FACULDADES INTEGRADAS DE CARATINGA

FACULDADE DE CIÊNCIA DA COMPUTAÇÃO

PESQUISA E DESENOLVIMENTO DE SOFTWARE APLICATIVO MULTIPLATAFORMA PARA AUTOMAÇÃO DO CONTROLE DE ORDENS DE SERVIÇO

ABÍLIO MARCONNI VIEIRA LEAL DE SÁ

FIC/CARATINGA

2011

Abílio Marconni Vieira Leal de Sá

PESQUISA E DESENVOLVIMENTO DE SOFTWARE APLICATIVO MULTIPLATAFORMA PARA AUTOMAÇÃO DO CONTROLE DE ORDENS DE SERVIÇO

Monografia apresentada à Faculdade de Ciência da Computação das Faculdades Integradas de Caratinga como exigência parcial da disciplina de Trabalho de Conclusão de Curso II, sob orientação do professora Msc. Fabrícia Pires Souza Tiola

FIC/CARATINGA

2011

Abílio Marconni Vieira Leal de Sá

PESQUISA E DESENVOLVIMENTO DE SOFTWARE APLICATIVO MULTIPLATAFORMA PARA AUTOMAÇÃO DO CONTROLE DE ORDENS DE SERVIÇO

Monografia apresentada à Faculdade de Ciência da Computação das Faculdades Integradas de Caratinga como exigência parcial da disciplina de Trabalho de Conclusão de Curso II, sob orientação do professora Msc. Fabrícia Pires Souza Tiola

_____
Prof. Msc. Fabrícia Pires Souza Tiola
Faculdades Integradas de Caratinga

_____
Prof.
Faculdades Integradas de Caratinga

_____
Prof.
Faculdades Integradas de Caratinga

FIC/CARATINGA

2011

AGRADECIMENTOS

Agradeço aos meus pais, pelo apoio incondicional, a minha esposa, por ter compreendido alguns momentos de ausência em função deste trabalho, e a Deus, por ter me dado forças para concluir este trabalho.

Agradeço especialmente a minha orientadora, Msc. Fabrícia Pires Souza Tiola, por ter tido enorme paciência e por ter sido tão prestativa.

RESUMO

Atualmente o mercado não oferece muitos software gratuitos para Automação Comercial. Quando se trata de Automação de controle de ordens de serviço em empresas que realizam serviços de manutenção, pode-se dizer que os softwares existentes neste contexto não são satisfatórios. Estes softwares, em sua maioria, apresentam os seguintes problemas, constatados na presente pesquisa: dependência de plataforma específica, no caso, o SO Microsoft Windows, impossibilidade de facilitar a comunicação entre o contratante e o contratado e impossibilidade de acesso a partir de dispositivos móveis (celulares, smartphones, tablets, etc). O presente trabalho tem por finalidade identificar estas carências existentes na automação de controle de ordens de serviço, desenvolver um software aplicativo, que além de servir para este fim, possa ser ampliado para outras finalidades.

O resultado do presente trabalho é um software escrito na linguagem de programação PHP, utilizando o paradigma de programação orientada a objetos, utilizando o padrão de arquitetura de software Model-View-Controller, utilizando o PostgreSQL como sistema gerenciador de bancos de dados.

O teste do software aqui apresentado na empresa de manutenção de equipamentos Tecnocéu Informática leva a concluir que o uso deste software em pequenas empresas de manutenção de equipamentos é viável.

Palavras-chaves: Software Aplicativo, Automação, Gestão Empresarial, Ordem de serviço, PostgreSQL, Model-View-Controller, PHP, Programação orientada a objetos.

ABSTRACT

Actually the market doesn't offer many free software for business management. When it comes to automation of service orders, there are not many softwares for this purpose. Most of these software has the following problems, found in this research: Dependence of the Microsoft Windows platform, inability to facilitate communication between the costumer and the enterprise and inability of being accessible from mobile devices (like cell phones, smart phones and tablets). This research aims identify these deficiencies in the software for automation of service orders and build a software, which not only serve this purpose, but also for may be extended for others goals.

The result of this research is a software written in PHP programming language, using the object-oriented programming paradigm, using the software architecture Model-View-Controller, using PostgreSQL as database management databases.

Testing of the software here presented in the company of equipment maintaining Tecnocéu Informática leads to the conclusion that the use of this software in equipment maintenance small company is feasible.

Keywords: Application software, Business Automation, Service Orders, PostgreSQL, Model-View-Controller, PHP, Object oriented programming.

Lista de tabelas

Lista de Figuras

Lista de Siglas

ANSI	American National Standards Institute – Instituto Nacional Americano de Padrões
API	Application Programming Interface – Interface de Programação de Aplicativos
CEP	Código de Endereçamento Postal
CNPJ	Cadastro Nacional de Pessoa Jurídica
CPF	Cadastro de Pessoa Física
HTML	Hypertext Markup Language – Linguagem de Marcação de Texto
ISO	International Standards Organization – Organização Internacional de Padrões
MVC	Model-View-Controller – Modelo Visão Controlador
PHP	PHP Hypertext Processor – Processador de Hipertexto PHP
POO	Programação orientada a objetos
RFID	Radio Frequency Identification – Identificação por radiofrequência
SGBD	Sistema gerenciador de bancos de dados
SMS	Serviço de Mensagens Curtas
SO	Sistema Operacional
SQL	Structured Query Language – Linguagem de consulta estruturada
TI	Tecnologia da Informação
VB	Visual Basic
VC++	Visual C++
WWW	World Wide Web – Rede Mundial de Computadores
W3C	World Wide Web Consortium – Consórcio da Rede Mundial de Computadores

Sumário

1. INTRODUÇÃO

A necessidade por softwares de automação dentro das mais diversas empresas é um fato. Estes softwares podem ser úteis de diversas maneiras dentro das empresas. Por exemplo, os softwares podem facilitar a gestão financeira da empresa, permitindo a esta obter de forma rápida informações relevantes a respeito de sua saúde financeira de modo que esta possa tomar decisões a este respeito. Os softwares podem também ajudar as empresas a obter informações a respeito do perfil de compra de determinado cliente, permitindo determinar qual tipo de produto este cliente costuma comprar, e assim, possibilitar a empresa oferecer este tipo de produto ao cliente.

Mas existem empresas que possuem demandas específicas de software, como é o caso das empresas que realizam manutenção de equipamentos. E a manutenção de equipamentos é um tipo de serviço muito importante nos dias de hoje. Segundo Dhillon (2002, p. 13), “a cada ano bilhões de dólares são gastos com manutenção de equipamentos em todo o mundo”.

Dada a grande importância deste tipo de serviço (e das empresas que o realizam), é de se imaginar que existam bons softwares que auxiliem na automação deste tipo de empresas, ou seja, softwares para controle de ordens de serviços. No entanto, a presente pesquisa constatou, que em pequenas empresas de manutenção de equipamentos, os softwares atualmente disponíveis para automação do controle de ordens de serviços, em sua maioria, apresentam as seguintes deficiências: são dependentes da plataforma Windows, da Microsoft, o que impede sua utilização em plataformas livres e gera custo de aquisição de licenças para a empresa usuária; impossibilidade de acesso a partir de dispositivos móveis; não oferecem uma interface a partir da qual o contratante possa acompanhar a ordem de serviço por ele requisitada.

Estas deficiências apresentadas por estes softwares são decorrentes de demandas surgidas recentemente no mercado de softwares do Brasil. A necessidade de softwares multiplataforma, por exemplo, não existia há poucos anos atrás, quando o Microsoft Windows era o sistema operacional mais utilizado entre os usuários, sendo extremamente raro encontrar usuários comuns utilizando outros sistema operacionais. Porém sistemas como o Linux vem ganhando espaço entre os usuários nos últimos anos, surgindo assim a necessidade de softwares que sejam independentes de plataforma.

O crescimento do uso da internet também é um fato a ser considerado neste contexto. Este crescimento trouxe consigo a tendência de desenvolver softwares que pudessem ser acessados a partir de navegadores. Isso permite que estes softwares sejam totalmente independentes de plataforma (teoricamente, qualquer dispositivo com a capacidade de acessar páginas web poderiam

acessar o software desenvolvido). Esta prática tem também a vantagem de eliminar a necessidade de instalar o software em cada dispositivo em que se pretende acessá-lo.

Levando-se em conta estas considerações, a presente pesquisa teve por objetivo geral otimizar a gestão de ordens de serviço em empresas de pequeno porte que prestam serviços de manutenção. Os objetivos específicos desta pesquisa fora: definir critérios que permitissem analisar os softwares existentes de forma crítica ;identificar, a partir da análise, recursos que deveriam e não são oferecidos pelos softwares; a partir disso, produzir um software que atenda pequenas empresas de manutenção oferecendo os recursos que ainda não são oferecidos na gestão de ordens de serviço.

Primeiramente, foi necessário definir critérios que permitissem selecionar os softwares que serviriam para análise. Depois, foi necessário definir critérios que permitissem analisar os softwares, isso é, mensurar o quanto os softwares selecionados podiam ser úteis no contexto da automação do controle de ordens de serviço.

O software foi escrito na linguagem de programação PHP, o que tornou simples a geração de páginas web que permitem ao software ser acessado a partir de navegadores de internet. O software seguiu o padrão de arquitetura MVC (Model, View, Controller), o que permitiu uma maior organização do código fonte. Foi utilizado como SGBD (sistema gerenciador de bancos de dados) o PostgreSQL porque este é, segundo Greschwinde e Schönig (2002, p. 7), “o banco de dados de código aberto mais avançado do mundo”.

Utilizou-se como paradigma a programação orientada a objetos que, dentro do contexto do presente trabalho, permitiu que o código escrito tivesse grande legibilidade, aproximação com o mundo real a partir da representação de classes e objetos, e fácil extensibilidade das funções já providas pelo software.

No desenvolvimento das telas, procurou-se manter grande semelhança com programas desktop, com os quais os gestores de empresas estão acostumados, fazendo-se uso de botões, caixas de texto, tabelas, menus, caixas de diálogo. Foi feito o teste do software na empresa de manutenção de equipamentos Tecnocéu Informática, o que permitiu avaliar a viabilidade do uso deste software em empresas deste tipo.

O presente trabalho está dividido nos seguintes capítulos: Referencial Teórico, onde é feita a revisão dos conceitos necessários para a compreensão do trabalho; Metodologia, onde é feito o estudo sobre os métodos utilizados para a pesquisa e para o desenvolvimento do software; Resultados, onde é apresentado o software desenvolvido neste trabalho; Conclusão, onde é apresentado a conclusão baseada nos testes realizados; e Trabalhos Futuros, capítulo no qual são apresentados algumas propostas de trabalhos que poderiam vir a complementar o trabalho já realizado.

2. REFERENCIAL TEÓRICO

Segundo a empresa SAP (2011), uma ordem de serviço é um “Contrato a curto prazo entre o fornecedor de serviços e o contratante de serviços, em que serviços únicos são especificados em uma ordem e para o qual o faturamento relacionado ao recurso é executado na conclusão. Ela é uma solicitação de uma atividade de serviços a ser executada em um objeto de manutenção, em uma sociedade do cliente em uma determinada data.”

Segundo Dhillon (2002, p. 13), manutenção é um trabalho que deve ser realizado sob circunstâncias normalmente adversas, e seu principal objetivo é restaurar rapidamente o equipamento ao seu funcionamento utilizando os recursos disponíveis.

Estes dois conceitos se relacionam, pois uma ordem de serviço é um contrato relacionado a manutenção de determinado equipamento.

Ainda seguindo Dhillon (2002, p. 13), “a cada ano bilhões de dólares são gastos com manutenção de equipamentos em todo o mundo”.

Dada a grande importância que tem esta atividade, faz-se necessária a existência de softwares que auxiliem as empresas que prestam este tipo de serviço. Estes softwares, no Brasil, são normalmente apresentados como softwares para automação de ordens de serviço. Nos tópicos a seguir, serão apresentados os conceitos necessários para a compreensão do presente trabalho.

2.1. Software

Segundo Medina e Fertig (2005, p. 1), um algoritmo é “um procedimento passo a passo para a solução de um problema”, e também “uma sequência detalhada de ações a serem executadas para realizar alguma tarefa”. Ainda segundo Medina e Fertig (2005, p. 6), um programa (ou software de computador) “nada mais é que um tipo de algoritmo. Sua particularidade é que suas operações são específicas para o computador e restritas ao conjunto de instruções que o processador pode executar.”.

Os softwares existem com a finalidade de auxiliar as pessoas nas mais diversas atividades que envolvam processamento de informações (Silveira, 2004, p. 1). Desse modo, os softwares são ferramentas necessárias a sociedade atualmente, de modo que pessoas e empresas se dispõem até mesmo a pagar pelos softwares de que precisam (Silveira, 2004, p. 1). Assim, existem empresas especializadas no desenvolvimento de softwares, como a Microsoft (Silveira, 2004, p. 1). Essas

empresas, normalmente disponibilizam seus softwares sob determinadas licenças, de modo que o usuário nunca é dono do software, mas sim tem o direito de usá-lo segundo os termos da licença pertinente ao software (Silveira, 2004, p. 1).

Por outro lado, existem os softwares livres, que serão explicados no tópico 2.1.1.

2.1.1. Software Livre

Em oposição aos softwares proprietários, existem os softwares livres, que são submetidos a licenças que garantem aos seus usuários os diretos de uso, cópia, modificações em seu código e redistribuição do software (Silveira, 2004, p. 1).

Uma das grandes vantagens oferecidas pelos softwares livres é o fato de que qualquer um que possua conhecimento de programação de computadores possa modificar o software, adaptando-o as necessidades de determinada comunidade, ou melhorando-o. Desse modo, projetos de software livre (como o sistema operacional GNU/Linux), costumam ser suportados por uma ampla comunidade de desenvolvedores de software que fazem contribuições para o código do software. Isso contribui significativamente para a velocidade em que erros do software possam ser corrigidos e novas funcionalidades possam ser implementadas, conforme se verifica na citação a seguir:

> "*A diferença fundamental de desenvolvimento entre o software livre e o proprietário fica mais evidente ao se observar o modelo de desenho e confecção dos programas. As empresas de software proprietário trabalham somente com programadores contratados, assalariados ou terceirizados. Todo o desenvolvimento do software é interno a empresa. Já o modelo de código aberto é o modelo colaborativo que envolve programadores da empresa e todos aqueles interessados no desenvolvimento daquele software, inclusivo voluntários espalhados pelo mundo. Por isso grande parte dos softwares livres possui sites na web para atrair desenvolvedores que trabalham coordenadamente pela rede mundial de computadores*". (SILVEIRA, 2004, p. 1)

Segundo Silveira (2004, p. 1), os maiores opositores do projeto de softwares livres são empresas de desenvolvimento de software que utilizam um modelo econômico baseado na exploração de licenças de uso de software, e do controle centralizado dos códigos essenciais de seus softwares. Essas empresas costumam alegar que o software livre não representa um modelo econômico rentável, visto que os usuários tem acesso gratuito ao software.

Em contrapartida, existem empresas que seguem um modelo econômico baseado em softwares livres, oferecendo as empresas que utilizam seus softwares serviços de consultoria e suporte técnico, comprovando assim que os softwares livres podem representar também um modelo econômico rentável.

