descrição	As ligações que forem identificadas como particulares ou que tiverem sua justificativa recusada devem ser cobradas de cada usuário.

Quadro 5 - Regras de Negócio
Fonte: Autoria própria.

4.4 Documentação de Atores

Todos os agentes externos ao sistema e que interajam com ele devem ser considerados como atores dos mesmos. A seguir são apresentados os atores que se relacionam com sistema:

- **Administrador:** funcionário responsável pela manutenção do sistema, com a responsabilidade de deixar o sistema ativo e com dados sempre atualizados.
- **Usuários:** tem a responsabilidade de identificar quais ligações lhe pertence e quais foram particulares para que seja feita cobrança.

4.5 Diagrama de Casos de Uso

Diagrama de casos de uso representa os processos que o sistema deve realizar, sem especificar como esse processo deve ser feito, e também quais atores se relacionam com cada processo e de que maneira. Para Eduardo Bezerra casos de uso é uma representação das funcionalidades externamente observáveis e do sistema e dos elementos externos ao sistema que interagem com eles. (BEZERRA, 2003).

A Figura 13 apresenta o Diagrama de Casos de Uso proposto neste trabalho. A sigla RUD utilizada em alguns casos de uso significa leitura (*read*), atualização (*update*) e exclusão (*delete*). Nesses casos de uso não há a inserção (*create*), pois a inserção é realizada através de importação dos dados da central telefônica.

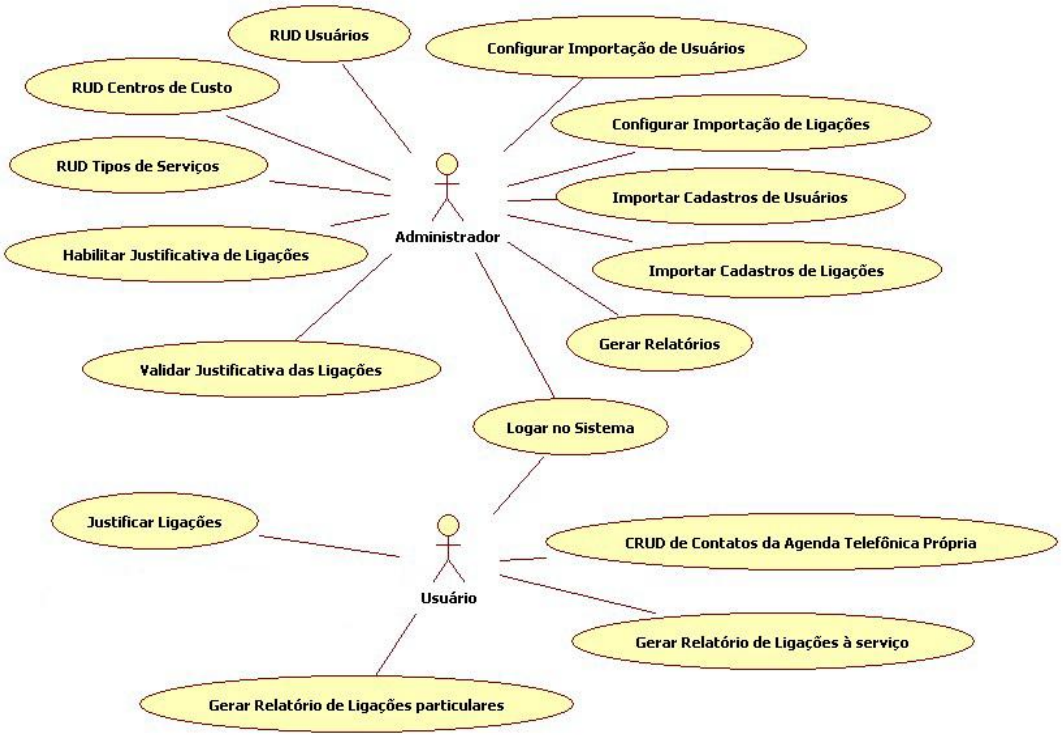

Figura 13 –Diagrama de Casos de Uso
Fonte: Autoria própria.

4.6 Casos de Uso Expandidos

Casos de uso expandido podem ser descritos com um detalhamento de todos os casos de uso detectados anteriormente, apresentando nome, identificador, ator primário, ator secundário, precondições, fluxos alternativos ou de exceção, pós-condições. Os principais casos de uso expandidos são apresentados nos quadros 6 a 20.

(continua)

Logar no sistema (CSU01)

Sumário: Os usuários desejam ter acesso ao sistema, para isso devem inserir seu código e senha.

Ator Primário: Administrador e Usuários.

Atores Secundários: -

Precondições: O usuário deve estar cadastrado no sistema.

Fluxo Principal

1. O usuário insere seu código e senha no sistema.
2. O sistema valida os dados informados e direciona para página inicial.

Fluxos de Exceção

2. Se o usuário não possui um perfil com permissão necessária o sistema informa o usuário.
   2.1. Retorna ao passo 1 do fluxo principal.

(conclusão)

Pós-condições: O usuário tem acesso ao sistema.

Regras de Negócio: RN01, RN02.

Requisitos Relacionados: F1.

Quadro 6 - Caso de Uso Expandido Logar no Sistema
Fonte: Autoria própria.

Configurar Importação de Usuários (CSU02)

Sumário: O administrador do sistema deve definir no sistema como será realizada a importação dos usuários da central telefônica.

Ator Primário: Administrador

Atores Secundários: -

Precondições: -

Fluxo Principal

1. Em uma funcionalidade específica do sistema o administrador deverá definir qual é a posição que os campos a serem importados se encontram no arquivo ".txt".

2. Em seguida deve definir qual será o caractere delimitador dos campos.

3. O sistema valida se está correto o preenchimento dos campos obrigatórios e realiza a gravação no banco de dados.

Fluxos de Exceção

3. Se o administrador não preencher adequadamente os campos.
   3.1. O sistema informa o campo a ser preenchido e não realiza a gravação no banco de dados.

Pós-condições: O sistema estará configurado para a realização da importação de usuários.

Regras de Negócio: RN03, RN05.

Requisitos Relacionados: F3.

Quadro 7 - Caso de Uso Expandido Configurar Importação de Usuários
Fonte: Autoria própria.

(continua)

Configurar Importação de Ligações (CSU03)

Sumário: O administrador do sistema deve definir no sistema como será realizada a importação das ligações da central telefônica.

Ator Primário: Administrador

Atores Secundários: -

Precondições: -

Fluxo Principal

1. Em uma funcionalidade específica do sistema o administrador deverá definir qual é a posição que os campos a serem importados se encontram no arquivo ".txt".

2. Em seguida deve definir qual será o caractere delimitador dos campos.

(conclusão)

3. O sistema valida se está correto o preenchimento dos campos obrigatórios e realiza a gravação no banco de dados.

Fluxos de Exceção

3. Se o administrador não preencher adequadamente os campos.
3.1. O sistema informa o campo a ser preenchido e não realiza a gravação no banco de dados.

Pós-condições: O sistema estará configurado para a realização da importação das ligações.

Regras de Negócio: RN04, RN05.

Requisitos Relacionados: F2.

Quadro 8 - Caso de Uso Expandido Configurar Importação de Ligações
Fonte: Autoria própria.

Importação de Usuários (CSU04)

Sumário: O administrador do sistema pode importar, a partir de um arquivo .txt, os usuários que estão cadastrados na central telefônica.

Ator Primário: Administrador

Atores Secundários: -

Precondições: O sistema deve estar com os parâmetros de importação de usuários definidos.

Fluxo Principal

1. Na funcionalidade específica para importação, o administrador deve especificar um arquivo .txt que contenha a relação de usuários cadastrados no central telefônica e ativar a importação.
2. O sistema verifica se os campos configurados previamente constam no arquivo em um padrão adequado.
3. Antes de cada inserção de usuário, o sistema verifica se ele já não está cadastrado.
4. O sistema realiza a importação dos usuários para banco de dados.

Fluxos de Exceção

2. Se o arquivo selecionado não seguir os padrões de importação.
2.1. O sistema informa que o arquivo é inválido e não realiza a importação.

3. Se o usuário já estiver cadastrado no sistema.

3.1. A inserção deste usuário é desconsiderada e o processo continua.

Pós-condições: O sistema estará alimentado com os mesmo usuários da central telefônica.

Regras de Negócio: RN03.

Requisitos Relacionados: F5.

Quadro 9 - Caso de Uso Expandido Importação de Usuários
Fonte: Autoria própria.

Importação das Ligações (CSU05)

Sumário: O administrador do sistema pode importar, a partir de um arquivo .txt, as ligações que foram realizadas pelos usuários da central telefônica.

Ator Primário: Administrador

Atores Secundários: -

Precondições: O sistema deve estar com os parâmetros de importação de ligações definidos.

Fluxo Principal

1. Na funcionalidade específica para importação, o administrador deve especificar um arquivo .txt que contenha a relação das ligações realizadas no central telefônica e ativar a importação.
2. O sistema verifica se os campos configurados previamente constam no arquivo em um padrão adequado.
3. Antes da inserção de cada ligação, o sistema verifica se ela já não está cadastrada.
4. O sistema realiza a importação das ligações para banco de dados.

Fluxos de Exceção

2. Se o arquivo selecionado não seguir os padrões de importação.
   2.1. O sistema informa que o arquivo é invalido e não realiza a importação.
3. Se a ligação já estiver cadastrada no sistema.
   3.1. A inserção desta ligação é desconsiderada e o processo continua.