2.1.2. Desenvolvimento de software

O desenvolvimento de softwares, segundo Birrel (1985, p. 3), é o ato de elaborar e implementar um sistema computacional, isto é, transformar a necessidade de um utilizador ou de um mercado em um produto de software.

Para que possa ser desenvolvido, um software deve passar por um processo que envolve diversos estágios, tais como Análise de requisitos de software, implementação e documentação. Todo software é construído, segundo Deitel, P. e Deitel, H. (2010, p. 5), utilizando-se linguagens de programação. O conceito de linguagem de programação será apresentado no tópico 2.1.3.

2.1.3. Linguagens de programação de software

Segundo Deitel, P. e Deitel, H. (2010, p. 5), "Os programadores escrevem instruções em várias linguagens de programação, algumas diretamente compreensíveis por computadores e, outras, requerendo passos intermediários de tradução."

Existem, segundo Kedar (2007, p. 3), basicamente quatro tipos de linguagens de programação:

- Linguagens de máquina;
- Linguagens assembly;
- Linguagens de alto nível;
- Linguagens 4GL.

As linguagens de máquina são, segundo Deitel, P. e Deitel, H. (2010, p. 5), as linguagens que o computador pode entender, que são definidas pelo seu design de hardware. Essas linguagens consistem geralmente em strings de números (em última instância reduzidas a 1s e 0s) que instruem os computadores a realizar suas operações mais elementares uma de cada vez.

As linguagens de máquina geralmente não são utilizadas diretamente para a confecção dos softwares, pois são dependentes de máquina (isto é, cada computador pode entender diretamente apenas sua própria linguagem de máquina) (DEITEL, P. e DEITEL, H. 2010, p. 5). Além disso, a programação na linguagem de máquina é um processo muito lento e tedioso para a maioria dos programadores (DEITEL, P. e DEITEL, H. 2010, p. 5).

As linguagens assembly são constituídas por abreviações (mnemônicos) em inglês das operações básicas que os computadores podem realizar, isto é, das instruções que podem ser dadas nas linguagens de máquina (DEITEL, P. e DEITEL, H. 2010, p. 5).

Como os computadores não entendem diretamente as linguagens assembly, é necessário o uso de

programas assemblers que convertem os programas de linguagem assembly em linguagem de máquina. (DEITEL, P.; DEITEL, H. 2010, p. 5).

Embora as linguagens assembly sejam mais fáceis de entender para seres humanos do que as linguagens de máquina, elas representam apenas traduções das instruções das linguagens de máquina, requirindo portanto que muitos comandos sejam dados para que o programa realize determinada operação prática. Por este motivo, as linguagens mais utilizadas para a produção de software são as linguagens de alto nível.

> "*As linguagens de alto nível são compostas por instruções que podem ser escritas para realizar tarefas substâncias, acelerando o processo de programação. Os programas tradutores chamados compiladores convertem os programas de linguagem de alto nível em linguagem de máquina. Linguagens de alto nível permitem aos programadores escrever instruções que se parecem com o inglês cotidiano e contêm notações matemáticas comumente utilizadas.*" (DEITEL, P.; DEITEL, H. 2010, p. 6)

Segundo Kedar (2007, p. 4), as linguagens de alto nível substituíram as linguagens de máquina e as linguagens assembly em todas as áreas da programação, porque elas apresentam os seguintes benefícios:

- Notações legíveis
- Independência de máquina
- Possibilidade de uso de livrarias (APIs, Application Programming Interface)
- Checagem de consistência de código durante a implementação, que permite a identificação de erros

Existem diversas linguagens de programação de alto nível. Cada linguagem é desenvolvida para atender a determinado paradigma de programação (este conceito será explicado no tópico 2.1.6), ou para construir com maior facilidade determinados tipos de programas. Para o presente trabalho, a linguagem de alto nível escolhida para o desenvolvimento do software foi o PHP, pelos motivos apresentados no tópico 2.1.4.

As linguagens 4GL, ou linguagens de quarta geração, são linguagens que tem profunda aproximação com as linguagens humanas. São linguagens como Oracle, VB (Visual Basic), VC++ (Visual C++), SQL (Structured Query Language), etc. (KEDAR, 2007, p. 4). Dessas, a linguagem SQL foi utilizada no presente trabalho para acessar o banco de dados PostgreSQL. A linguagem SQL será explicada no tópico 2.1.5, e o sistema gerenciador de bancos de dados PostgreSQL será explicado no tópico 2.1.4.

A maioria das linguagens 4GL são utilizadas para acessar bancos de dados, e elas permitem aos programadores definir o quê é necessário que o computador faça, sem que se precise especificar como implementar isso. (KEDAR, 2007, p. 4)

2.1.4. Linguagem de programação PHP

A escolha da linguagem de programação para este trabalho foi feita levando em conta o fato de a linguagem permitir, com facilidade, a geração de páginas web que possam ser acessadas por navegadores.

Segundo Lerdorf, Tatroe e Intyre (2006, p. 1), o PHP é uma linguagem, que originalmente, é utilizada para desenhar paginas para a web. Ainda segundo Lerdorf, Tatroe e Intyre (2006, p. 1), o PHP é livre e pode ser utilizado nos principais sistemas operacionais, desde as variantes do Unix incluindo o Linux, FreeBSD, até o Solaris, Windows e até o Mac OS X.

Conforme verifica-se, o PHP adapta-se muito bem aos propósitos deste trabalho, por ser independente de plataforma, ser desenvolvido para geração de páginas para World Wide Web, e permitir a utilização do paradigma de programação POO (Programação Orientada a Objetos) (LERDORF;TATROE e INTYRE; 2006, p. 143), que será descrito no tópico 2.1.6.4.

2.1.5. Linguagem de quarta geração SQL

Segundo Harrington (2003, p. 17), SQL (Structured Query Language, ou Linguagem de Consulta estruturada) é uma linguagem de manipulação de bancos de dados que tem sido implementada, virtualmente, em todos os sistemas gerenciadores de bancos de dados relacionais (DBMS ou SGBD).

Esta linguagem de quarta geração está atualmente padronizada pelo ANSI (American National Standards Institute, ou Instituto Americano Nacional de Padrões) e pelo ISO (International Standards Organization) como uma linguagem padrão de consulta para bancos de dados relacionais.

Segundo Kedar (2007, p. 4), a linguagem SQL, sendo uma linguagem de quarta geração, possui os seguintes recursos e vantagens:

- Facilidade de uso: Como a sintaxe das linguagens de quarta geração são mais próximas das linguagens humanas, elas são fáceis de aprender. Adicionalmente, devido a natureza não procedural da maioria destas linguagens, as de construções são também simples, de modo que os resultados acontecem de forma rápida.
- Número limitado de funções: As linguagens de quarta geração são tipicamente desenhadas para um conjunto limitado de funções. Isso representa mais um ponto que facilita o aprendizado destas linguagens.
- Disponibilidade de opções:As linguagens de alto nível sempre restringem as opções

disponíveis aos usuários, enquanto as linguagens de quarta geração provem diversas opões para o usuário, o que causa uma verificação automática do programa antes de testá-lo.

- Opções padrão:Um usuário de uma linguagem de quarta geração, com o SQL, não precisa especificar todos os parâmetros. Na verdade, o compilador ou interpretador é capaz de fazer deduções inteligentes a partir dos parâmetros especificados pelo usuário.

2.1.6. Paradigmas de programação de softwares

Segundo Kedar (2007, p. 24), os paradigmas de programação de software, basicamente, estão relacionados com os diversos tipos de linguagens de programação.

Existem 5 tipos de modelos computacionais que descrevem a maioria das linguagens de programação atualmente:

- Linguagens Imperativas ou Linguagens Procedurais;
- Linguagens Aplicativas ou Linguagens Funcionais;
- Linguagens Lógicas ou Linguagens Orientadas a Regras;
- Linguagens Orientadas a Objetos;
- Linguagens Simultâneas.

Segundo Kedar (2007, p. 24), as linguagens imperativas, ou linguagens procedurais, são orientadas a comandos. Um programa construído com este paradigma consiste em uma sequência de comandos, e a execução de cada comando causa a mudança de valores em um ou mais locais da memória do computador em que é executado esse programa.

Segundo Kedar (2007, p. 25), nas linguagens aplicativas, maior ênfase é dada a função, de modo que um programa construído com este paradigma representa um conjunto de funções (não sendo executado, portanto, comando por comando como nas linguagens orientadas a comandos).

Segundo Kedar (2007, p. 26), as linguagens orientadas a regras estabelecem um modelo de programação no qual o programa checa pela presença de determinada condição, e quando presente, executa determinada ação.

A linguagem de programação mais comum que segue este paradigma é o Prolog, que também é chamado de Programação Lógica.

O paradigma de programação orientado a objetos (POO) é, segundo Poo, Kiong e Ashok (2008, p. 1), um paradigma de programação de software baseado na composição e interação entre diversas unidades de software chamadas de objetos.

A programação orientada a objetos, dentro do contexto do presente trabalho, permite que o

código escrito tenha grande legibilidade, aproximação com o mundo real a partir da representação de classes e objetos, e fácil extensibilidade das funções já providas pelo software. E aliada ao padrão de arquitetura de software MVC (descrito no tópico 2.1.8), permite também dividir o software em módulos, facilitando assim a identificação e correção de problemas no código.

Num contexto geral, a programação orientada a objetos permite ainda que determinado problema possa ser decomposto em objetos, e então construir métodos ao redor destes objetos.

A programação orientada a objetos tem, segundo Kedar (2007, p. 27), os seguintes conceitos básicos, que são utilizados na construção das classes dos programas:

- Encapsulamento;
- Abstração de dados;
- Herança;
- Polimorfismo;
- Comunicação entre objetos através de mensagens;
- Extensibilidade.

Segundo Kedar (2007, p. 27), o conceito fundamental de programação simultânea é a noção de processo. Um processo corresponde a uma computação sequencial, com sua própria linha de controle.

A linguagem de programação Ada, foi desenvolvida com o propósito de permitir a construção de programas que funcionem de forma simultânea. Outro nome bastante comum para esta prática computacional é multi-threading.

2.1.7. Padrões de projeto de software

Segundo Shalloway e Trott (2002, p. 96), um Padrão de Projeto de Software, ou Design Pattern descreve uma solução geral para um problema repetidamente cometido no desenvolvimento de softwares.

Segundo Shalloway e Trott (2002, p 97), um Padrão de Projeto de Software, é constituído pelas seguintes características principais:

- Nome: Todos os padrões têm um nome único que os identifica.
- Intenção: O propósito do padrão.
- Problema: O problema que o padrão tenta resolver.
- Solução: Como o padrão provê uma solução para o problema no contexto em que ele aparece.

- Consequências: As consequências da solução adotada pelo padrão.

2.1.8. Padrão de projeto de software MVC (Model-View-Controller)

Segundo Miller, Vandome e McBrewster (2010, p. 1), Model-view-controller (MVC) é um padrão de arquitetura de software que visa a separar a lógica de negócio da lógica de apresentação, permitindo o desenvolvimento, teste e manutenção isolado de ambos.

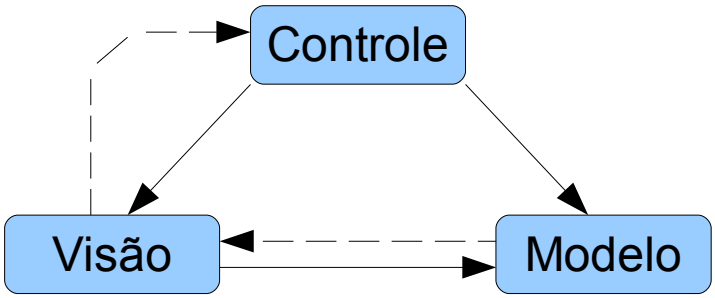


Figura 1: Diagrama exemplificando a relação entre Model, View e Controller.

No diagrama da Figura 1, verifica-se que as linhas sólidas representam associação direta, e as linhas pontilhadas indicam associação indireta.

2.2. Banco de Dados

Um banco de dados é, segundo Ramez e Navathe (2006, p. 10), “uma coleção de dados relacionados. Os dados são fatos que podem ser gravados e possuem um significado implícito.”

Segundo Ramez e Navathe (2006, p. 10), o conceito de banco de dados possui as seguintes propriedades implícitas:

- Um banco de dados representa alguns aspectos do mundo real, sendo chamado as vezes, de mini mundo ou de universo de discurso (UoD). As mudanças no mini mundo são refletidas em um banco de dados.
- Um banco de dados é uma coleção lógica e coerente de dados com algum significado inerente. Uma organização de dados ao acaso (randômica) não pode ser corretamente interpretada como um banco de dados.
- Um banco de dados é projetado, construído e povoado por dados, atendendo a uma proposta

específica. Possui um grupo de usuário definido e algumas aplicações preconcebidas, de acordo com o interesse desse grupo de usuários.

Em outras palavras, um banco de dados possui algumas fontes das quais os dados são derivados, alguns níveis de interação com os eventos do mundo real e um público efetivamente interessado em seus conteúdos (RAMEZ e NAVATHE, 2006, p. 10).

2.3. Sistema Gerenciador de Banco de Dados (SGBD)

Um sistema gerenciador de banco de dados (SGBD), é, segundo Ramez e Navathe (2006, p. 11), uma coleção de programas que permite aos usuários criar e manter um banco de dados. O SGDB é, portanto, um sistema de software de propósito geral que facilita os processos de definição, construção, manipulação e compartilhamento de banco de dados entre vários usuários e aplicações. A definição de um banco de dados implica especificar os tipos de dados, as estruturas e as restrições para os dados a serem armazenados em um banco de dados.

> "*Outras funções importantes do SGDB são a proteção e a manutenção do banco de dados por longos períodos. A proteção inclui a proteção do sistema contra o mau funcionamento ou falhas (crashes) no hardware ou software, e segurança contra acessos não autorizados ou maliciosos.*" (RAMEZ e NAVATHE, 2006, p. 11)

Segundo Douglas, K; e Douglas, S; (2003, p. 1) o PostgreSQL é um sistema gerenciador de bancos de dados relacional gratuito, que é hoje desenvolvido por um grupo internacional de desenvolvedores de software de código aberto conhecido como PostgreSQL Global Development Group (Grupo global de desenvolvimento do PostgreSQL).

O PostgreSQL foi escolhido para ser utilizado no software aqui apresentado, pois além de ser gratuito e possuir diversos recursos avançados (DOUGLAS, K; e DOUGLAS, S; 2003, p. 1 até 2), possui recursos avançados de gerenciamento de banco de dados, e tem amplo suporte no PHP.

Segundo a empresa DKAL (http://www.automacaocomerc ial.com.br/index.php), a Automação Comercial pode ser entendida como "um esforço para transformar tarefas manuais repetitivas em processos automáticos, realizados por uma máquina." E segundo a empresa SAP, uma ordem de serviço é um "Contrato a curto prazo entre o fornecedor de serviços e o contratante de serviços". E também "é uma solicitação de uma atividade de serviços a ser executada em um objeto de manutenção, em uma sociedade do cliente em uma determinada data".

Com base nesses conceitos, pode-se deduzir que o controle de ordens de serviço é pertinente ao campo da Automação Comercial, porque constitui uma forma de aplicar métodos para automatizar processos comerciais, que são neste contexto, os processos envolvidos na execução da ordem de

serviço. E percebe-se ainda, dentro dos conceitos, que um software para automação de controle de ordens de serviço constituiria uma ferramenta importante não somente para o fornecedor de serviços de manutenção, mas também para o contratante de serviços, visto que o software poderia permiti-lo acompanhar a execução do serviço contratado.

Todo software, deve ser desenvolvido em determinada linguagem de programação. O software desenvolvido no presente trabalho utiliza a linguagem de programação PHP, que é apresentada no tópico 2.3. É possível que a camada de apresentação do software seja escrita numa linguagem diferente da linguagem utilizada para desenvolver as outras camadas do software. Isto costuma ocorrer por mera conveniência. No software apresentado no presente trabalho, a camada de apresentação foi construída utilizando-se não uma linguagem de programação, mas sim a linguagem de marcação HTML (Hypertext Markup Language) e a linguagem de script Javascript.

Segundo BIGDOLI (2004, p. 124), "o W3C (World Wide Web Consortium), diz que o "HTML é a *língua franca* para a publicação de hipertexto na rede de computadores mundial"". Isto é, o HTML é uma linguagem de marcação para construção de páginas da Web. O uso de HTML aliado ao Javascript possibilita a construção de páginas Web com maiores recursos.

Segundo FLANAGAN (2002, p. 19), o Javascript "é uma linguagem de programação leve, interpretada, e com recursos de orientação a objetos".

Ainda segundo FLANAGAN (2002, p. 19), o Javascript é utilizado para controlar o comportamento da página no navegador. Isso possibilita a criação de páginas web que se assemelham a programas desktop, permitindo uma experiência mais amigável com o usuário final.

2.4. RFID (Radio-Frequency Identification, Identificação por Rádio-Frequência)

Segundo a empresa Simber (2011) a tecnologia RFID, o princípio de funcionamento da tecnologia RFID é o seguinte:

> "*uma fonte, normalmente chamada de antena/leitora, emite uma onda de rádio freqüência, podendo ter diferentes comprimentos de onda. Um outro componente, chamado transponder, tag ou mesmo de etiqueta RFID, ao receber o sinal da antena/leitora, responde emitindo um sinal que por sua vez é identificado pela fonte emissora.*"(SIMBER, 2011)

2.5. TI (Tecnologia da Informação)

SAUR (1997) nos apresenta o seguinte conceito para tecnologia da informação:

"*conjunto de técnicas, máquinas e meios de suporte auxiliar para coletar, processar, armazenar e disseminar dados, que, tratados convenientemente, passam a constituir individual ou coletivamente, informações. Individualmente, essas várias ferramentas e essas várias ações nada tem de novo. São os progressos tecnológicos de cada uma dessas partes que formam hoje um conjunto realmente diferenciado, que integra profissionais com diversas funções antes exercidas separadamente, com diversas máquinas e meios de suporte muito além do velho papel e do quase velho computador.*"

3. METODOLOGIA

3.1 A pesquisa

A presente pesquisa propôs a identificação dos principais requisitos de um software para automação do controle de ordens de serviço em empresas que realizam o trabalho de manutenção em geral no ramo de TI. Tendo sido identificados estes recursos, foi desenvolvido um software aplicativo para automação de controle de ordens de serviço que sana as carências identificadas.

3.2 A seleção dos softwares para análise

Para a presente pesquisa, foram definidos alguns critérios que permitissem a seleção de softwares de ordens de serviço que servissem para análise, que são:

- O software deve ser gratuito, ou pelo menos poder ser testado gratuitamente.
- O software deve destinar-se a pequenas empresas.
- O software deve ter suporte ao sistema operacional Linux ou Windows**.**

Com base nos critérios, foi feita a seleção dos três softwares a seguir:

- Henning Comércio/Serviços for Windows SILVER 1.3.3
- Sistema VGDataTech da VGData Systems – Versão Freeware 1.111.2.15
- REPTecno O.S. Master 2.0.0.467

3.3 A definição dos critérios de análise dos softwares

Para que os softwares possam ser submetidos a uma análise crítica, faz-se necessária a adoção de critérios que sirvam como base de comparação. Critérios estes que permitam mensurar o quanto um dado software poderia ser útil em algum contexto, neste caso, na automação do controle de ordens de serviço em pequenas empresas. Estes critérios devem, pois, ser indagações sobre a existência de determinado recurso do software sob análise.

Para a elaboração dos critérios de análise, é preciso que exista um escopo que limite-os, os justifique e os classifique. Para este contexto, é pertinente que todos os critérios adotados servissem para fazer uma análise do software nos seguintes aspectos:

- Automatização - Automatizar determinada tarefa ou processo
- Disponibilidade - Aumentar a disponibilidade dos recursos de automação oferecidos pelo software
- Ampliação - Ampliar os recursos já oferecidos pelo software

Tendo sido estabelecido o escopo a que pertencem os critérios de análise dos softwares, é possível elaborá-los:

- Disponibilidade
    - O software funciona em rede ?
    - O software é gratuito ?
    - O software é multiplataforma ?
    - O software pode ser acessado de dispositivos móveis (celulares, smartphones ou tablets) ?
    - O software oferece uma interface para que o contratante (cliente) possa acompanhar o status da ordem de serviço ?
- Automatização
    - O software classifica a ordem de serviço de acordo com seu status?
    - O software calcula o valor da ordem de serviço de acordo com os serviços ou procedimentos executados ?
    - O software permite o cadastramento prévio dos técnicos ?
    - O software permite o cadastramento prévio dos atendentes ?
    - O software permite o cadastramento prévio dos clientes ?
    - O software permite o cadastramento prévio dos produtos ?
    - O software permite o cadastramento prévio dos equipamentos ?
    - O software permite a identificação do atendente responsável pela abertura da ordem de serviço ?
    - O software emite relatórios de abertura/fechamento das ordens de serviço ?
- Amplicação
    - O software é de código aberto ?

3.4 Análise dos softwares sob os critérios definidos

Para que fosse feita a análise dos softwares, os critérios de avaliação foram organizados numa tabela dividia em seções coloridas, onde cada cor representa o tipo de análise que está sendo feita

dos softwares. A cor vermelha representa os critérios de análise de disponibilidade. A cor verde representa a análise de automatização. E a cor lilás representa a extensibilidade, como vê-se na tabela 1.

Tabela 1: Análise dos recursos dos softwares a partir dos critérios de análise definidos

Critérios de Análise dos Softwares	**Henning Comércio/Serviços**	**VGDataTech**	**REPTecno O.S**
O software funciona em rede?	X	X	X
O software é gratuito?	X	--	--
O software é multiplataforma?	--	--	--
O software é acessível por dispositivos móveis (celulares, smartphones, tablets)?	--	--	--
O software oferece uma interface para que o contratante (cliente) possa acompanhar o status da ordem de serviço?	--	--	--
O software é de código aberto?	--	--	--
O software classifica a ordem de serviço de acordo com seu status?	X	--	X
O software calcula o valor da ordem de serviço de acordo com os serviços ou procedimentos executados?	X	--	X
O software permite o cadastramento prévio dos técnicos?	--	X	X
O software permite o cadastramento prévio dos atendentes?	X	X	--
O software permite o cadastramento prévio dos clientes?	X	X	X
O software permite o cadastramento prévio dos produtos?	X	X	X
O software permite o cadastramento prévio dos equipamentos?	X	--	X
O software permite a identificação do atendente responsável pela abertura da ordem de serviço?	X	X	--
O software emite relatórios de abertura/fechamento da ordens de serviço?	X	--	X
O software calcula o valor da ordem de serviço de acordo com os serviços ou procedimentos executados?	X	--	X

A partir da análise dos softwares escolhidos, percebe-se que existem alguns recursos que nenhum

dos escolhidos preocupa-se em fornecer, como está evidenciado na Tabela 2:

Tabela 2: Evidenciação dos recursos não oferecidos pelos softwares

Critérios de Análise dos Softwares	Henning Comércio/Serviços	VGDataTech	REPTecno O.S
O software é multiplataforma?	--	--	--
O software é acessível por dispositivos móveis (celulares, smartphones, tablets)?	--	--	--
O software oferece uma interface para que o contratante (cliente) possa acompanhar o status da ordem de serviço?	--	--	--

Cada um destes recursos que os softwares não oferecem são de grande importância, conforme descrito a seguir:

- O software é multiplataforma? É de grande importância que os softwares da atualidade sejam desenvolvidos de forma que não dependam de plataformas específicas. No passado o Microsoft Windows era o sistema operacional mais utilizado entre os usuários, sendo extremamente raro encontrar usuários comuns utilizando outros sistema operacionais. No entanto, sistemas como o Linux vem ganhando muito espaço entre os usuários comuns, devido ao fato de ser livre, de código aberto, estável e confiável (Morimoto 2009). Por este motivo, é importante disponibilizar também para esta plataforma os softwares de que as pessoas possam precisar.
- O software é acessível por dispositivos móveis (celulares, smartphones, tablets)? Com o avanço das telecomunicações no brasil, as pessoas têm utilizado cada vez mais os dispositivos móveis para, além de se comunicar, trabalhar. Por este motivo, softwares que podem ser utilizados a partir de dispositivos móveis já se tornaram um diferencial de competitividade para as empresas que os utilizam. Deste modo, é de grande importância que os softwares, principalmente os que estejam relacionados com a automação comercial, sejam desenvolvidos de modo a poder ser acessados através de dispositivos móveis.
- O software oferece uma interface para que o contratante (cliente) possa acompanhar o status da ordem de serviço? Oferecer uma interface de software que permita ao cliente interagir com a empresa nada mais é que levar ao cliente as informações de que ele necessita, sem necessidade de intervenção humana para isso. Deste modo, um software com este recurso representa uma grande ferramenta para a empresa que o utiliza. Também é importante que o cliente possa interagir com a empresa sem a necessidade de se deslocar. Dentro do contexto do controle de ordens de serviço,

isso significa oferecer ao cliente a possibilidade de aprovar ou rejeitar o orçamento referente a sua ordem de serviço contratada on-line (e também através de dispositivos móveis).

São principalmente estes três recursos que procurou-se oferecer no software desenvolvido.

A seguir as indicações da metodologia para o desenvolvimento do sistema que contemple as características discutidas nesta seção.

3.5 Desenvolvimento do software

Pelos diversos motivos já citados neste trabalho, o software foi desenvolvido utilizando-se como linguagem de programação o PHP. Segundo Lerdorf, Tatroe e Intyre (2006, p. 1), o PHP é gratuito, roda em sistemas Linux, Unix, Windows, e Macintosh OS X.

A escolha do PHP como linguagem de programação foi feita também porque para que o software que pudesse agregar os recursos propostos pela pesquisa, ele deveria poder ser acessado por navegadores. Isso possibilitou que o software fosse absolutamente independente de plataforma, e ainda que pudesse ser acessado de dispositivos móveis.

Outro motivo importante para a escolha do PHP como linguagem de programação, foi o fato de o PHP suportar diversos recursos de programação orientada a objetos (LERDORF;TATROE e INTYRE; 2006, p. 143),. Isso permitiu a produção de um código legível, de fácil reaproveitamento e de fácil manutenção.

A escolha do MVC como padrão de arquitetura de software se deu devido a possibilidade de separar a lógica de negócio da lógica de apresentação (MILLER, VANDOME e MCBREWSTER,2010, p. 1). Este padrão evitou, entre outras coisas, que no momento da construção das telas fosse necessário trabalhar com regras de negócio ou com interações com o banco de dados.

O MVC é também muito flexível, o que permitiu uma aplicação bem adaptada a realidade do software. Desse modo, o software foi projetado para ser um conjunto de módulos, e cada módulo seria desenvolvido utilizando-se a arquitetura MVC. O conjunto de todos os módulos do sistema está descrito no diagrama da Figura 2:

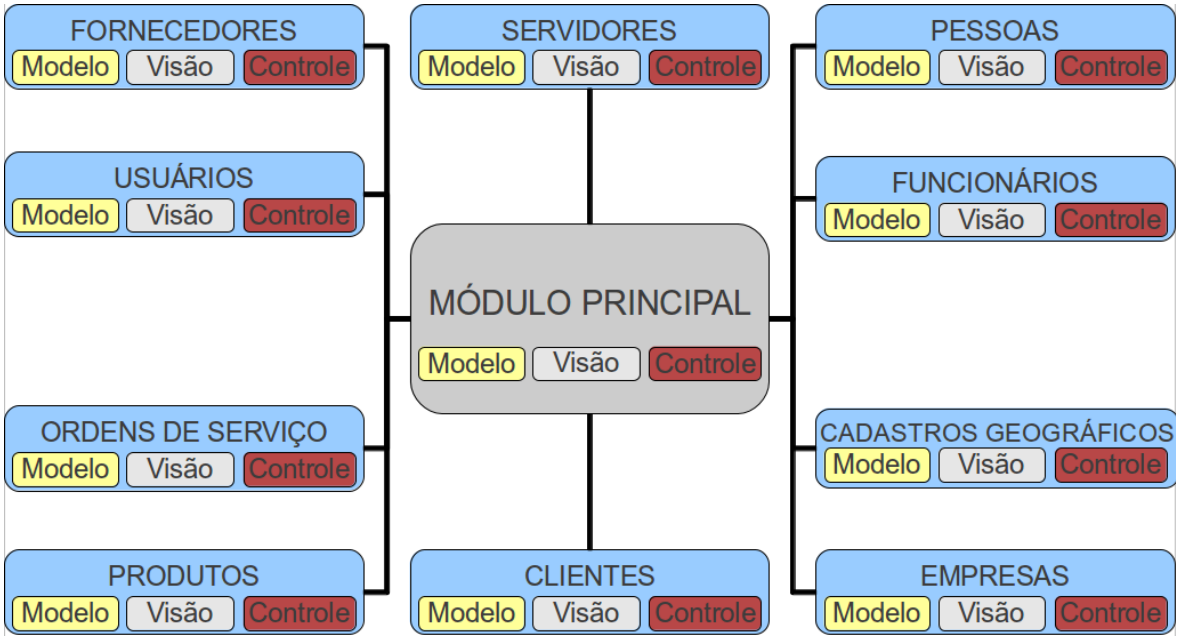


Figura 2: Estrutura modular do software

Neste diagrama, a caixa cinza grande representa o módulo principal, que conforme poderá ser visto nos resultados, define classes das quais dependem os demais módulos. As caixas azuis representam os demais módulos do sistema. Cada módulo, seguindo o padrão MVC, tem a camada modelo (representada pelas caixas amarelas no diagrama), a camada visão (representada pelas caixas cinzas pequenas no diagrama) e a camada controle (representada pelas caixas vermelhas). Cada uma destas camadas, em cada módulo do sistema, está implementada num arquivo separado. É importante ressaltar que existem outros arquivos que integram os módulos, que serão apresentados em cada módulo no capítulo 4 (resultados).