Pós-condições: O sistema estará alimentado com as ligações realizadas na central telefônica.

Regras de Negócio: RN04.

Requisitos Relacionados: F4.

Quadro 10 - Caso de Uso Expandido Importação das Ligações
Fonte: Autoria própria.

(continua)

RUD de Usuários (CSU06)

Sumário: O administrador pode consultar, editar ou remover os usuários do sistema. A inserção de usuários apenas pode ser realizada pela importação da central telefônica.

Ator Primário: Administrador

Atores Secundários: -

Precondições: -

Fluxo Principal

1. O usuário escolhe a operação:

   Variante Consultar.

   Variante Editar.

   Variante Remover

Variantes

(conclusão)

1.1. Consultar

1.1.1. O sistema apresenta uma lista com usuário e nome dos usuários.

1.1.2. O administrador seleciona um elemento da lista.

1.1.3. O sistema apresenta todos os dados do usuário selecionado.

1.2. Editar

1.2.1. O sistema apresenta uma lista com usuário e nome dos usuários.

1.2.2. O administrador seleciona qual usuário deseja editar, altera os valores cadastrados e confirma a edição.

1.3. Remover

1.3.1. O sistema apresenta uma lista com usuário e nome dos usuários.

1.3.2. O administrador seleciona um elemento a ser removido e confirma o procedimento.

Fluxos de Exceção

1.2.2. Campos preenchidos de maneira incorreta
1.2.2.a. O sistema informa o preenchimento inadequado e não realiza a edição.

Pós-condições: -

Requisitos Relacionados: F6.

Quadro 11 - Caso de Uso Expandido RUD Usuários
Fonte: Autoria própria.

(continua)

RUD de Centros de Custo (CSU07)

Sumário: O administrador do pode consultar, editar ou remover os Centros de Custo das ligações. A inserção de um Centro de Custo apenas pode ser realizada pela importação da central telefônica.

Ator Primário: Administrador

Fluxo Principal

1. O usuário escolhe a operação:

1.1 Variante Consultar.

1.1 Variante Editar.

1.2 Variante Remover

Variantes

1.1 Consultar

1.1.1 O sistema apresenta uma lista com código e nome do centro de custo.

1.1.2 O administrador seleciona um elemento da lista.

1.1.3 O sistema apresenta todos os dados referentes ao centro de custo selecionado.

1.2 Editar

1.2.1 O sistema apresenta uma lista com código e nome do centro de custo.

1.2.2 O administrador seleciona qual centro de custo deseja editar, altera os valores cadastrados e confirma a edição.

1.3 Remover

(conclusão)

1.3.1 O sistema apresenta uma lista com código e nome do centro de custo.

1.3.2 O administrador seleciona um elemento a ser removido e confirma o procedimento.

Fluxos de Exceção

1.2.2. Campos preenchidos de maneira incorreta

1.2.2.a. O sistema informa o preenchimento inadequado e não realiza a edição.

Requisitos Relacionados: F7.

Quadro 12 - Caso de Uso Expandido RUD Centros de Custo
Fonte: Autoria própria.

RUD de Tipos de Serviço (CSU08)

Sumário: O administrador do pode consultar, editar ou remover os Tipos de Serviços cadastrados. A inserção de um Tipo de serviço apenas pode ser realizada pela importação da central telefônica.

Ator Primário: Administrador

Fluxo Principal

1. O usuário escolhe a operação:

1.1 Variante Consultar.

1.2 Variante Editar.

1.3 Variante Remover

Variantes

1.1 Consultar

1.1.1 O sistema apresenta uma lista com código e nome do tipo de serviço.

1.1.2 O administrador seleciona um elemento da lista.

1.1.1 O sistema apresenta todos os dados referentes ao tipo de serviço selecionado.

1.2 Editar

1.2.1 O sistema apresenta uma lista com código e nome do tipo de serviço.

1.2.2 O administrador seleciona qual tipo de serviço deseja editar, altera os valores cadastrados e confirma a edição.

1.3 Remover

1.3.1 O sistema apresenta uma lista com código e nome do tipo de serviço.

1.3.2 O administrador seleciona um elemento a ser removido e confirma o procedimento.

Fluxos de Exceção

1.2.2. Campos preenchidos de maneira incorreta

1.2.2.a. O sistema informa o preenchimento inadequado e não realiza a edição.

Pós-condições: -

Requisitos Relacionados: F8.

Quadro 13 - Caso de Uso Expandido RUD Tipo de Serviço
Fonte: Autoria própria.

Habilitar Justificativa de Ligações (CSU09)

Sumário: O administrador do sistema deve habilitar uma funcionalidade que torna possível os usuários identificarem se suas ligações, durante um intervalo de tempo, foram a serviço ou particulares.

Ator Primário: Administrador

Atores Secundários: -

Precondições: Usuários, ligações, tipos de serviços e centros de custos devem ter sidos importados anteriormente.

Fluxo Principal

1. O administrador deve selecionar por qual intervalo de tempo as justificativas de ligações ficaram habilitadas para os usuários.
2. O administrador deve selecionar de quais datas as ligações deveram ser justificadas.
3. O administrador ativa as justificativas de ligações.

Fluxos de Exceção

1. Datas preenchidas de maneira incorreta
   1.a. O sistema verifica se as datas preenchidas já não passaram ou se estão preenchidas na ordem incorreta. Caso algum problema seja detectado o sistema informa o erro.
2. Datas preenchidas de maneira incorreta
   2.a. O sistema verifica se as datas preenchidas já não passaram ou se estão preenchidas na ordem incorreta. Caso algum problema seja detectado o sistema informa o erro.

Pós-condições: Os usuários terão uma nova funcionalidade em seu modulo que os permite identificar quais ligações são a trabalho e quais são a serviço durante o intervalo de tempo estipulado pelo administrador.

Regras de Negocio: RN06

Requisitos Relacionados: F9.

Quadro 14 - Caso de Uso Expandido Habilitar Justificativa de Ligações
Fonte: Autoria própria.

(continua)

Justificar de Ligações (CSU10)

Sumário: Todas as ligações realizadas pelo usuário durante um intervalo de tempo serão listadas para o mesmo. O usuário deve identificar quais ligações foram feitas a serviço e quais foram particulares.

Ator Primário: Usuário.

Atores Secundários: -

Precondições: O módulo de justificativa de ligações deve estar habilitado.

Fluxo Principal

1. O sistema apresentará uma listagem com as ligações que o usuário realizou.
2. O usuário deve selecionar quais ligações são particulares e quais são a trabalho, justificando as últimas.

(conclusão)

Fluxos de Exceção

2. Preenchimento incorreto

2.a. O sistema verifica o preenchimento de todos os campos, caso algum problema seja identificado o sistema informa erro.

Regras de Negocio: RN06

Requisitos Relacionados: F9.

Quadro 15 - Caso de Uso Expandido Justificar de Ligações
Fonte: Autoria Própria.

Validar Justificava de Ligações (CSU11)

Sumário: As ligações que forem identificadas como a serviço devem ser justificadas. Essas justificativas devem ser avaliadas pelo administrador do sistema, para verificar se serão cobradas ou não.

Ator Primário: Administrador.

Atores Secundários: -

Precondições: O módulo de justificativa de ligações deve estar inativo.

Fluxo Principal

1. O sistema apresentará uma listagem com todos os usuários que justificaram suas ligações.
2. O administrador selecionará de qual usuário deseja validar as justificativas.
3. As justificativas do usuário aparecerão e o administrador deverá selecionar se aceita ou não.

Fluxos de Exceção

1. Modulo de justificativa ativo.

1.a. Se o módulo de justificativa estiver ativo o sistema informará para o administrador.

Regras de Negocio: RN07

Requisitos Relacionados: F9.

Quadro 16 - Caso de Uso Expandido Validar Justificativa de Ligações
Fonte: Autoria própria.

(continua)

Gerar Relatórios (CSU12)

Sumário: O administrador do sistema poderá visualizar relatórios que apresentaram dados sobre o fluxo de ligações particulares ou a serviço de todos os usuários da central telefônica da UTFPR – Câmpus Pato Branco.

Ator Primário: Administrador.

Atores Secundários: -

Precondições: -.

Fluxo Principal

(conclusão)

1. O administrador seleciona qual tipo de relatório deseja visualizar, se de ligações particulares e se de ligações a serviço, e define de qual intervalo de tempo.
2. O sistema retorna o relatório solicitado.

Fluxos de Exceção

1. Intervalo de tempo incorreto

1.a. O sistema informa que o intervalo de tempo selecionado é invalido.

Regras de Negocio: -

Requisitos Relacionados: -

Quadro 17 - Caso de Uso Expandido Gerar Relatórios
Fonte: Autoria própria.

Relatório de Ligações Particulares (CSU13)

Sumário: Os usuários poderão acessar mensalmente um relatório com todas as ligações identificadas como particulares e as que tiveram suas justificativas recusadas.

Ator Primário: Usuário.

Atores Secundários: -

Precondições: -.

Fluxo Principal

1. O usuário seleciona de qual mês deseja visualizar suas ligações particulares ou com a justificativa recusada.
2. O sistema retorna o relatório solicitado.

Fluxos de Exceção

1. Mês incorreto

1.a. O sistema informa que o mês selecionado é invalido.