3.5.1 Sistema gerenciador de banco de dados

A escolha do PostgreSQL como SGBD foi feita por que ele possui, segundo Douglas, K; e Douglas, S; (2003, p. 1 até 2), diversos recursos, dentre os quais foram importantes ao presente trabalho os seguintes:

- Objeto-relacional: No PostgreSQL, toda tabela define uma classe. O PostgreSQL implementa herança entre tabelas. Funções e operadores são polimórficos.
- Compatível com os padrões: A sintaxe do PostgreSQL implementa a maioria dos padrões estabelecidos do SQL92 e muitos recursos do SQL99. As diferenças na sintaxe do PostgreSQL e a dos padrões, são em sua maioria relacionadas a recursos específicos deste banco de dados.
- Processamento de Transações: O PostgreSQL protege os dados e coordena o uso

concorrente de múltiplos usuários através do processamento completo de transações.

- Integridade referencial: O PostgreSQL implementa integridade referencial de forma completa, suportando relacionamentos do tipo chaves primárias e estrangeiras, e também *triggers*.

O PostgreSQL conta também com vasta documentação disponível em diversas línguas, via internet, e está em constante desenvolvimento.

Para a utilização do banco de dados junto a linguagem PHP, foi necessário a utilização do pacote phppgsql, que provê as funções necessárias para estabelecer a conexão entre o software e o servidor do banco de dados PostgreSQL.

No próximo capítulo, será apresentado o resultado do presente trabalho que é o software Smart Mill.

4 RESULTADOS

Neste capítulo são apresentados os resultados do presente trabalho, o software Smart Mill, que será apresentado módulo por módulo.

4.1 O módulo principal

O módulo principal é responsável por estabelecer as classes principais que precisam ser herdadas pelos demais módulos. No capítulo 8 (anexos), este módulo será descrito mais profundamente.

4.2 O módulo Servidores

No estado atual do software, este módulo armazena informações relevantes a respeito do servidor (computador) que roda o software. Ele faz isso através da tabela “servidores”, pertinente a este módulo.

O módulo Servidores foi desenvolvido, também, com a finalidade de que se possa, futuramente, tornar o software utilizável em filiais geograficamente separadas. Neste caso seria necessário que cada registro do banco de dados compartilhado entre duas ou mais filiais contivesse informações que permitissem identificar qual o servidor foi responsável pela geração do registro, evitando assim conflitos de registros de valores únicos (unique key) no banco de dados compartilhado. Desse modo, todas as outras tabelas do sistema que não a “servidores”, se relacionam com essa através de chaves estrangeiras. Na Figura 3 abaixo é possível ver o formulário de cadastramento do servidor, que é utilizado apenas nas configurações iniciais do software.

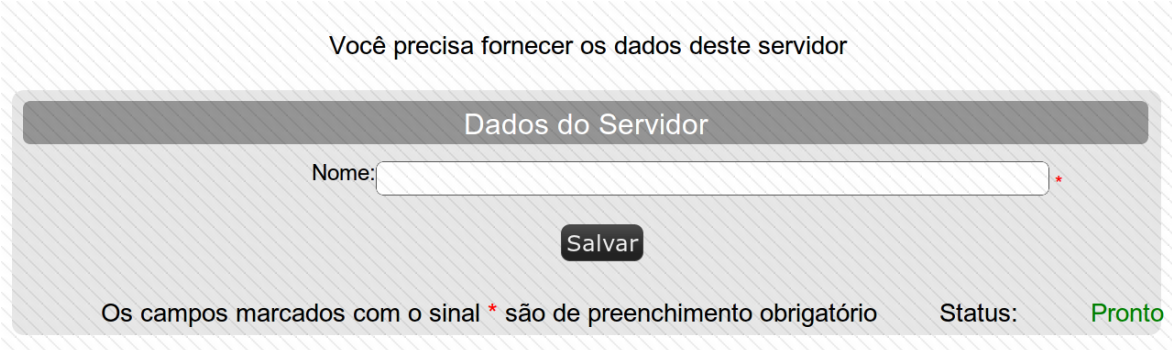


Figura 3: Formulário para definição do nome do servidor

Os formulários do software, como o que está demostrado na Figura 3, são acessados através de navegadores de internet, e se comunicam com o servidor através de requisições.

4.3 O módulo Usuários

O módulo Usuários foi desenvolvido com as seguintes finalidades:

1. Cadastrar, nas configurações iniciais do software, um grupo de usuários administradores e um usuário pertencente a este grupo para acesso inicial ao software;

2. Permitir que novos usuários e grupos sejam cadastrados;

3. Permitir que os dados dos usuários e grupos existentes sejam alterados;

4. Permitir que os usuários e grupos existentes sejam excluídos (exceto se um usuário tentar excluir a si mesmo, neste caso o módulo não permite a exclusão);

5. Permitir que se possa adicionar ou remover permissões para determinado grupo de usuários que não seja administrador.

6. Exibir uma tela de login para que se possa identificar o usuário que acessa o sistema, mediante digitação de login (ou e-mail) e senha.

Esse módulo é responsável por apresentar os diversos formulários que fornecem as funcionalidades acima descritas. Esses formulários podem ser vistos nas Figuras 4, 5, 6 e 7.

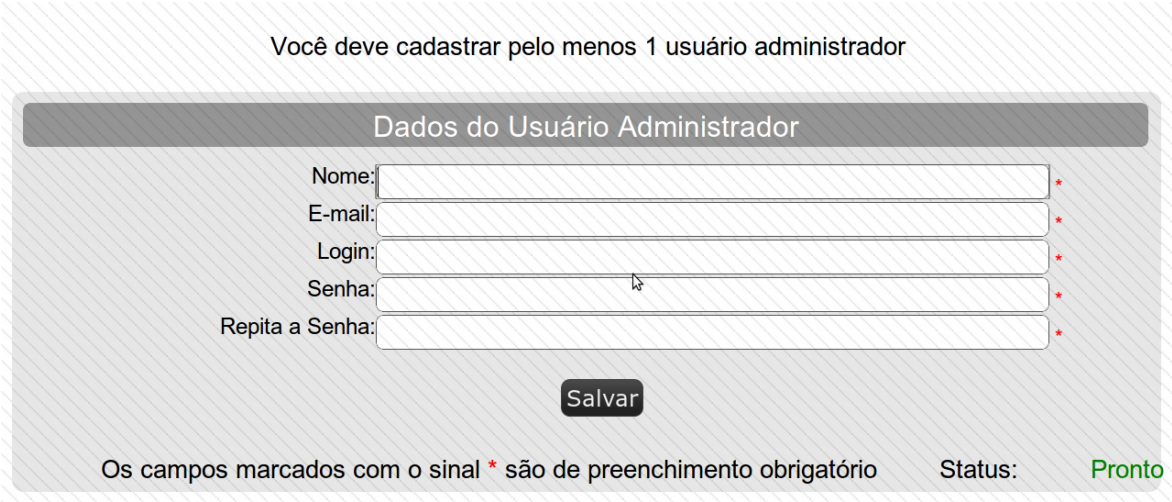


Figura 4: Formulário para cadastramento do usuário administrador inicial

A Figura 4 mostra o formulário de cadastramento do usuário administrador do software. Este formulário é acessado sempre nas configurações iniciais do software, de modo que só precisa ser acessado uma vez.

A existência deste formulário se deve ao fato de que é necessário que haja pelo menos 1 usuário

administrador para que o sistema prossiga com seu funcionamento. Assim, este formulário permite que seja cadastrado um usuário inicial para o software, que é cadastrado como administrador (isto é, pertencente ao grupo dos administradores).

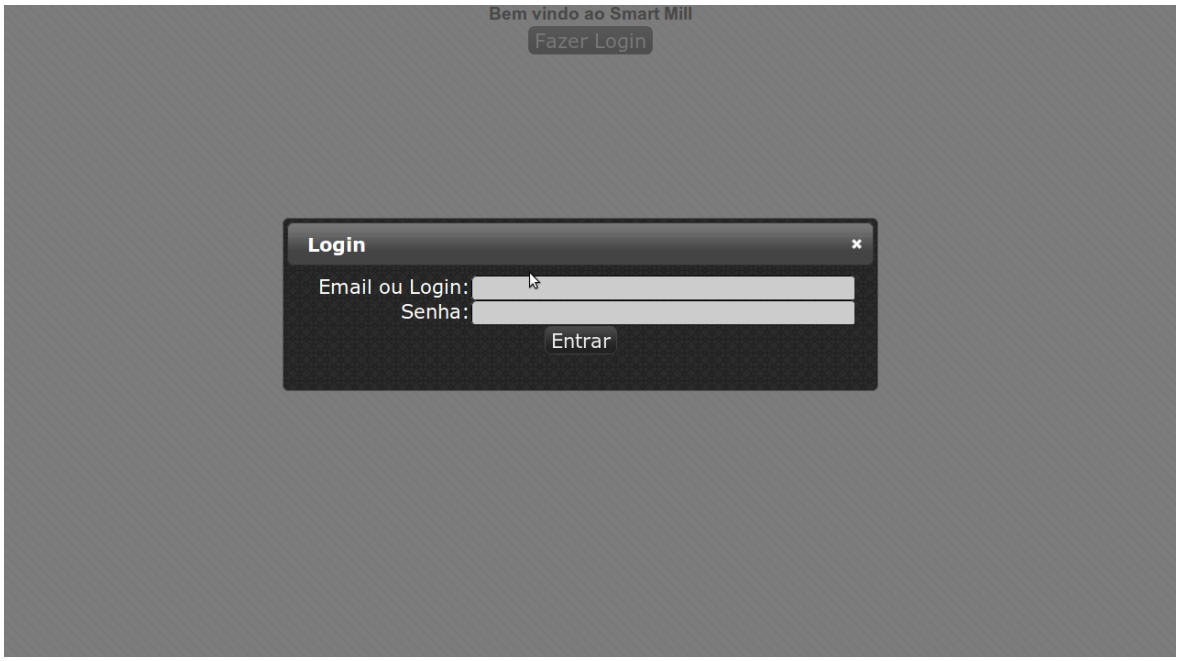


Figura 5: Formulário para efetuar login no sistema

A Figura 5 apresenta a tela de login, que é apresentada sempre que não há nenhum usuário logado no software. A partir desta tela, o usuário pode se autenticar no sistema digitando seu e-mail ou login, e a sua senha. O software compara estas informações com as informações existentes no banco de dados e libera o acesso do usuário, desde que seus dados tenham sido inseridos corretamente.

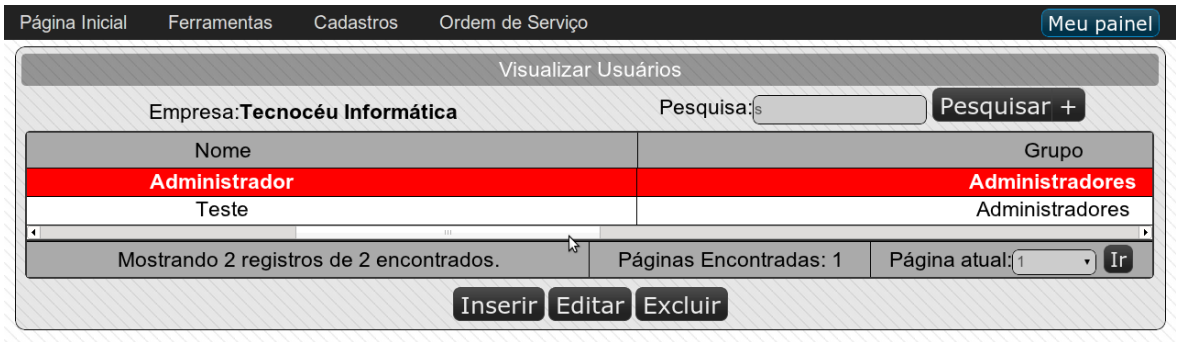


Figura 6: Tela para visualização dos usuários cadastrados, através de pesquisa

A Figura 6 mostra a tela de visualização dos usuários cadastrados. Esta tela é útil no sentido de permitir que se possa visualizar os usuários cadastrados no software, bem como seus respectivos grupos.

Nesta tela é possível pesquisar por um usuário específico utilizando o campo de pesquisa no canto superior direito. É possível também editar ou excluir usuários que tenham sido cadastrados no

software.

Figura 7: Formulário para edição dos dados do usuário

Na Figura 7 é mostrada a tela de edição de usuários. Esta tela pode ser acessada com o propósito de editar os dados de determinado usuário. Também existe uma tela similar para a inserção de novos usuários no sistema, na qual se pode também especificar o grupo a que pertence o usuário que se pretende inserir.

4.4 O módulo Cadastros Geográficos

O módulo Cadastros Geográficos tem por finalidades :

1. Permitir que se possa cadastrar países;
2. Permitir que se possa editar dados de países cadastrados;
3. Permitir a exclusão de países cadastrados;
4. Permitir que se possa cadastrar estados;
5. Permitir que se possa editar dados de estados cadastrados;
6. Permitir a exclusão de estados cadastrados;
7. Permitir que se possa cadastrar Cidades;
8. Permitir que se possa editar cidades cadastradas;
9. Permitir a exclusão de cidades cadastradas.

O desenvolvimento deste módulo foi de grande importância, pois existem outros módulos que dependem dos cadastros geográficos. O módulo Pessoas é um deles (pois uma pessoa cadastrada

pode pertencer a uma cidade, que deve ser cadastrada previamente no módulo Cadastros Geográficos).

A camada de apresentação do módulo Cadastros Geográficos tem por finalidade apresentar formulários que permitam que o usuário possa realizar as diversas operações oferecidas por esse módulo, que serão descritas no capítulo 8 (anexos).

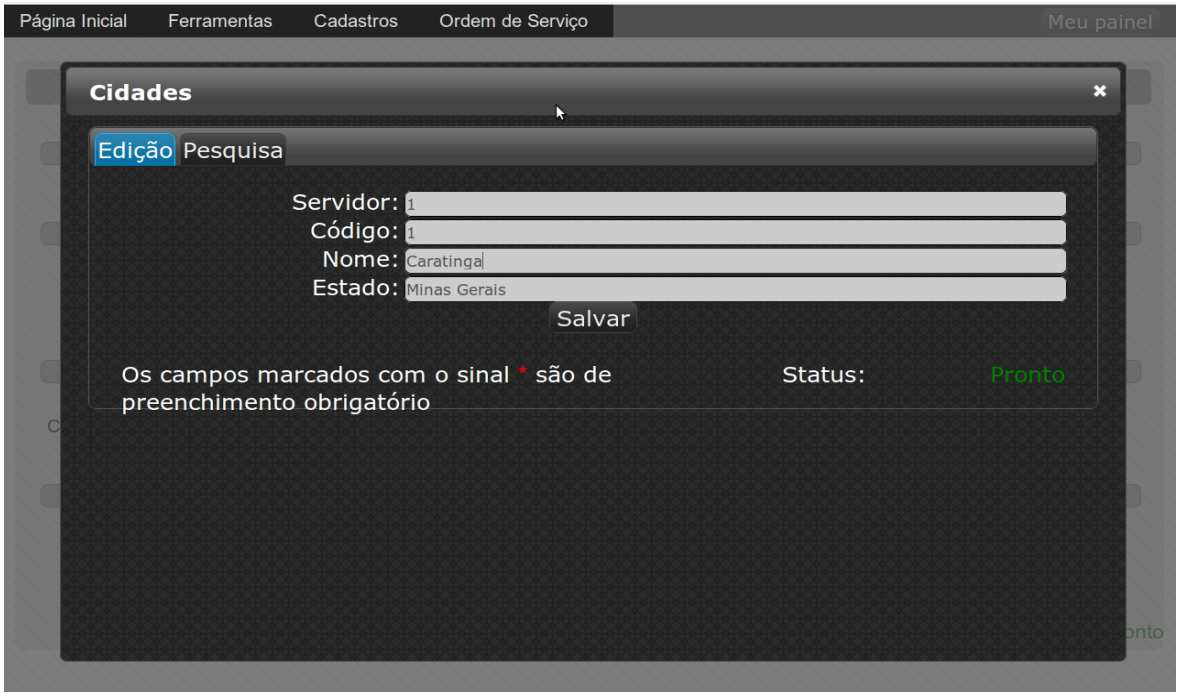


Figura 8: Tela de cadastramento de cidades

A Figura 8 exibe a tela cadastramento de cidades. Esta tela permite não apenas cadastrar novas cidades, mas também editar os dados das cidades já cadastradas no software. Existe a possibilidade e também a necessidade de que se especifique o estado a que pertence a cidade que se pretende cadastrar. Caso o estado que se queira associar a cidade não tenha sido cadastrado no software, é possível cadastrá-lo a partir da tela de cadastramento de estados, que pode ser vista na Figura 14:

Figura 9: Tela de cadastramento de estados

A Figura 9 mostra a tela de cadastramento de estados, que funciona de forma similar a tela de cadastramento de cidades. Ao cadastrar determinado estado, é necessário associá-lo a algum país cadastrado previamente no software.