Regras de Negocio: -

Requisitos Relacionados: -

Quadro 18 - Caso de Uso Expandido Relatório de Ligações Particulares
Fonte: Autoria própria.

(continua)

Relatório de Ligações à Serviço (CSU14)

Sumário: Os usuários poderão acessar mensalmente um relatório com todas as ligações identificadas como a serviço e as que tiveram suas justificativas aceitas.

Ator Primário: Usuário.

Atores Secundários: -

Precondições: -.

Fluxo Principal

1. O usuário seleciona de qual mês deseja visualizar suas ligações a serviço ou com a justificativa aceitas.

(conclusão)

2. O sistema retorna o relatório solicitado.

Fluxos de Exceção

1. Mês incorreto

1.a. O sistema informa que o mês selecionado é invalido.

Regras de Negocio: -

Requisitos Relacionados: -

Quadro 19 - Caso de Uso Expandido Relatório de Ligações à Serviço
Fonte: Autoria própria.

CRUD de Contatos da Agenda Telefônica Própria (CSU15)

Sumário: Cada usuário possuirá uma agenda telefônica particular, facilitando a identificação posterior de ligações particulares ou a serviço.

Ator Primário: Usuário.

Fluxo Principal

1. O usuário informa qual é o numero de telefone a ser cadastrado.
2. Após deverá identificar um padrão de tipo de ligação para o número informado: particular ou a serviço.
3. Nos números a serviço ele poderá identificar o destino e gerar uma justificativa padrão.

Fluxos de Exceção

Regras de Negocio: -

Requisitos Relacionados: F10

Quadro 20 - Caso de Uso Expandido CRUD de Contatos da Agenda Telefônica
Fonte: Autoria própria.

4.7 Diagrama de Entidades e Relacionamentos

O Diagrama de Entidades e Relacionamentos tem o objetivo de expressar graficamente a estrutura lógica do banco de dados a ser desenvolvido posteriormente. Este diagrama é constituído por pelo menos 3 (três) componentes: as entidades, os atributos e os relacionamentos. Uma entidade é o objeto real a ser representado no sistema. Atributos são características que descrevem a entidade as quais pertencem. Relacionamento indica por qual maneira duas ou mais entidades se associam.

A Figura 14 apresenta o digrama de entidades e relacionamentos do sistema proposto. Como pode ser visualizado nessa figura a principal entidade do sistema é a tabela de ligações (tab_ligaçoes). Esta contém todas as informações referentes às ligações feitas pelos usuários, como ramal de origem, número de destino, duração, valor entre outros. Seus dados são preenchidos por meio da funcionalidade de importação da central telefônica.

Para poder realizar a importação das ligações o sistema deve estar configurado. Esta configuração deve ser feita pelo administrador do sistema, ficando armazenada na tabela de importação de ligações (tab_importacao_ligacoes).

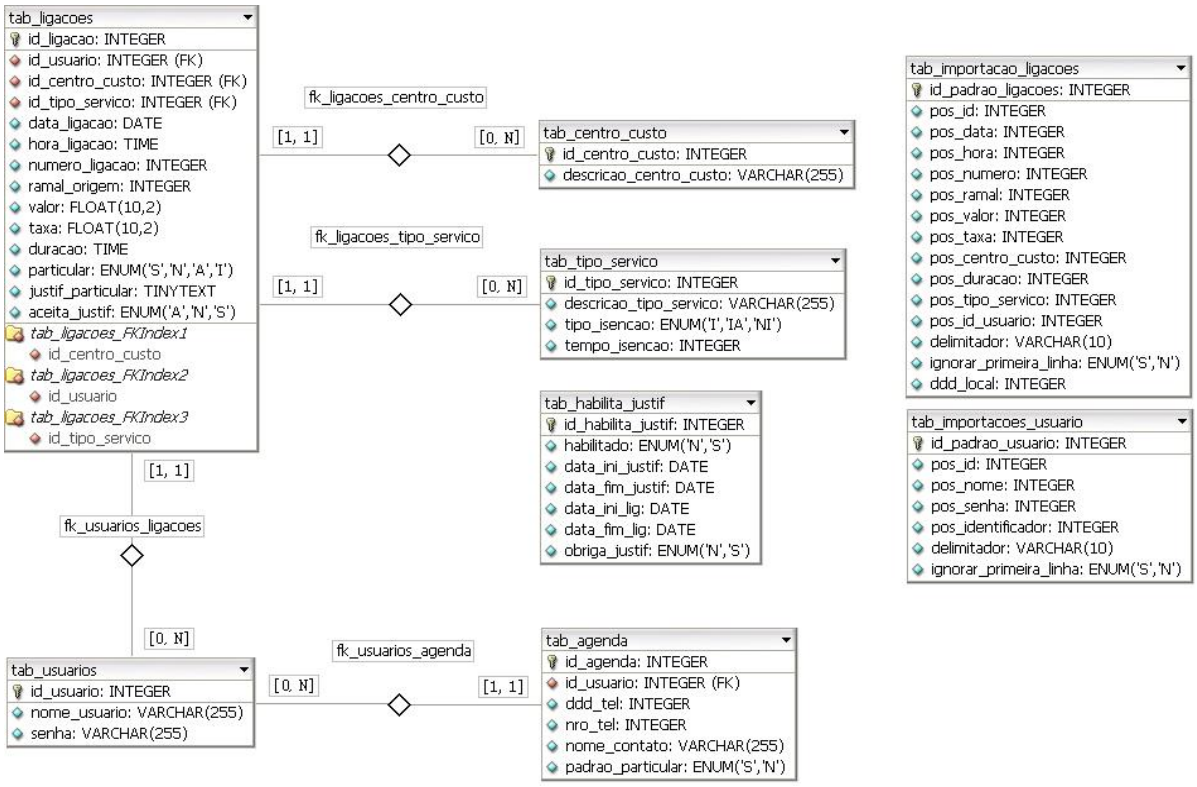

Figura 14 – Diagrama de Entidades e Relacionamentos
Fonte: Autoria própria.

Outra tabela de configuração do sistema é a de importação de usuários (tab_importacoes_usuarios), em que será armazenado as posições dos campos e o delimitador que possibilita a importação.

A tabela de usuários (tab_usuarios) armazena as informações referentes aos usuários cadastrados na central telefônica. Esta possui relacionamento com a tabela de ligações, sendo que uma ligação pertence a apenas um usuário e um usuário possui muitas ligações.

Outra tabela que se relaciona com a tabela de ligações é a de centro de custos (tab_centro_custo). Centros de custo são grupos a quais os usuários pertencem e possibilitando um controle de gastos telefônicos por setores.

A tabela tipo de serviço (tab_tipo_servico) também está relacionada com a tabela de ligações. Esta tabela tem por finalidade identificar qual tipo de ligação foi realizada. Entre os tipos tem-se Local, DDD, DDI, Celular, entre outros.

Os usuários podem cadastrar os números de telefones para os quais mais realizam ligações e identificar previamente se esse destinatário pertence às ligações particulares ou a serviço. Os dados ficarão armazenados na tabela agenda que se relaciona diretamente com a tabela de usuários.

Todas as ligações realizadas pelos usuários devem ser identificadas como: a serviço ou particular. Porém, esta identificação apenas será feita em um intervalo de tempo que o administrador do sistema permitir. Para isso, a tabela de habilitação de justificativa armazena o intervalo de tempo que a justificativa (tab_halibita_justif) de ligações estará disponível.

4.8 Digrama de Classes

Diagrama de Classes é a representação estática da estrutura do sistema por meio de classes, relações e métodos, tendo foco nas principais interfaces da arquitetura.

A Figura 15 apresenta o Diagrama de Classes do sistema em estudo.

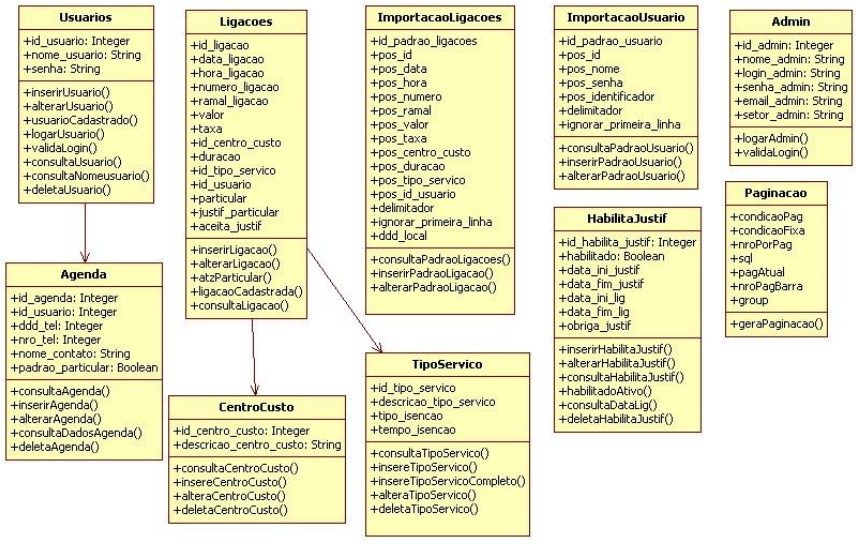

Figura 15 – Diagrama de Classes
Fonte: Autoria própria.