Figura 10: Tela de cadastramento de países

A tela de cadastramento de países, exibida na Figura 10, permite o cadastramento de países no software, onde se pode informar o nome do país e a sigla do mesmo, que deve ter no mínimo dois

caracteres, e pode ter até 4 caracteres.

4.5 O módulo Pessoas

O módulo Pessoas é um dos módulos mais importantes dentro do software. Ele permite, através da tabela pessoas, que todos os cadastros referentes a pessoas centralizem-se numa única tabela. Por exemplo, todos os clientes e fornecedores ficam cadastrados nesta tabela, o que os diferencia são as chaves estrangeiras que os referenciam nas tabelas “clientes” (do módulo Clientes) e “fornecedores” (do módulo Fornecedores). Deste modo, é possível que uma mesma pessoa possa ser cliente e fornecedor ao mesmo tempo (caso, é claro, isso seja necessário), evitando cadastros redundantes.

A camada de apresentação desse módulo tem por finalidade apresentar formulários que permitam que o usuário possa realizar as diversas operações oferecidas por esse módulo, que serão descritas no capítulo 8 (anexos).

Figura 11: Tela de cadastramento de pessoas

A Figura 11 mostra a tela de cadastramento de Pessoas. Nesta tela se pode especificar diversos dados a respeito da pessoa que se pretende cadastrar, tais como o tipo de pessoa (pessoa física ou pessoa jurídica), a razão social da pessoa, o nome fantasia da pessoa (no caso de pessoa jurídica), o CNPJ da pessoa (no caso de ser uma pessoa jurídica) ou o CPF da pessoa (no caso de ser uma pessoa física), dados a respeito do endereço da pessoa, dados para contatação remota, etc.

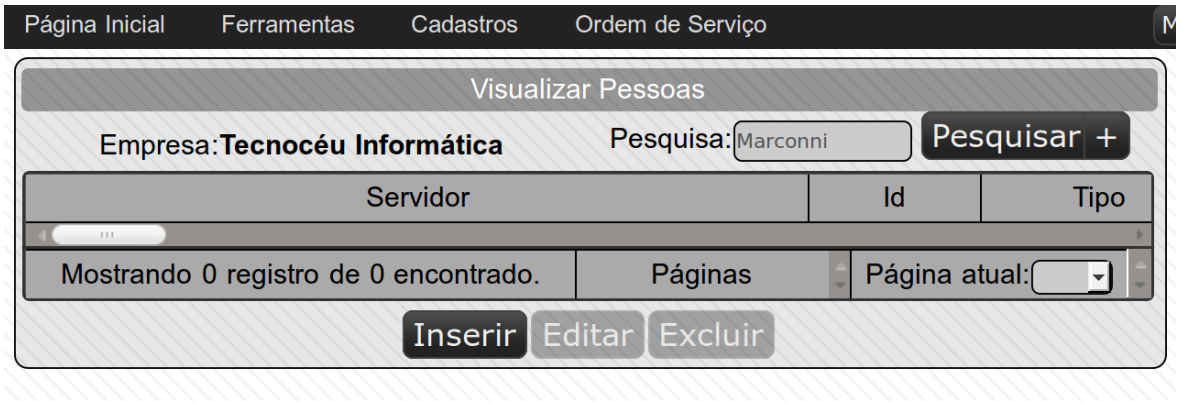


Figura 12: Tela de visualização e pesquisa de pessoas

A tela de visualização de pessoas, mostrada na Figura 12, permite ao usuário do sistema pesquisar e obter dados a respeito das pessoas cadastradas no sistema. É possível a partir desta tela que se selecione determinada pessoa para editar seus dados ou mesmo excluí-la. A exclusão de determinada pessoa se dá mediante determinadas condições. Por exemplo, se determinada pessoa que se pretenda excluir estiver associada a um cliente (isto é, se a pessoa for o cliente) ao qual estejam relacionadas ordens de serviço cadastradas no software, o software não permitirá a exclusão da pessoa e informará ao usuário este motivo, para que o usuário possa tomar alguma decisão.

4.6 O módulo Empresas

O módulo Empresas foi desenvolvido com o intuito de possibilitar que se armazene as informações referentes a empresa que faz uso do software. Este módulo poderá, futuramente, ganhar novas funcionalidades, tais como relatórios referentes a vida da empresa.

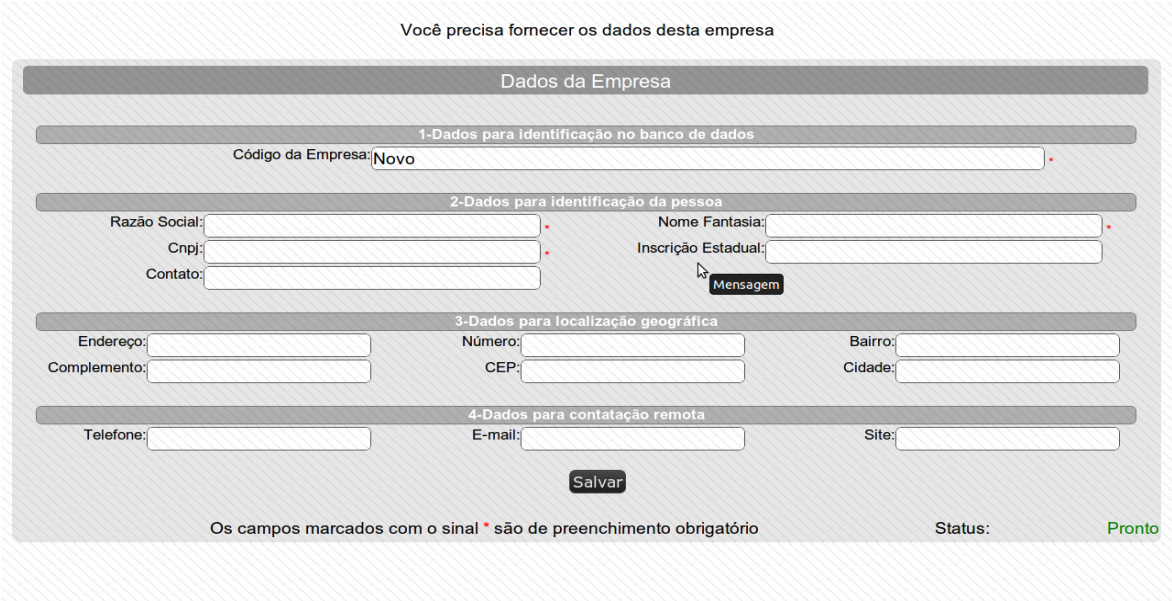


Figura 13: Tela de cadastramento da empresa usuária

A Figura 13 exibe a tela que permite a empresa usuário do software informar seus dados para que sejam registrados no banco de dados. Existe grande semelhança entre esta tela e a tela de cadastramento de pessoas, afinal uma empresa é uma pessoa. A diferença está no fato de que nesta tela não se pode especificar o tipo da pessoa, visto que a empresa usuário deve ser sempre uma pessoa jurídica. Desse modo, os dados que podem ser informados são referentes apenas a pessoas jurídicas (CNPJ por exemplo).

4.7 O módulo Clientes

O módulo Clientes permite o cadastramento dos clientes da empresa que utiliza o software. Este módulo depende do módulo pessoas, pois um cliente é, naturalmente, uma pessoa.

Futuramente este módulo poderá ganhar novos recursos, tais como relatórios que permitam identificar a situação financeira de cada cliente, relatórios de clientes de maior frequência, etc.

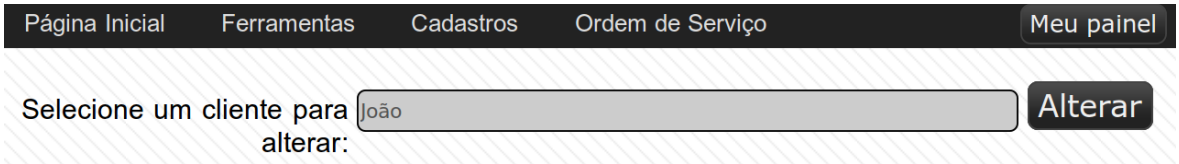


Figura 14: Tela de seleção de cliente para edição de dados

A Figura 14 mostra a tela de seleção de clientes para alteração. Nesta tela existe um campo, no qual se pode digitar parcialmente o nome do cliente que se deseja editar. Então é necessário que se pressione a tecla enter para que o software faça a pesquisa no banco de dados, na tabela de clientes. O software retorna então uma lista contendo os clientes cujos nomes parecem-se com o digitado. O

usuário pode então selecionar um cliente nesta lista e clicar no botão “Alterar”. O software então redirecionará o navegador do software para uma página a edição dos dados do cliente selecionado.

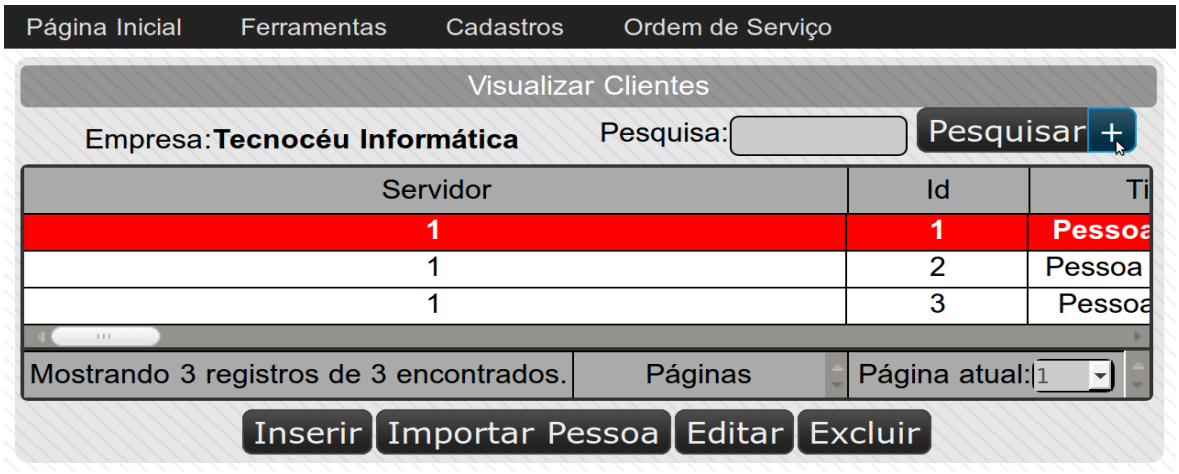


Figura 15: Tela de visualização e pesquisa de clientes

A Figura 15 mostra a tela de visualização de clientes. Nesta tela se pode pesquisar entre os clientes existentes no banco de dados.

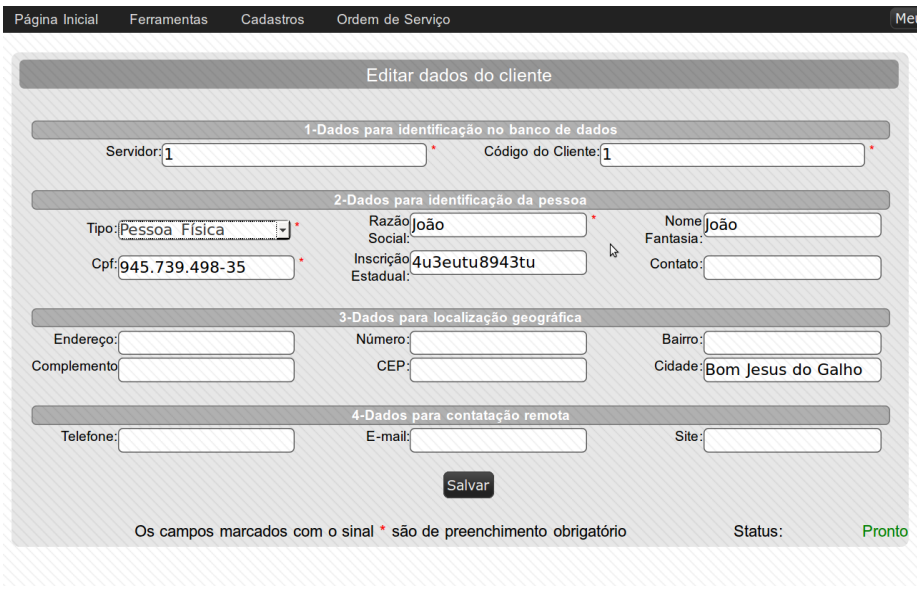


Figura 16: Tela de edição de dados de clientes

Na Figura 16 é mostrada a tela de edição de dados de clientes. Esta tela permite que se altere dos dados dos clientes existentes no banco de dados do software.

Nota-se grande semelhança entre esta tela e a tela de edição (ou de cadastramento) de pessoas. Isto ocorre porque esta tela é derivada da tela de edição de pessoas, afinal um cliente é uma pessoa.

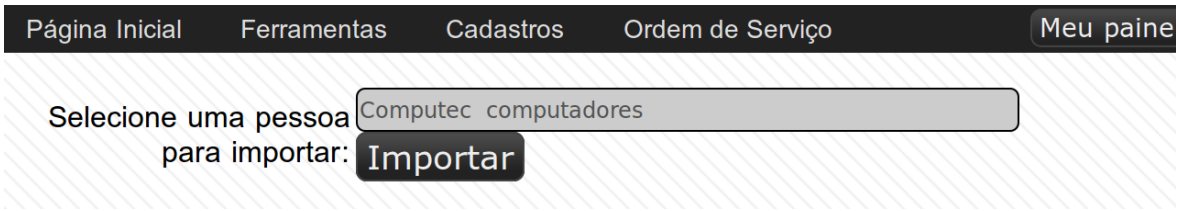


Figura 17: Tela de importação de pessoas

A tela de importação de pessoas, mostrada na Figura 17, permite que uma pessoa previamente cadastrada no software se torne também um cliente, evitando assim cadastros redundantes.