Este diagrama apresenta basicamente as entidades que foram analisadas anteriormente e os métodos que cada classe possui. Além dessas, duas outras classes foram criadas, a Admin e a Paginação. A primeira é responsável pela comunicação com o arquivo .txt que contém os dados do administrador do sistema. A classe Paginação é responsável apresentar todos os dados do sistema em páginas separadas, diminuindo a quantidade de itens que usuário visualiza ao mesmo tempo, tornando a navegação mais acessível. Por exemplo, em uma pesquisa que ligações de determinado usuário, a quantidade de dados que retorna pode ser muito grande, a classe paginação é responsável por dividir o resultado em inúmeras paginas, facilitando o uso no sistema.

4.9 Apresentação do Sistema

O sistema desenvolvido para este trabalho é dividido em dois módulos: Administrativo e Usuário. O primeiro é acessado apenas pelo administrador do sistema e tem a função de gerenciar todo o sistema, configurando-o, importando as ligações da central telefônica e habilitando-as para a justificativa dos usuários. O módulo do usuário tem as funcionalidades que torna possível o usuário identificar as suas ligações e justificá-las. A Figura 16 apresenta a tela de *login* do console administrativo.

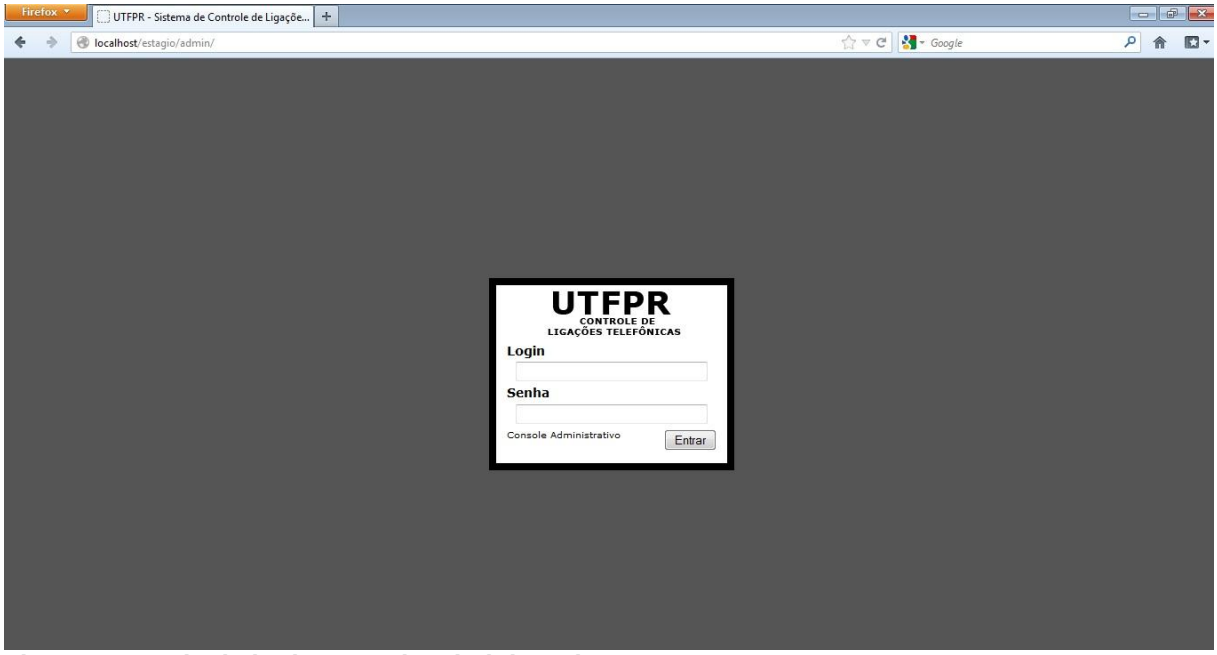

Figura 16 – Tela de login console administrativo
Fonte: Autoria própria.

Conforme o levantamento de requisitos, o *login* e senha do administrador encontra-se armazenado em um arquivo .txt e o acesso no sistema é realizado pela comunicação com este arquivo.

A tela principal do modulo administrativo é simples, apresentando apenas qual usuário esta conectado ao sistema e um menu horizontal como pode ser visualizado na Figura 17.

Figura 17– Tela principal do console administrativo
Fonte: Autoria própria.

A primeira opção que o administrador possui no menu é a de cadastros. A Figura 18 mostra os itens que o menu de cadastros possui. Por esta opção é possível gerenciar o cadastro de todos os usuários do sistema, os tipos de serviço e os centros de custo que a central telefônica possui.

Figura 18– Menu do console administrativo
Fonte: Autoria própria.

A Figura 19 apresenta a tela inicial do cadastro de usuários. Nessa tela existe uma listagem de todos os usuários já importados para o sistema e campos para busca de um usuário especifico. Também é possível visualizar que no topo da

listagem existe uma ferramenta de paginação que facilita a navegação pela listagem de usuários.

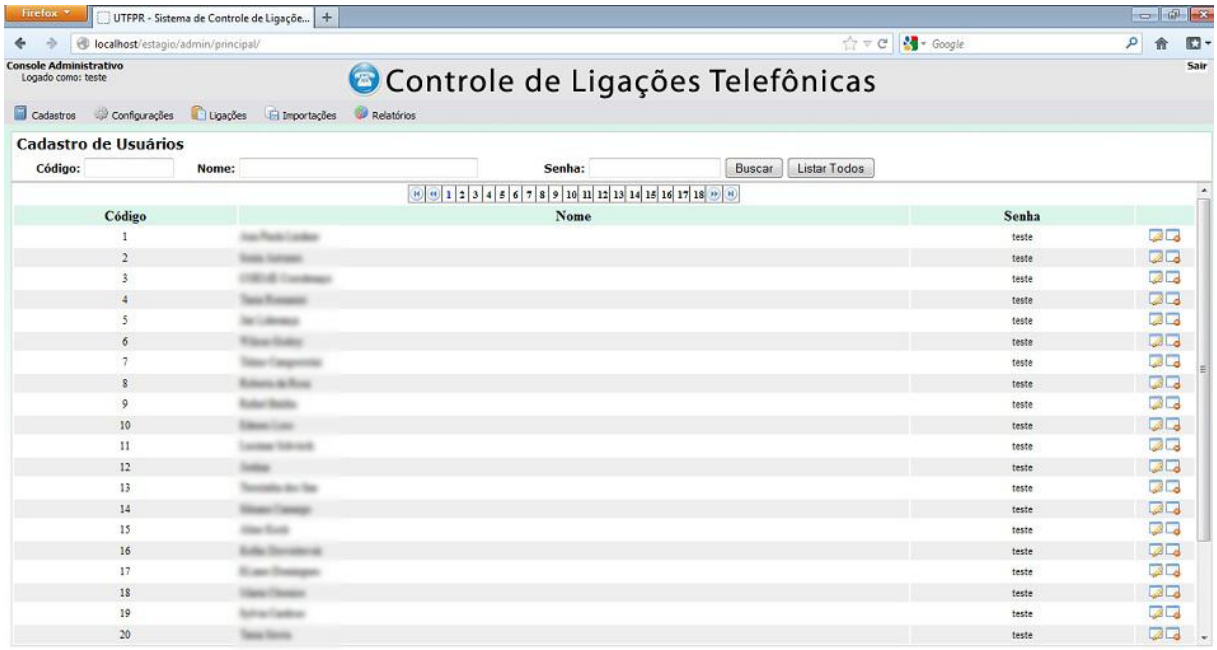

Figura 19 – Cadastro de Usuários
Fonte: Autoria própria.

Cada usuário listado possui dois ícones para a manipulação de suas informações. O primeiro é para dar acesso à tela de edição de informações do cadastro de usuários, esta tela pode ser visualizada na Figura 20.

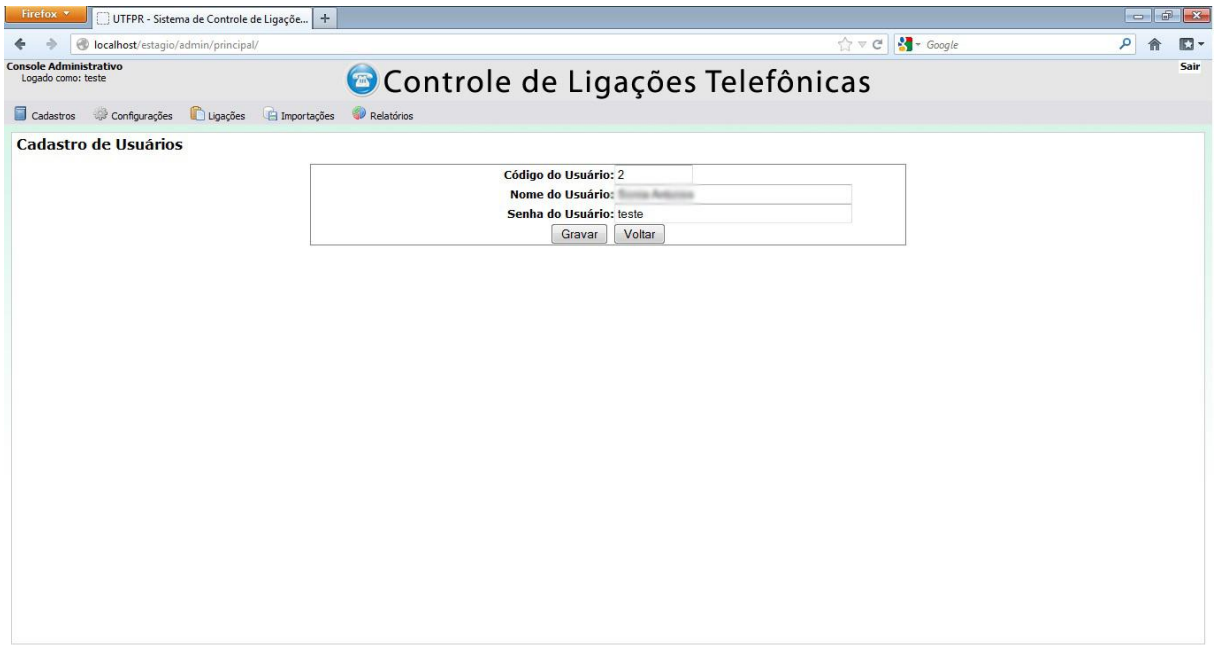

Figura 20– Edição do cadastro de um usuário
Fonte: Autoria própria.