4.8 O módulo Fornecedores

O módulo Fornecedores permite o cadastramento dos fornecedores de produtos da empresa que utiliza o software. Este módulo depende do módulo pessoas, pois um fornecedor é, naturalmente, uma pessoa. E desse módulo depende diretamente o módulo Produtos, pois um produto pode estar relacionado a seu respectivo fornecedor.

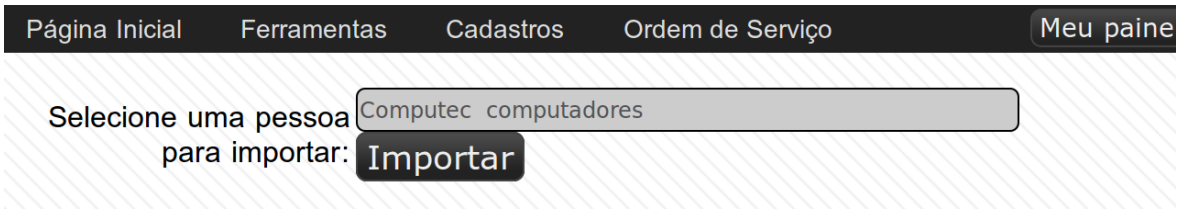


Figura 18: Tela de importação de pessoas

A Figura 18 mostra a tela de importação de pessoas. Esta tela funciona de modo similar a tela de importação de pessoas do módulo Clientes, neste caso permitindo que pessoas previamente cadastradas no software se tornem também fornecedores.

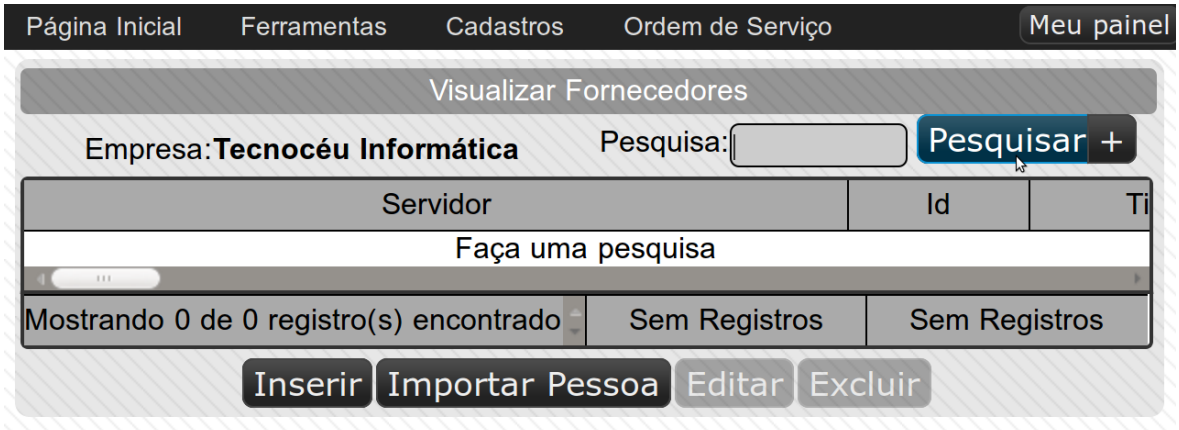


Figura 19: Tela de visualização e pesquisa de fornecedores

A Figura 19 mostra a tela de visualização de fornecedores, onde é possível pesquisar pelos fornecedores que existem no banco de dados do software e então executar ações tais como exclusão

ou edição dos dados (neste caso numa tela específica para este fim).

4.9 O módulo Estoque

O módulo Estoque foi desenvolvido de modo a permitir que a empresa que utiliza o software obtenha um maior controle de seu estoque. Este módulo permite o cadastramento dos produtos do estoque, relacionados a seus respectivos fornecedores, e permite que se obtenha controle da quantidade disponível de cada produto no estoque.

Figura 20: Tela de edição de dados de produtos

A Figura 20 mostra atela de edição de dados de produtos. Nesta tela é possível editar diversos dados do produto, tais como o Nome do produto, o fornecedor que fornece este produto a empresa que utiliza o software, o custo do produto, a margem de lucro e o valor de venda do produto.

É importante ressaltar que nesta tela o valor de venda do produto pode ser calculado automaticamente pelo software, bastando apenas que o usuário informe o custo do produto e a margem de lucro pretendida para o mesmo.

Apesar de o software ainda não possuir um módulo de vendas, o valor de venda do produto é importante, pois auxilia na determinação do valor das ordens de serviço.

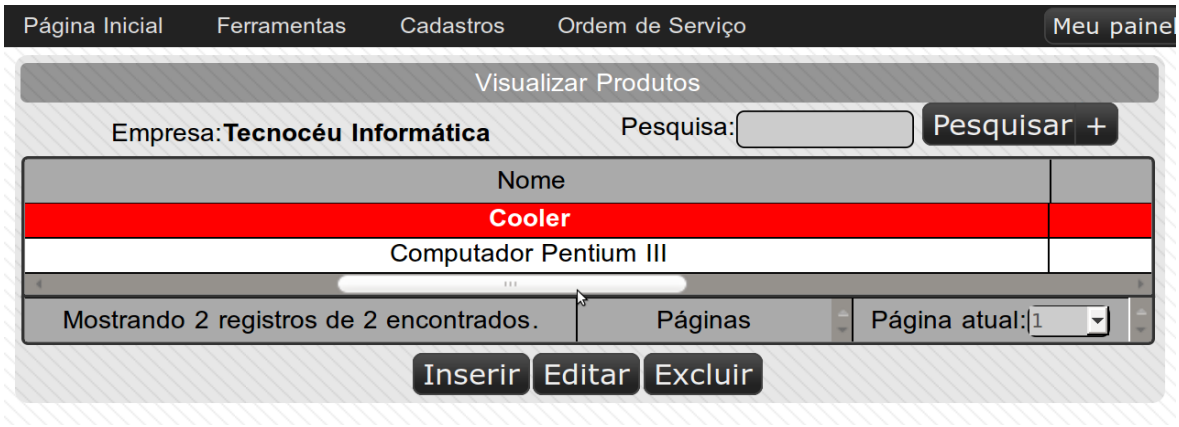


Figura 21: Tela de visualização e pesquisa de produtos

A Figura 21 mostra a tela de visualização de produtos, onde é possível pesquisar pelos produtos cadastrados no banco de dados do software, e a partir dos resultados da pesquisa realizar operações tais como a exclusão de um ou mais produtos (caso haja necessidade).

4.10 O módulo Funcionários

O módulo Funcionários foi desenvolvido para permitir que a empresa que utiliza o software possa cadastrar e gerenciar seus funcionários. Este módulo também é importante porque dele dependem outros módulos, tais como o módulo Técnicos, e o módulo Atendentes.

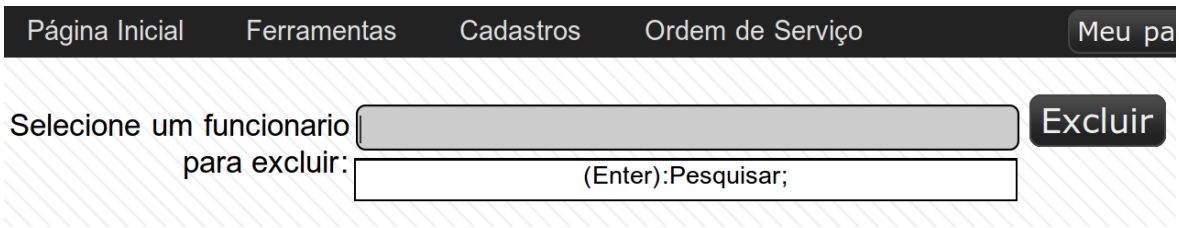


Figura 22: Tela de seleção de funcionários para exclusão

A Figura 22 mostra a tela de exclusão de funcionários, onde é exibido um campo no qual o usuário deve digitar parte do nome do funcionário que quer excluir e pressionar a tecla enter. O software então faz uma busca no banco de dados e retorna os funcionários cujos nomes contém o texto digitado. O usuário pode então escolher na lista o funcionário que deseja excluir, e então clicar no botão “Excluir”.

A exclusão de funcionários no software ocorre mediante algumas restrições, por exemplo, não é possível excluir funcionários aos quais existam técnicos de manutenção associados. Para isso, faz-se necessária a exclusão prévia do técnico para que se possa então excluir o funcionário a ele associado.

É importante ressaltar que ao excluir determinado funcionário, o software não exclui também a pessoa a ele associada. Caso se queira excluir também a pessoa, o usuário deve então acessar o

módulo Pessoas e efetuar a exclusão, que também ocorrerá mediante determinadas restrições que já foram exemplificadas.

Figura 23: Tela de edição de dados de funcionários

A Figura 23 apresenta a tela de edição de dados de funcionários. Assim como no caso da tela de edição de dados de clientes, nota-se grande semelhança entre esta tela e a tela de edição de dados de pessoas. Isto ocorre porque esta tela é derivada da tela de edição de pessoas.

4.11 O módulo Ordens de Serviço

O módulo Ordens de Serviço é o módulo de maior responsabilidade dentro do sistema, porque é responsável por resolver grande parte dos problemas propostos . Este módulo foi desenvolvido com o propósito de permitir que o uso do sistema, no que se refere ao controle de ordens de serviço, ocorresse dentro das seguintes etapas para cada ordem de serviço:

1. O cliente (ou contratante) solicita um serviço de reparo para determinado equipamento. O cliente é então cadastrado, caso seu cadastro ainda não exista.

2. O atendente faz então a abertura da ordem de serviço. O software imprime o relatório de abertura de ordem de serviço (em duas vias, uma para a empresa e outra para o cliente), que contém diversos dados, tais como: O número da ordem de serviço, os dispositivos que acompanham o equipamento, estado de conservação do equipamento, número de série do equipamento, marca do

equipamento, problema relatado pelo cliente, e a senha gerada para que o cliente possa fazer o acompanhamento on-line da ordem de serviço por ele contratada. Neste momento a ordem de serviço assume o status “Aguardando”.

3. O software então coloca a ordem de serviço contratada na fila de espera, com prioridade proporcional a data de abertura da ordem de serviço, de modo que ordens de serviço mais antigas possuam maior prioridade. Esta lista de espera estará disponível para que os técnicos da empresa a visualizem, possibilitando que eles possam assumir as ordens de serviço que desejarem levando em consideração a prioridade das mesmas.

4. Quando um determinado técnico assume o controle de determinada ordem de serviço, a ordem de serviço assume o status “Em Análise”, que significa que o técnico está analisando o equipamento referente a ordem de serviço. Neste momento, o técnico responsável pela ordem de serviço poderá acessar uma tela que lhe permita adicionar os procedimentos de reparo que forem necessários, de acordo com o problema identificado no equipamento do cliente. Esta tela permite também que o técnico visualize os procedimentos já adicionados por ele.

5. Após ter concluído o orçamento da ordem de serviço (isto é, após ter identificado todos os procedimentos de reparo necessários ao equipamento), o técnico poderá acionar um botão que mude o status da ordem de serviço para “Orçamento Pronto”.

6. O cliente poderá então ser contatado pelos atendentes da empresa a fim de que avalie o orçamento feito pelo técnico e aprove ou rejeite o orçamento. O cliente pode aprovar ou rejeitar o orçamento utilizando uma tela on-line, ou pode fazer sua aprovação junto aos atendentes por telefone ou de maneira presencial. Neste caso são os atendentes que, através de uma tela especial para eles, aprovam ou rejeitam o orçamento (sob consentimento do cliente). A ordem de serviço pode então assumir dois status: “Orçamento Aprovado” ou “Orçamento Rejeitado”.

7. O técnico responsável pela ordem de serviço deverá então agir de acordo com o status da ordem de serviço, ou seja, se o cliente aprovou o orçamento feito pelo técnico, este deve concluir os procedimentos de reparo propostos no orçamento. Caso contrário o técnico deve agir a fim de que deixe o equipamento no mesmo estado de quando entrou na empresa. Então ele deverá, através de uma tela, mudar o status da ordem de serviço para “Entregar Equipamento”. Isto indicará que o equipamento deve ser devolvido ao cliente, ou que este deverá vir até a empresa buscar seu equipamento. Neste momento o software possibilitará ainda a impressão de um relatório de fechamento de ordem de serviço (em duas vias, uma para a empresa e outra para o cliente) contendo os dados que permitem a identificação da ordem de serviço (número da O.S., nome do cliente, dados do equipamento, data de abertura), os procedimentos nela executados, e o valor total do serviço realizado.

8. Após o equipamento ter sido entregue ao cliente, algum atendente (preferencialmente o atendente que abriu a ordem de serviço) deverá mudar o status da ordem de serviço para “Equipamento Entregue”. Este status indicará que a ordem de serviço está concluída, isto é, todos os procedimentos necessários foram executados e o equipamento foi entregue.

O módulo Ordens de Serviço foi desenvolvido de modo que permitisse que todas as etapas descritas pudessem ser executadas. Para isso, foi necessário que este módulo agregasse os seguintes recursos: Possibilidade de cadastro e gerenciamento de técnicos, atendentes, procedimentos de reparo executáveis, equipamentos e abertura e gerenciamento de ordens de serviço.

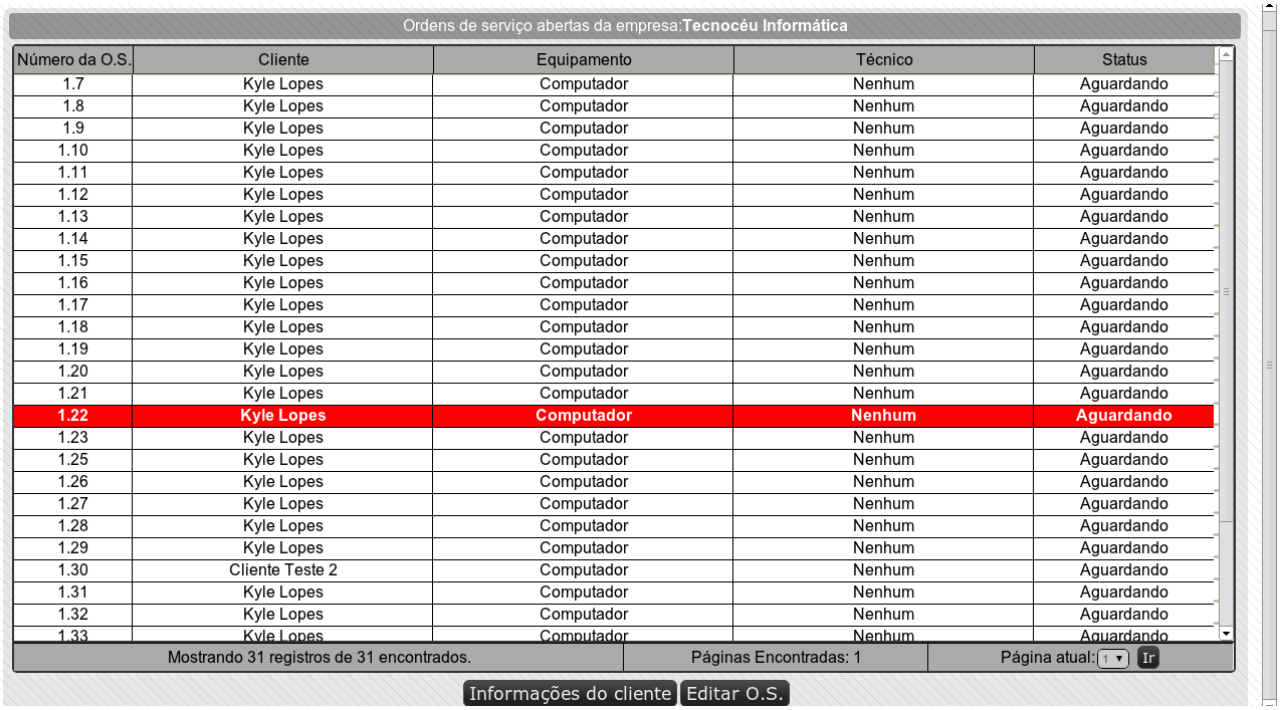


Figura 24: Tela de visualização de ordens de serviço dos atendentes

A Figura 24 mostra a tela de visualização de ordens de serviço dos atendentes. É importante que se entenda que existem duas telas que permitem a visualização de ordens de serviço, uma para o atendente e uma para o técnico de manutenção. Cada uma destas telas permitem que diferentes tipos de operações sejam realizadas.

A tela de visualização de ordens de serviço dos atendentes, por exemplo, permite que o atendente aprove ou rejeite o orçamento feito pelo técnico (mediante consentimento do cliente).

A tela de visualização de ordens de serviço dos técnicos de manutenção, por outro lado, permitem a este executar ações tais como o cadastramento de procedimentos de reparo efetuados no equipamento pertinente a ordem de serviço em que o técnico estiver trabalhando.

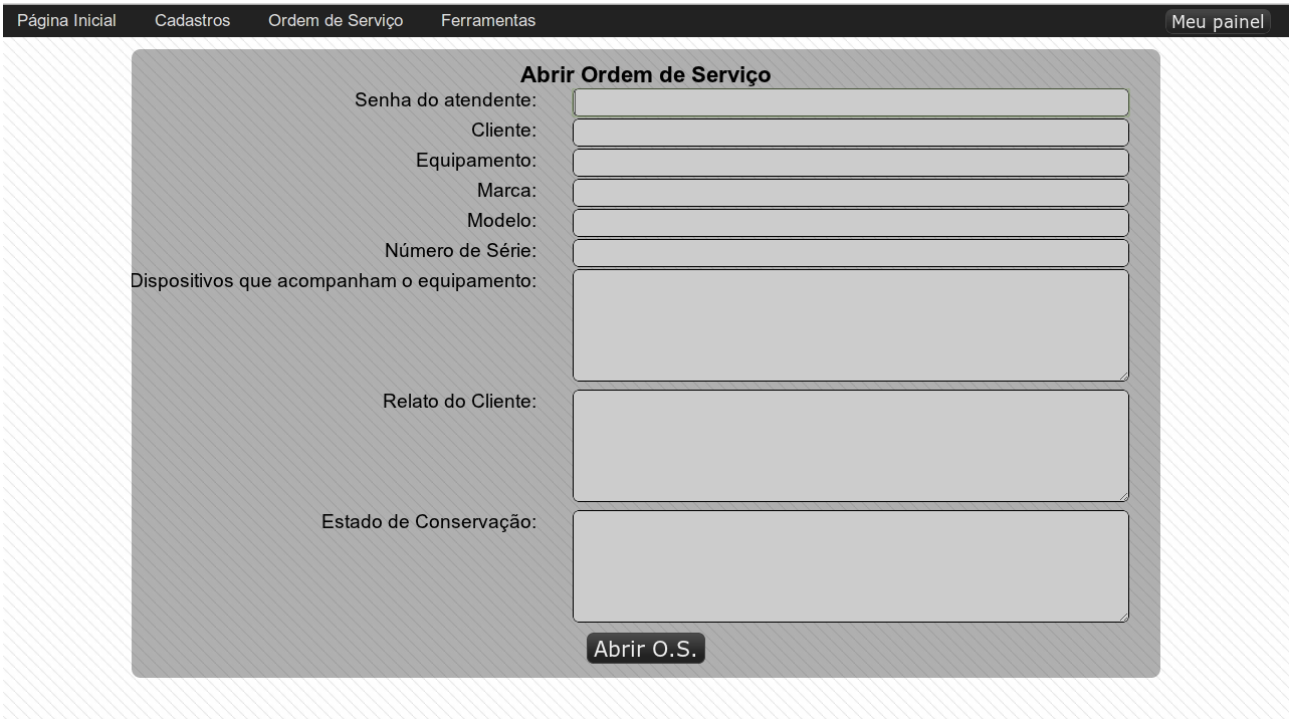


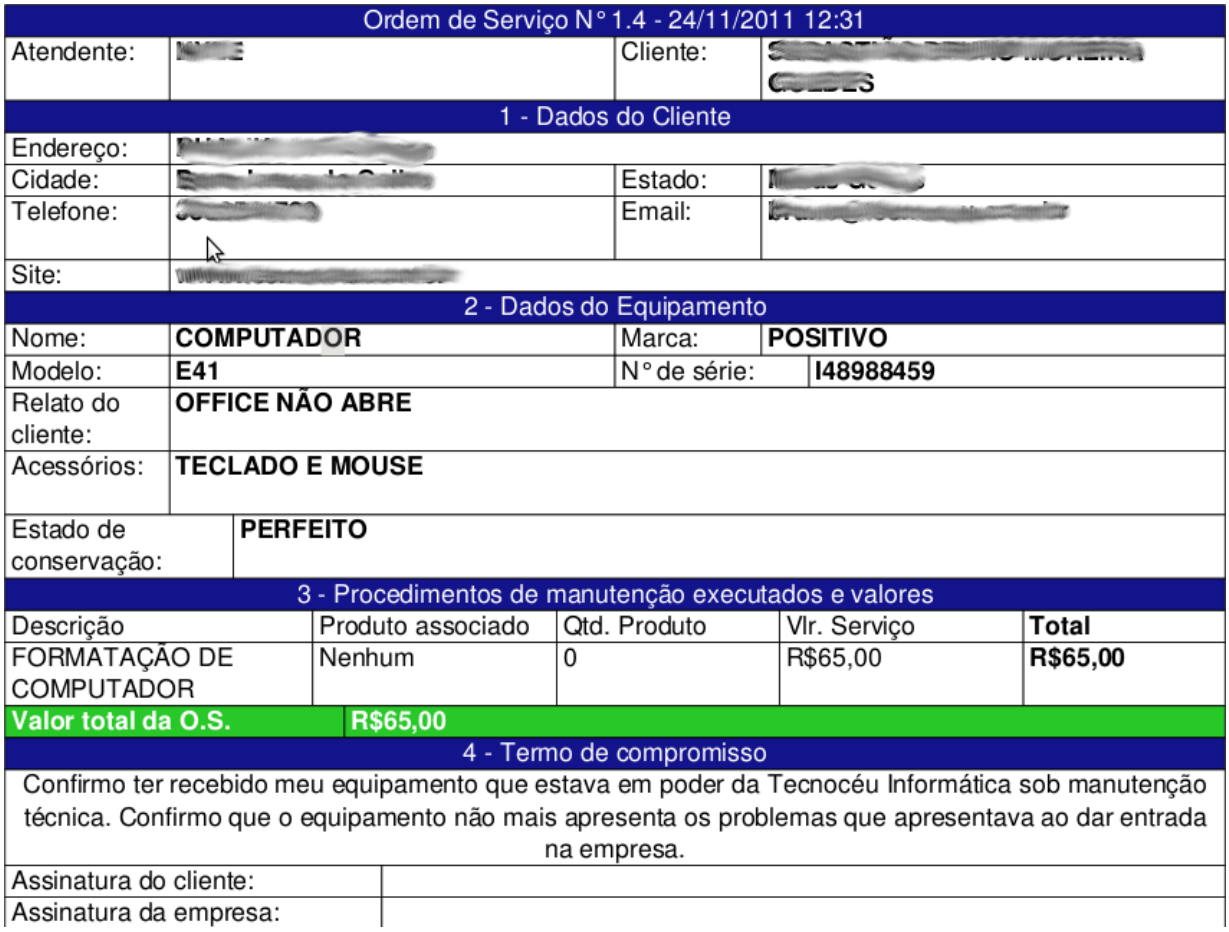
Figura 25: Tela de abertura de ordens de serviço

A Figura 25 mostra a tela de abertura de ordens de serviço, que deve ser acessada principalmente pelos atendentes. Nesta tela são informados diversos dados relvantes a respeito da ordem de serviço que se deseja abrir, como os dados relacionados ao equipamento referente a ordem de serviço, o cliente que pretende contratar o serviço de manutenção, etc.

Após a abertura da ordem de serviço nesta tela, o software possibilita a impressão de um relatório de abertura de ordem de serviço, que pode ser visto na Figura 26.

Ordem de Serviço N° 1.15 - 08/12/2011 09:43			
Atendente:	[redacted]	Cliente:	[redacted]
1 - Dados do Cliente			
Endereço:	[redacted]		
Cidade:	[redacted]	Estado:	[redacted]
Telefone:	[redacted]	Email:	[redacted]
Site:	[redacted]		
2 - Dados do Equipamento			
Nome:	**NOTEBOOK**	Marca:	**KENNEX**
Modelo:	**U50SA**	N° de série:	**2394820934**
Relato do cliente:	**TELA COM RISCOS HORIZONTAIS**		
Acessórios:	**MOUSE USB**		
Estado de conservação:	**TAMPA ARANHADA**		
3 - Dados adicionais			
Senha para acompanhamento on-line:	**LihYseba.**		
4 - Termo de compromisso			
Confirmo deixar em poder da Tecnocéu Informática o equipamento descrito acima para análise técnica. Para acompanhar o andamento do serviço on-line, acesse o site: www.tecnoceu.com.b/os, digite o número da sua ordem de serviço e asenha acima.			
Assinatura do cliente:			
Assinatura da empresa:			

Figura 26: Exemplo de relatório de abertura de ordem de serviço gerado pelo software



Ordem de Serviço N° 1.4 - 24/11/2011 12:31			
Atendente:	[redacted]	Cliente:	[redacted]
1 - Dados do Cliente			
Endereço:	[redacted]		
Cidade:	[redacted]	Estado:	[redacted]
Telefone:	[redacted]	Email:	[redacted]
Site:	[redacted]		
2 - Dados do Equipamento			
Nome:	**COMPUTADOR**	Marca:	**POSITIVO**
Modelo:	**E41**	N° de série:	**I48988459**
Relato do cliente:	**OFFICE NÃO ABRE**		
Acessórios:	**TECLADO E MOUSE**		
Estado de conservação:	**PERFEITO**		

3 - Procedimentos de manutenção executados e valores				
Descrição	Produto associado	Qtd. Produto	Vlr. Serviço	**Total**
FORMATAÇÃO DE COMPUTADOR	Nenhum	0	R$65,00	**R$65,00**
Valor total da O.S.	**R$65,00**			

4 - Termo de compromisso	
Confirmo ter recebido meu equipamento que estava em poder da Tecnocéu Informática sob manutenção técnica. Confirmo que o equipamento não mais apresenta os problemas que apresentava ao dar entrada na empresa.	
Assinatura do cliente:	
Assinatura da empresa:	

Figura 27: Exemplo de relatório de fechamento de ordem de serviço gerado pelo software

A Figura 27 mostra um relatório de fechamento de ordem de serviço, gerado no momento do fechamento da ordem de serviço (neste momento a ordem de serviço assume o status “Equipamento Entregue”).

Este módulo possui uma interface através da qual o cliente pode interagir com a empresa que realiza a manutenção de seu equipamento. Esta interface foi desenvolvida de modo que o seu desempenho em aparelhos móveis fosse o melhor possível, por isso ela aproveita ao máximo o espaço disponível na tela. As figuras 28, 29 e 30 mostram estas telas sendo acessadas através de um celular, comprovando assim a possibilidade de que o software seja acessado por dispositivos móveis.

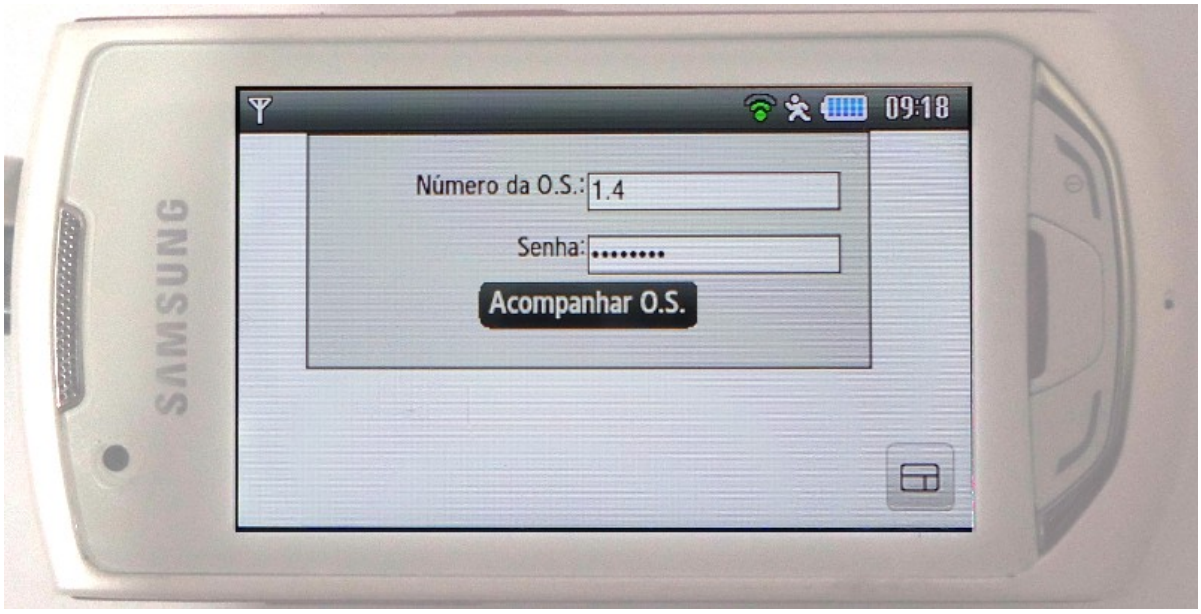


Figura 28: Tela de login que possibilita ao cliente acompanhar sua ordem de serviço

A Figura 28 mostra a tela de login que possibilita ao cliente acompanhar a ordem de serviço por ele contratada.

É importante notar que embora esta tela possa ser acessada normalmente por um navegador comum, ela foi desenvolvida para que se adaptasse perfeitamente a dispositivos móveis, tais como celulares, smartphones, tablets, etc.

Desse modo, o cliente pode acompanhar a ordem de serviço por ele contratada de onde quer que ele esteja, desde que ele possua um aparelho móvel com acesso a internet, o que gera uma enorme praticidade e conforto.