Todas as telas de cadastro do sistema possuem validação para o não preenchimento de algum campo obrigatório ou para o preenchimento indevido. Nesses casos o sistema retorna uma mensagem de erro e não permite a conclusão da operação. Um exemplo desta mensagem é apresentado na Figura 21.

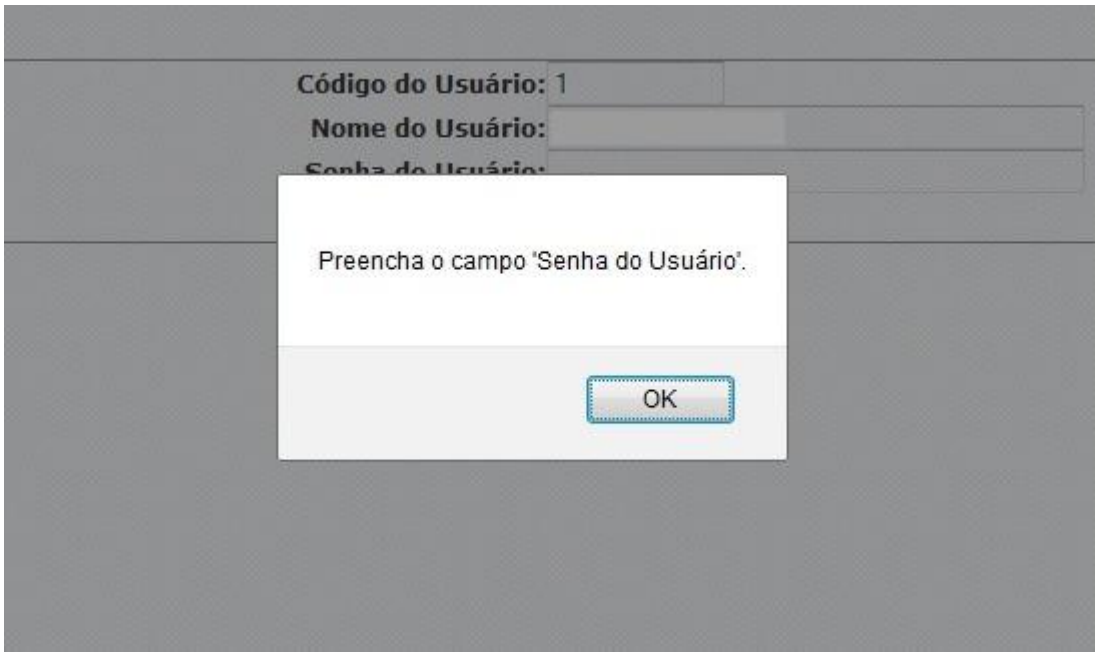

Figura 21– Erro no preenchimento de um campo
Fonte: Autoria própria.

O segundo ícone que aparece na listagem é o que permite a exclusão de um usuário, mas antes da efetivação da ação uma confirmação é exigida. Esta confirmação é apresentada na Figura 22.

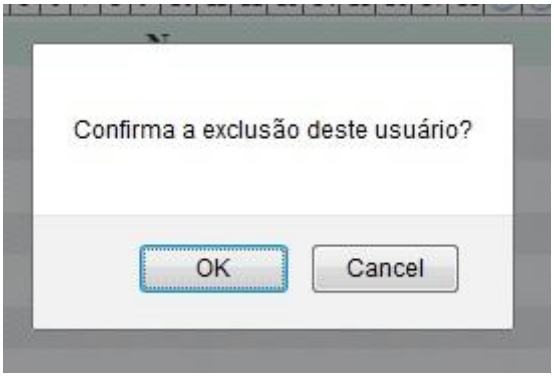

Figura 22 – Confirmação da exclusão de um usuário
Fonte: Autoria própria.

No menu de cadastro também existe as opções de Cadastro de Tipos de Serviço e de Centros de Custos. Esses possuem as mesmas características do cadastro de usuários, porem permitem a inserção de um novo item. A inserção de um usuário só é possível por meio da importação de usuários diretamente da central telefônica. Uma tela de inserção de tipo de serviço é apresentada na Figura 23.

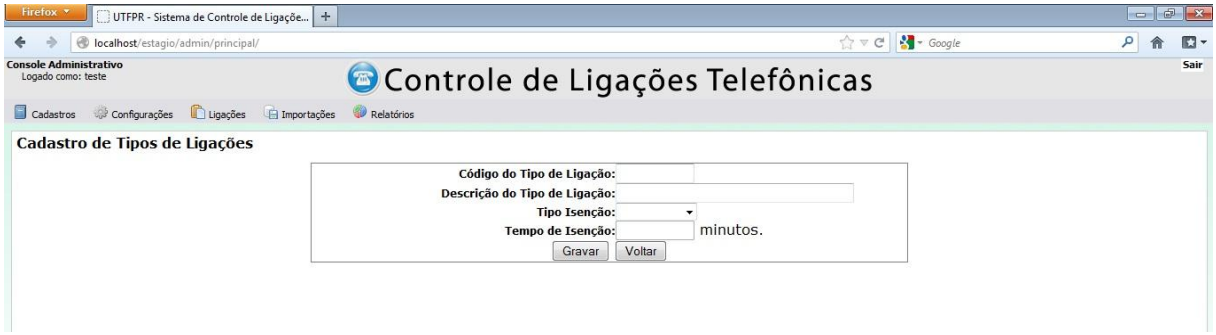

Figura 23– Inserção de um tipo de serviço
Fonte: Autoria própria.

O segundo menu que o console administrativo possui é o de configuração que pode ser visualizado na Figura 24.

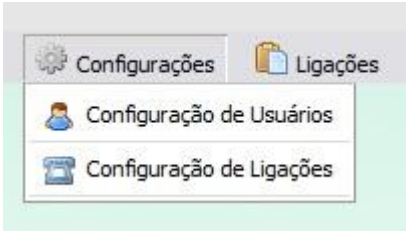

Figura 24– Menu de configurações
Fonte: Autoria própria.

A tela de configuração da importação de usuários é apresentada na Figura 25. O administrador do sistema deve especificar em qual posição da linha do arquivo .txt se encontra cada um dos campos necessários para o cadastro de um usuário. Também deve especificar o delimitador que separa esses campos e se a primeira linha do arquivo ou linha de cabeçalho deve ser ignorada.

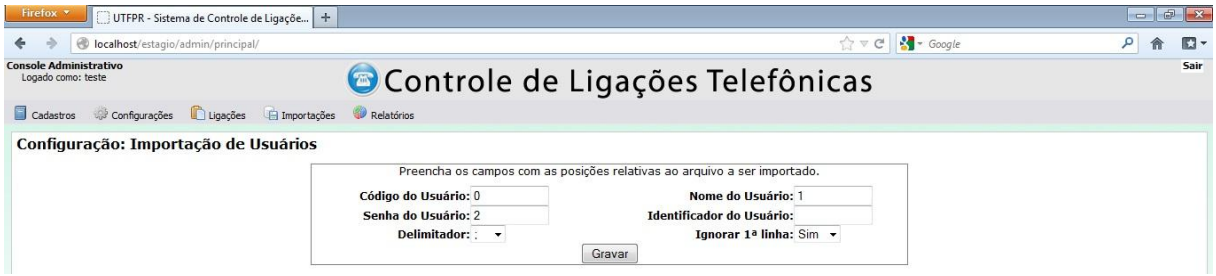

Figura 25– Configuração da importação de usuários
Fonte: Autoria própria.

A tela de configuração da importação de ligações funciona da mesma maneira que a de importação de usuário. Porém essa tela, que pode ser visualizada na Figura 26, permite habilitar a justificativa de ligações para os usuários do sistema.

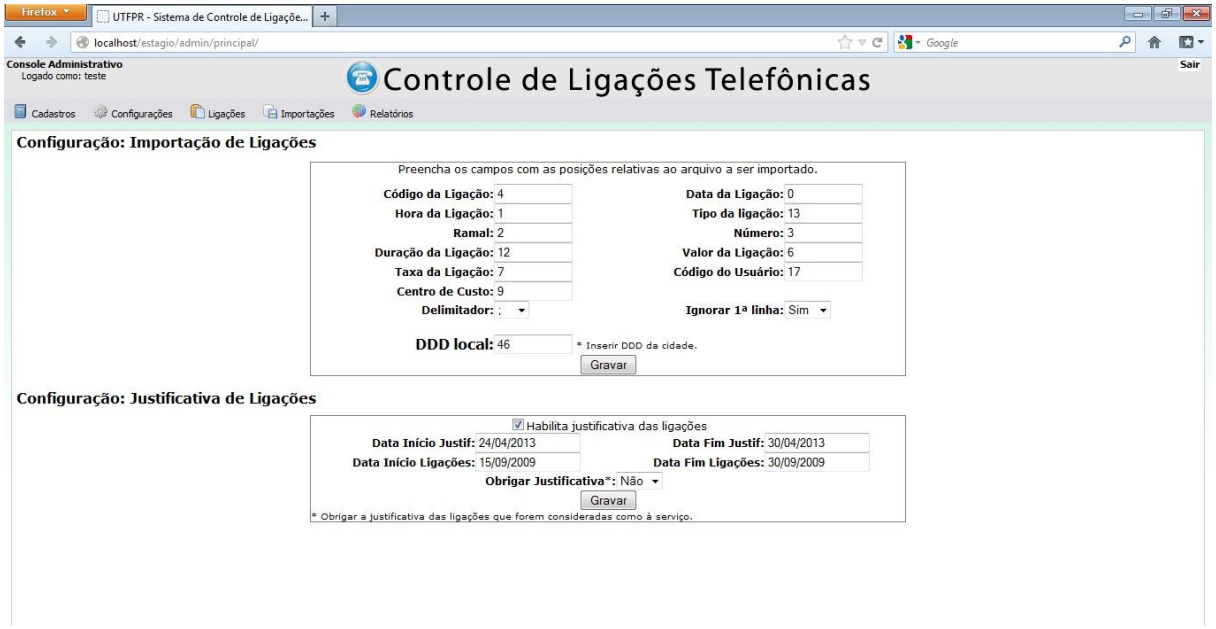

Figura 26– Configuração da importação de ligações
Fonte: Autoria própria.

Para habilitar a justificativa de ligações o administrador deve especificar o intervalo de tempo que a ferramenta de justificativa ficará habilitada para os usuários, de qual intervalo de tempo as ligações serão justificadas e se esta justificativa é obrigatória ou não.

O próximo menu do sistema é o de importação de dados. Existem dois tipos de dados que devem ser importados diretamente da central telefônica, os usuários e as ligações. A Figura 27 apresenta a tela de importação de usuários.

Nesta tela o administrador do sistema deve localizar o arquivo .txt a ser importado e realizar o *upload* para o servidor, tornando a manipulação mas rápida. Existe um caixa de seleção que permite o administrador do sistema optar se deseja substituir os dados que já estiverem no sistema ou ignorar os dados que forem repetidos. Para evitar a importação de maneira inadequada o administrador pode pré-visualizar os dados a serem importados, por meio de uma listagem que informa quais dados serão substituídos e quais não serão alterados. Após a validação destes dados o administrador deve importar os dados para o sistema.

Figura 27 – Importação de usuários
Fonte: Autoria própria.

O funcionamento da importação de ligações é semelhante à importação de usuários como pode ser visualizado na Figura 28.

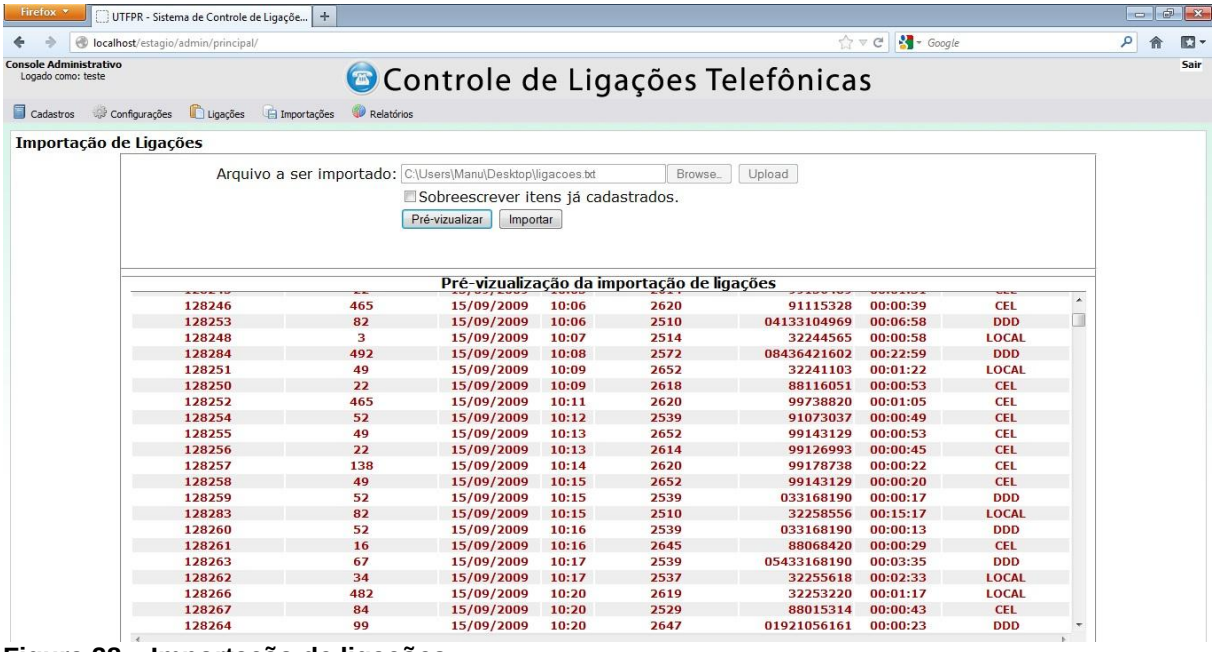

Figura 28 – Importação de ligações
Fonte: Autoria própria.

A próxima funcionalidade que pertence ao administrador do sistema é a avaliação das justificativas que os usuários derem para as suas ligações a serviço. Como pode ser visto na Figura 29 a primeira tela desta função apresenta uma relação com todos os usuários que possuem justificativas a ser avaliada.

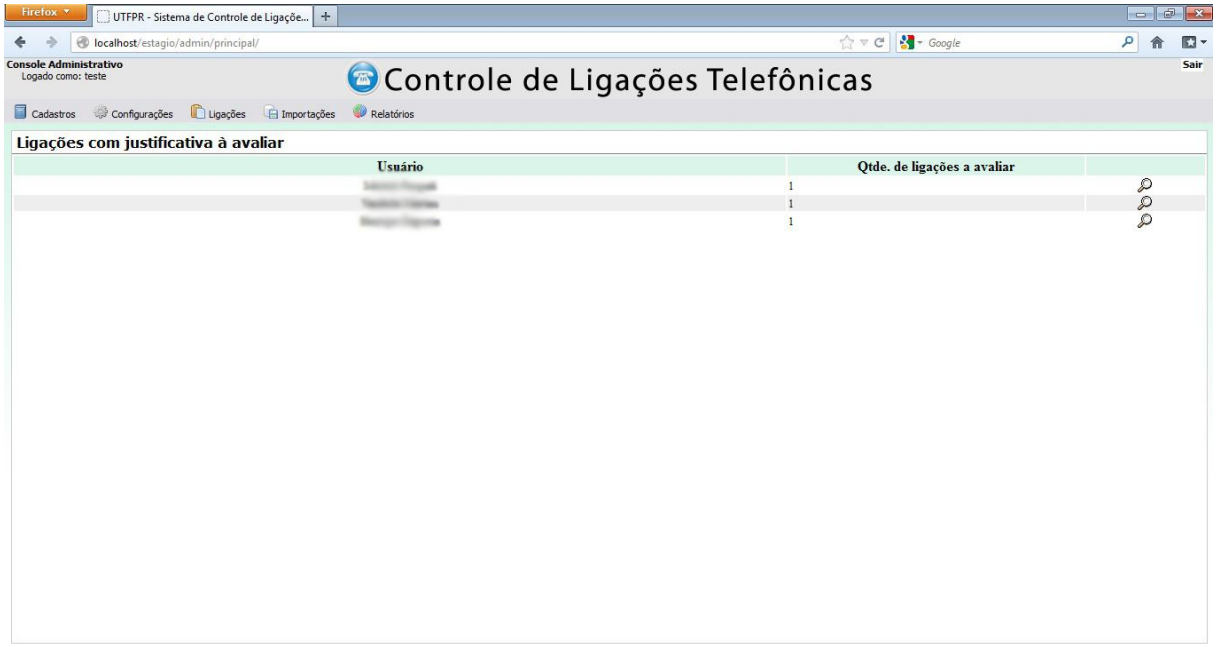

Figura 29 – Avaliação da justificativa de ligações
Fonte: Autoria própria.

Ao clicar a lupa de determinado usuário aparece uma tela com a relação de ligações aguardando a avaliação das justificativas. O administrador deve marcar se aceita ou não a justificativa e gravar a seleção. Essa tela pode ser visualizada na Figura 30.

Figura 30 – Avaliação da justificativa de ligações
Fonte: Autoria própria.

A última funcionalidade que o administrador do sistema possui é uma relação com todos os usuários que já tiveram suas ligações avaliadas, porém possui algum debito com a instituição. A Figura 31 apresenta esta tela.

Figura 31 – Listagem de usuários com débitos
Fonte: Autoria própria.

Ao clicar na lupa uma tela com a relação de ligações do usuário é apresentada, juntamente com o total devido pelo mesmo. Em caso de recebimento dos débitos o administrador deve clicar em Pagar Ligações, com isso o usuário sairá da lista de ligações devidas.

O módulo do usuário possui uma tela de *login* semelhante a do módulo administrativo que pode ser vista na Figura 32. O *login* do usuário será um código que é fornecido pelo administrador do sistema e a senha é a mesma utilizada para efetuar ligações na central telefônica.

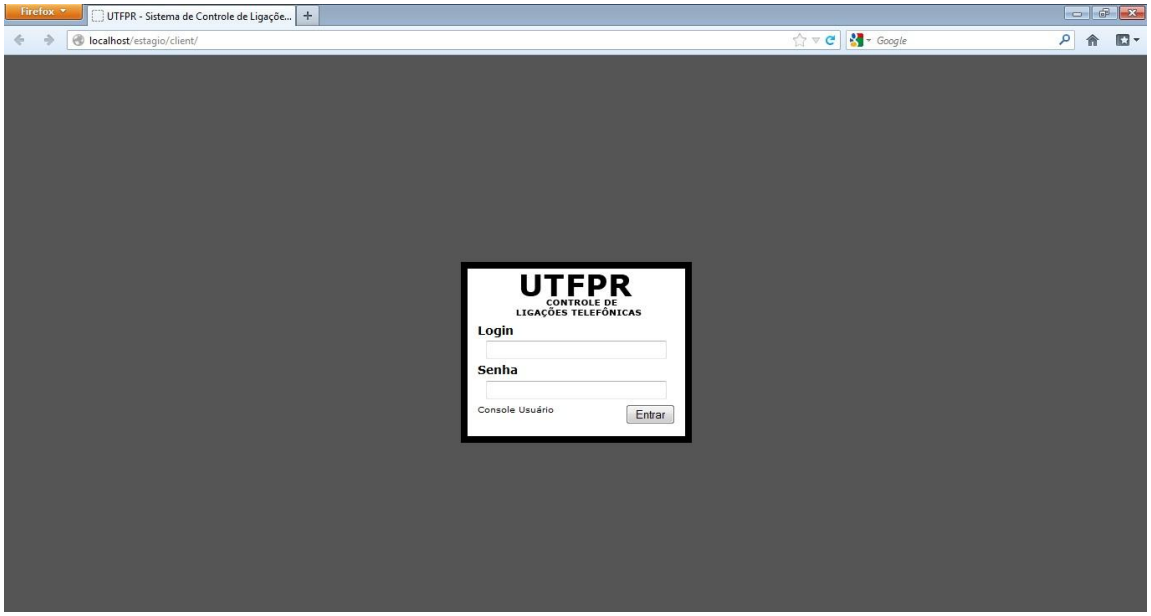

Figura 32 – Tela de login módulo usuário
Fonte: Autoria própria.

A primeira funcionalidade que o módulo do usuário possui é a de justificar as ligações. Na Figura 33 é possível visualizar que nesta tela existe uma listagem das ligações para os usuários identificar se foram à serviço ou particular.

Figura 33 – Justificativas de ligações dos usuários
Fonte: Autoria própria.

A segunda funcionalidade que o usuário possui é a agenda telefônica. Com ela o usuário pode identificar a que pertence determinado número de telefone, e criar uma identificação padrão para o determinado número. Essas telas são apresentadas nas Figuras 34 e 35.

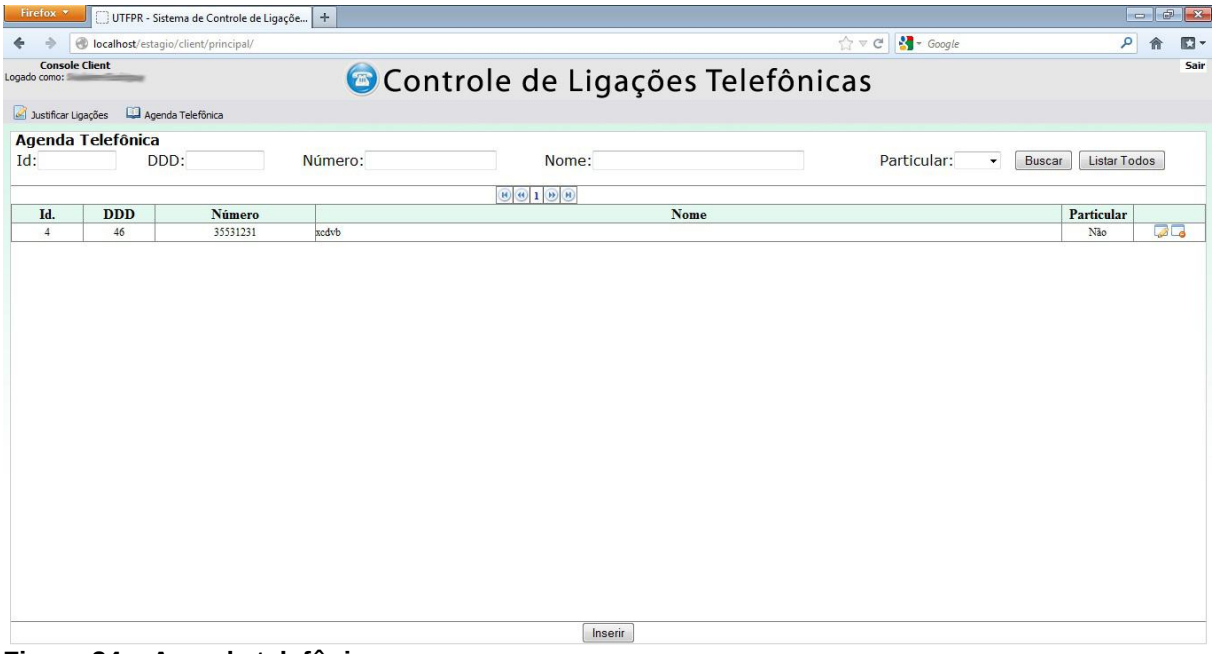

Figura 34 – Agenda telefônica.
Fonte: Autoria própria.

Figura 35 – Inserir agenda telefônica.
Fonte: Autoria própria.

4.10 Implementação do Sistema

Nesta seção é apresentado como o sistema foi implementado. O sistema foi codificado na linguagem PHP, juntamente com a tecnologia JQuery que torna o sistema mais dinâmico. A ferramenta utilizada para a codificação foi o Adobe Dreamweaver.

Inicialmente o projeto foi dividido em módulos distintos, sendo um para o administrador do sistema e o outro para os usuários. No mesmo nível hierárquico existem as pastas que podem ser utilizadas pelos dois módulos, sendo elas: arquivos, imagens, classes e inc. Esta estrutura pode ser visualizada na Figura 36.

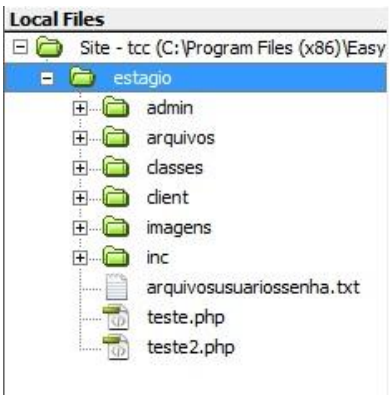

Figura 36 – Estrutura de pastas
Fonte: Autoria própria.

A linguagem PHP possui comunicação nativa com o banco de dados MySQL. Para fazer a conexão foi criado um arquivo que é incluído nas páginas que possuem comunicação com o banco de dados. Essa conexão e inclusão podem ser visualizadas nas Figuras 37 e 38.

base.php

Code | Split | Design | Title: | Check Page

```php
<?php
    mysql_connect("localhost:3306",  "root", "") or die("ERRO:".mysql_error());
    mysql_select_db("estagio");
?>
```

Figura 37 – Comunicação com o banco de dados
Fonte: Autoria própria.

inserir_centro_custo.php

Code | Split | Design | Title: Cadastro de Usuários

```php
<?php
    require_once("../../../inc/base.php");
    require_once("../../../inc/config.php");
```

Figura 38 – Inclusão de arquivo do banco de dados
Fonte: Autoria própria.

O sistema foi desenvolvido com a estrutura de orientação a objetos, para isso foram criadas classes que possuem todas as funções que possuem comunicação com o banco de dados. Quando é necessário fazer este contato deve-se criar um objeto da classe necessária. As Figuras 39 e 40 exemplificam uma classe e o seu processo de invocação.

Ligacoes.php

Code | Split | Design | Title: | Check Page

```php
<?php
class Ligacoes{
    private $id_ligacao;
    private $data_ligacao;
    private $hora_ligacao;
    private $numero_ligacao;
    private $ramal_origem;
    private $valor;
    private $taxa;
    private $id_centro_custo;
    private $duracao;
    private $id_tipo_servico;
    private $id_usuario;
    private $particular;
    private $justif_particular;
    private $aceita_justif;

    public function inserirLigacao(){
        $sql = "INSERT INTO tab_ligacoes (id_ligacao, data_ligacao, hora_ligacao, numero_ligacao, ramal_origem, valor, taxa,
        id_centro_custo, duracao, id_tipo_servico, id_usuario, particular, justif_particular)
        VALUES ('".$this->id_ligacao."', '".$this->data_ligacao."', '".$this->hora_ligacao."', '".$this->numero_ligacao."',
        '".$this->ramal_origem."', '".$this->valor."', '".$this->taxa."', '".$this->id_centro_custo."','".$this->duracao."', '".$this->id_tipo_se
        '".$this->id_usuario."', '".$this->particular."', '".$this->justif_particular."');";
        mysql_query($sql) or die(mysql_error());
    }

    public function alterarLigacao(){
        $sql = "UPDATE tab_ligacoes SET data_ligacao = '".$this->data_ligacao."', hora_ligacao = '".$this->hora_ligacao."',
        numero_ligacao = '".$this->numero_ligacao."', ramal_origem = '".$this->ramal_origem."', valor = '".$this->valor."',
        taxa = '".$this->taxa."', id_centro_custo = '".$this->id_centro_custo."', duracao = '".$this->duracao."', id_tipo_servico = '".$this->id_
        id_usuario = '".$this->id_ususario."', particular = '".$this->particular."', justif_particular = '".$this->justif_particular."'
```

Figura 39 – Codificação classe
Fonte: Autoria própria.

```php
require_once("../../../classes/Usuarios.php");
$usuario = new Usuarios();
if($_GET['act'] == 'gravar'){
    $usuario->setId_usuario($_POST['id_usuario']);
    $usuario->setNome_usuario($_POST['nome_usuario']);
    $usuario->setSenha($_POST['senha']);
    $usuario->alterarUsuario();
```

Figura 40 - Invocação de classe
Fonte: Autoria própria.

Para dinamizar o sistema foi utilizado o framework JQuery, que permite a invocação de funções sem a necessidade de recarregar a página inteira. Na Figura 41 é apresentado a codificação do framework JQuery .

```html
<link rel="stylesheet" type="text/css" href="../../inc/jqueryui/themes/cupertino/jquery-ui-1.7.1.custom.css">
<script src="../../inc/jquery/jquery-1.3.2.js" type="text/javascript" language="javascript"></script>
<script src="../../inc/jqueryui/ui/ui.datepicker.js" type="text/javascript" language="javascript"></script>
<script src="../../inc/script.js" type="text/javascript" language="javascript"></script>
<script>
    $(document).ready(function() {
        $("#data_ini_lig, #data_fim_lig, #data_ini_justif, #data_fim_justif").datepicker({
        dateFormat: 'dd/mm/yy',
        dayNames: [
            'Domingo','Segunda','Terça','Quarta','Quinta','Sexta','Sábado','Domingo'
        ],
        dayNamesMin: [
            'D','S','T','Q','Q','S','S','D'
        ],
        dayNamesShort: [
            'Dom','Seg','Ter','Qua','Qui','Sex','Sáb','Dom'
        ],
        monthNames: [
            'Janeiro','Fevereiro','Março','Abril','Maio','Junho','Julho','Agosto','Setembro',
            'Outubro','Novembro','Dezembro'
        ],
        monthNamesShort: [
            'Jan','Fev','Mar','Abr','Mai','Jun','Jul','Ago','Set',
            'Out','Nov','Dez'
        ],
        nextText: 'Próximo',
        prevText: 'Anterior'
        });
    });
```

Figura 41 – Codificação JQuery
Fonte: Autoria própria.

A Figura 42 apresenta um calendário desenvolvido com o framework JQuery.

Figura 42 – Calendário em JQuery
Fonte: Autoria própria.

4.11 Testes do Sistema

Nesta seção são apresentados os testes realizados durante o processo de desenvolvimento do sistema. Inicialmente foram realizados testes informais pelo próprio desenvolver.

Nesses testes foram verificados se todas as telas estão sendo exibidas de maneira adequada, independente de navegador e resolução de tela. Também foi verificado se todos os botões estavam acionando as funções adequadas. Após foi verificado se as validações de obrigatoriedade e formato de campos estavam funcionando de maneira correta.

A cada etapa desenvolvida o sistema era apresentado para o orientador do trabalho, Prof. Dr. Fabio Favarim, e se detectado algum erro, as correções foram realizadas.

4.12 Implantação do Sistema

Esta seção descreve como o sistema foi implantado na UTFPR – Câmpus Pato Branco.

A Coordenadoria de Tecnologia de Informação disponibilizou um servidor virtual, executando o sistema operacional Debian, para que fosse acessado por remotamente por meio do protocolo SSH (*Secure Shell*). Inicialmente foi instalado o servidor PHP que tem como função a interpretação dos códigos PHP dentro das páginas WEB. Após foi instalado o servidor WEB Apache e o banco de dados Mysql.

Foi criado o banco de dados com o nome telefone com usuários *root* e senha *root*, e importada as tabelas a partir de um arquivo .sql. Todos os arquivos do sistema foram copiados para a pasta www do servidor Apache.

Por fim foi instruído o administrador do sistema a como configurá-lo, importar os dados e utilizar o sistema.

5 CONSIDERAÇÕES FINAIS

O objetivo deste trabalho foi o desenvolvimento de um sistema para controle de gastos com ligações realizadas na UTFPR – Câmpus Pato Branco.

O desenvolvimento desse sistema surgiu da demanda da universidade em se ter um melhor controle de gastos com ligações telefônicas, informatizando o processo.

Para entender o sistema a ser modelado, atividade realizada como estágio acadêmico do autor deste trabalho, foram realizadas reuniões com o orientador deste trabalho e também com o servidor responsável pelo setor que faz a gestão das ligações telefônicas. A partir de cada reunião os requisitos do sistema eram documentados e após a documentação nova reunião era marcada para ratificar os requisitos previamente acordados e verificar se havia novas necessidades. Visando facilitar a compreensão do sistema, foi desenvolvido, a cada conjunto de requisitos o Diagrama dos Casos de Uso e a descrição detalhada de cada um.

A implementação do sistema foi realizada com a linguagem PHP juntamente com o framework JQuery, que permite uma maior dinamicidade aos sistemas.

Após a implementação, o sistema será implantado na UTFPR, Câmpus Pato Branco, de modo que o resultado de todo trabalho realizado possa efetivamente trazer benefícios à universidade e seus servidores, agilizando o processo de controle de ligações realizadas.

Espera-se com a realização deste trabalho, por meio de um software desenvolvido, ter contribuído com a Universidade, retornando um pouco do conhecimento adquirido na instituição.

REFERÊNCIAS BIBLIGRAFICAS

ADOBE. **Dreamweaver CS3**. Disponível em: http://www.adobe.com/products/dreamweaver.html. Acesso em 29 de abr. de 2013.

ALEXANDRINO, Olavo. **Polimorfismo e PHP 5**. Disponível em: http://phpbrasil.com/artigo/F0gMv5itYAD_/3/polimorfismo-e-php-5. Acesso em: 12 mar. 2013.

ANICETO, Jefferson. **Aplicações Web. Apostila ASP.net**. Escola Técnica da Univale (ETEIT), 2009.

BEZERRA, Eduardo. **Princípios de analise e projeto de sistemas com UML**. Rio de Janeiro: Elsevier, 2003.

CASTELA, Rodrigo T. Introdução à linguagem PHP. Disponivel em: http://www.dotsharp.com.br/programacao/php/introducao-a-linguagem-php.html. Acesso em 29 de abr de 2013.

CORREIA, Carlos Henrique; TAFNER, Malcon Anderson. **Análise orientada a objetos**. Florianópolis: Visual Books, 2006.

COSTA, Carlos J. **Desenvolvimento para Web**. Lisboa: Lusocredito, 2007.

Easy-PHP. **Servidor WAMP para desenvolvimento PHP e hospedagem**. Disponível em: http://www.easyphp.org/. Acesso em 29 de abr. de 2013.

FABFORCE. **DBDesigner 4.** Disponível em http://fabforce.net/dbdesigner4/index.php. Acesso em 16 de fev. de 2013.

FERRANTE, Augustin Juan; RODRIGUEZ, Martius Vinicius Rodriguez y. **Tecnologia de informação e gestão empresarial**. Rio de Janeiro: E-papers, 2000.

LAUDON, Kenneth C; LAUDON, Jane. **Management information system**, 4ª edição, Nova Jersey: Prentice Hall,1998.

LEITE, Mario. **Técnicas de programação - Uma abordagem moderna**. Rio de Janeiro: Brasport, 2006.

IBM. **Regras de negócio.** Disponível em: http://www-01.ibm.com/software/awdtools/rmc/library/ . Acesso em: 22 fev. 2013.

MYSQL. **Sistema de gerenciamento de banco de dados MySql.** Disponível em: http://www.mysql.com/. Acesso em: 29 abr. 2013.

MySQL-Front. **Software de administração de banco de dados Mysql.** Disponível em: http://www.mysqlfront.de/. Acesso em: 29 abr. 2013.

RUMBAUGH, James; BLAHA, Michael. **Modelagem e projetos baseados em objetos com UML 2**. Rio de Janeiro: Campus, 2006.

SERSON, Roberto Rubinstein. **A Bíblia da certificação Java 6.** 1. ed. Rio de Janeiro: Brasport, 2009.

STARUML. **StarUML** Disponível em: http://staruml.sourceforge.net/en/. Acesso em: 18 fev. 2013.

SUH, Woojong. **Web engineering principles and techniques**. Pensilvânia: IGI Publishing, 2004.