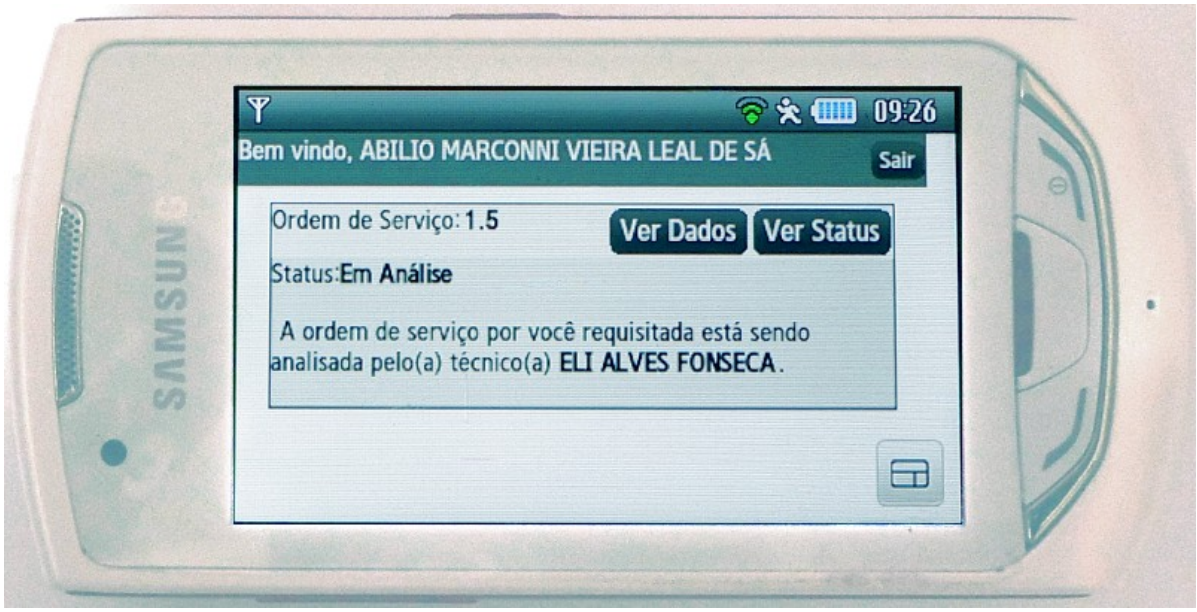


Figura 29: Tela de exibição de status da ordem de serviço - Status "Aguardando"

Na tela de exibição de status da ordem de serviço, mostrada na Figura 29, o cliente pode verificar o status da ordem de serviço por ele contratada. Alguns status específicos permitem ao

cliente interagir de alguma forma com a empresa que realiza o serviço de manutenção. Este conceito fica mais claro com a verificação da Figura 30.

Figura 30: Tela que permite ao cliente aprovar ou rejeitar o orçamento feito pelo técnico

Na Figura 30, é exibido ao cliente o status de sua ordem de serviço, que neste caso é “Orçamento Pronto”. Este status significa que o técnico responsável pela ordem de serviço já identificou os procedimentos de reparo necessários para o equipamento do cliente.

Esta tela permite ao cliente interagir, conforme pode ser visto, aprovando ou rejeitando o orçamento feito pelo técnico.

4.12 A realização dos testes

Para a realização dos testes, foi feita a instalação do software na empresa Tecnocéu Informática, bem como o treinamento dos funcionários da mesma para utilização daquele. Foi feito o cadastramento dos produtos, serviços, peças de reparos, técnicos, atendentes e os dispositivos que os técnicos estavam aptos a reparar.

Segundo os técnicos da empresa, houve grande melhora na organização da bancada, bem como na obtenção de informações específicas sobre determinado equipamento, conforme pode ser visto no depoimento do técnico Eli Alves Fonseca:

“Pude notar, que através do uso do software Smart Mill, conseguimos organizar melhor a bancada. Pessoalmente, gostei do fato de o software calcular automaticamente as prioridades das ordens de serviço, de modo que eu pude dar mais ênfase em ordens de serviço que tinham maior prioridade, o que diminuiu consideravelmente as reclamações dos clientes a respeito de possíveis

atrasos”

Os benefícios citados pelo técnico em seu depoimento são fornecidos por algumas das principais funcionalidades do módulo de Ordens de Serviço e do módulo Estoque, como a possibilidade de cadastrar os produtos da empresa, os equipamentos que podem ser reparados, e o cálculo automático do valor da ordem de serviço.

Também convém apresentar aqui o depoimento do técnico Ronaldo Vieira da Silva:

“O que eu mais gostei no software, foi a não necessidade de comunicação verbal entre os diversos setores de atendimento da empresa, pois uma ordem de serviço que é aberta pelo atendente, automaticamente cai na fila de espera dos nossos computadores, sem a necessidade de que a gente verifique as ordens de serviço em aberto. E melhor ainda, quando concluíamos uma análise de um determinado dispositivo, tínhamos que ficar cobrando dos atendentes para que os mesmos entrassem em contato com o cliente indagando-o se ele aprovava ou não o orçamento.

Com a implantação do Smart Mill, após eu ter concluído um orçamento o mesmo sai da tela dos técnicos e retorna para a tela dos atendentes, que por sua vez, podem entrar em contato com o cliente, ou o mesmo cliente pode visualizar o orçamento on-line e aprová-lo ou rejeitá-lo, até mesmo através de um celular que tenha acesso a internet. Eu mesmo fiz um teste, e é muito simples de ser utilizado, economizando assim muito tempo e diminuindo os erros que aconteciam anteriormente de que se um responsável pela sua tarefa falhasse, todo o resto falhava.

Com o Smart Mill, se eu concluí o reparo, consigo mostrar ao meu patrão que se o equipamento não for entregue, isso se deve ao fato de que o cliente não deu seu parecer, ou o atendente pode não estar executando suas tarefas.”

O que o técnico exemplificou em seu comentário foi o fato de o software oferecer uma interface através da qual o contratante possa interagir com a empresa que realiza o serviço de manutenção de seu equipamento.

Os atendentes da empresa também notaram considerável melhoria no que tange a obtenção de informações (como qual técnico está executando qual ordem de serviço, etc) solicitadas pelos clientes por telefone ou e-mail. Um dos atendentes, o senhor Sebastião Bruno Moreira Guedes, concordou em também oferecer um depoimento para ser aqui apresentado:

“Eu gostei muito da possibilidade do cliente aprovar a ordem de serviço on-line, ou até mesmo imprimi-la de sua residência ou seu local de trabalho, pois economiza o tempo que gastávamos contatando o mesmo, tendo em vista que o serviço as vezes aperta, e a consequência era que acabávamos acumulando ordens de serviço, pois não podiam concluir tais ordens sem o consentimento do cliente.

Também gostei do fato de podermos saber qual técnico é responsável por determinada ordem

de serviço, pois essa informação geralmente é solicitada pelo cliente via telefone ou e-mail, e com o Smart Mill, podemos atendê-lo prontamente."

O técnico em redes da empresa Kyle Lopes de Mattos Silva, também deu seu depoimento sobre o software:

"Eu sou um grande entusiasta e usuário do sistema GNU/Linux. O software que utilizávamos para automação de ordens de serviço na empresa possuía suporte apenas para o sistema Microsoft Windows, o que não me agradava muito.

Com a possibilidade de testarmos um software que roda na plataforma Linux, como é o caso do Smart Mill, eu pude providenciar um servidor Linux Debian para que o software fosse instalado.

O fato de estarmos utilizando uma plataforma segura e open-source, me deixa mais tranquilo do que quando utilizávamos uma plataforma sem muita segurança, como é o caso do Windows."

O que o técnico Kyle Lopes de Mattos Silva apreciou no software foi basicamente o fato de este ser independente de plataforma, o que permitiu que a empresa em que foram realizados os testes utilizasse-o na plataforma de sua preferência, que no caso, era o sistema operacional GNU/Linux.

Conforme pode ser visto, foi alcançado o objetivo de desenvolver um software cuja finalidade é de preencher as demandas não supridas em softwares que atendem as empresas de pequeno porte de manutenção de equipamentos, como é o caso da Tecnocéu Informática, empresa que se dispôs a testar o software.

5 CONCLUSÃO

Os objetivos propostos neste trabalho foram atingidos. O software produzido é capaz de sanar as dificuldades encontradas nos softwares analisados.

- O software é multiplataforma: O software desenvolvido pelo presente trabalho é totalmente independente de plataforma. Por ter sido desenvolvido na linguagem de programação PHP, ele pode ser instalado em qualquer dos sistemas operacionais que vão desde o Unix incluindo o Linux, FreeBSD, Solaris, Windows e até o Mac OS X (conforme visto no referencial teórico, o PHP funciona nestas plataformas, segundo Lerdorf, Tatroe e Intyre (2006, p. 1)). Também é importante que se note que o software pode ser acessado em qualquer plataforma que possua algum tipo de acesso a internet via browsers.
- O software é acessível por dispositivos móveis (celulares smartphones, tablets): O software é comprovadamente acessível através de dispositivos móveis (vide Figuras 40, 41 e 42, que representam casos reais de uso).
- O software oferece uma interface para que o contratante (cliente) possa acompanhar o status da ordem de serviço: O software possui uma interface desenvolvida especificamente para este fim. No desenvolvimento desta interface, deu-se grande ênfase na acessibilidade através de dispositivos móveis, de modo que a interface otimiza a exibição de informações para telas pequenas (as Figuras 40, 41 e 42 mostram o software sendo acessado através de um celular cuja tela tem 3 polegadas). Nos testes realizados na empresa Tecnocéu Informática, os clientes que se dispuseram a fazer uso deste recurso do software, afirmaram que a interface gera um grande conforto, pois permite que eles possam interagir com a empresa e obter informações relevantes a respeito da ordem de serviço por eles contratadas sem sequer precisar se deslocar, ou ao menos entrar em contato com a empresa. Procurou-se também tornar o uso desta interface simples, de modo que todas as ações podem ser acionadas através de botões (no caso de telas touch screen, o cliente pode fazer estes acionamentos simplesmente tocando na tela, sem necessidade de uso do teclado).

Estes fatos levam a concluir que o uso do software aqui apresentado é viável em pequenas empresas de manutenção, produzindo vantagens não só para a empresa que utiliza o software, mas também para seus funcionários e para seus clientes principalmente.

A importância do presente trabalho reside no fato de que ele supre as demandas aqui identificadas no ramo de automação do controle de ordens de serviço, disponibilizando para as empresas um software inovador que pode ser utilizado sem a aquisição de licenças.

O software aqui apresentado está disponível para download na página

http://www.tecnoceu.com.br/downloads/smartmill.tar.gz.

O presente trabalho trouxe maior compreensão no que diz respeito a automação comercial e controle de ordens de serviço, e o uso de softwares que auxiliam neste contexto.

6 TRABALHOS FUTUROS

- Analisar a viabilidade de integração com a tecnologia RFID (veja seção 2.4 do referencial teórico).
- Analisar a viabilidade de implantação de um sistema de pagamento de débitos on-line, tal como o PagSeguro, que permita ao cliente resgatar seu débito junto a empresa através de dispositivos móveis e efetuar o pagamento do mesmo através de cartão de crédito ou boletos bancários.
- Estudar a possibilidade de inclusão do conjunto de recursos e tecnologias que permitam ao cliente acompanhar a execução da ordem de serviço por câmeras de vídeo.
- Adicionar um recurso que permita ao software contatar o cliente via e-mail e SMS quando o orçamento pertinente a ordem de serviço por ele contratada tiver sido concluído.
- Adicionar controle financeiro no software, permitindo que este controle contas a pagar e contas a receber.

7 REFERÊNCIAS BIBLIOGRÁFICAS

BIGDOLI, Hossein. **The Internet Encyclopedia.** 1 ed. Hoboken: John Wiley & Sons, 2004.

BIRREL, N.D. **A Pratical Handbook for Software Development.** 1 ed. Cambridge University Press, 1985.

DEITEL, Paul; DEITEL, Harvey; **Java:** Como Programar. 8 ed. São Paulo: Pearson, 2010.

DHILLON, B.S. **Engeneering Maintenance:** A Modern Approach. 1 ed. Washington D.C: CRC Press.

DKAL – AUTOMAÇÂO COMERCIAL. **O que é automação comercial.** Disponível em: http://www.automacaocomercial.com.br/index.php, acesso em 14 de outubro de 2011.

FLANAGAN, David. **Javascript:** O guia definitivo. 4 ed. Porto Alegre: Bookman, 2002.

HARRINGTON, Jan L. **SQL Clearly Explained.** 2 ed. San Francisco: Elsevier Science: 2003.

KEDAR, Seema V. **Programming Paradigms And Methodology.** 3 ed. India Technical Publications Pune, 2007.

LERDORF, Rasmus; TATROE, Kevin; MACLNTYRE, Peter; **Programming PHP.** 2 ed. Sebastopol: O'Reilly, 2006.

MEDINA, Marco; FERTIG, Cristina; **Algoritmos e Programação – Teoria e Prática.** Novatec,

2005.

MILLER, Frederic P; VANDOME, Agnes F; MCBREWSTER, John; **Model-View-Controller.** VDM Publishing House Ltd., 2010**.**

GRESCHWINDE, EWALD; HANS-JÜRGEN, Schönig; **PostgreSQL:** Developers Handbook. Sams, 2002.

MORIMOTO, Carlos Eduardo. **Servidores Linux:** guia prático**.** 1 ed. Porto Alegre: Sul Editores, 2009.

POO, Dany; KIONG, Derek; ASHOK, Swarnalatha; **Object-Oriented Programming and Java.** 2 ed. Londres: Springer, 2008.

RAMEZ, Elmasri; NAVATHE, Shamkant B; **Sistemas de Banco de Dados.** 4 ed. São Paulo: Pearson, 2006.

SAP. **Ordem de serviço.** Disponível em:http://help.sap.com/saphelp_40b/helpdata/pt/60/8dac0f7ad211d189380000e8284931/content.htm, acesso em 25 de outubro de 2011.

SAUR, Ricardo Adolfo de Campos. **A Tecnologia da Informação na Reforma do Estado:** Considerações sobre a prestação de serviços de informática na área pública. Brasília: 1997.

SHALLOWAY, Allan; TROTT, James R. **Explicando Padrões de Proeto:** Uma Nova Perspectiva em Projeto Orientado a Objeto. 1 ed. São Paulo: BOOKMAN, 2002.

SILVEIRA, Sérgio Amadeu da. **Software livre – A luta pela liberdade do conhecimento.** 1 ed. 2004.

SIMBER, **RFID:** O que é? Disponível em http://www.simber.com.br/erp/o-que-e-rfid, acesso em 25 de novembro de 2011.

8 ANEXOS

8.1 Visão geral do código do software desenvolvido

Neste tópico será feito um estudo mais profundo sobre a estrutura de arquivos dos módulos do software e também sobre alguns aspectos do código fonte do software.

8.1.1 Módulo Principal

O módulo principal é composto por quatro arquivos: “modelPrincipal.php”, “viewPrincipal.php”, “controlPrincipal.php” e “configPrincipal.php” , seguindo o padrão MVC, conforme pode ser visto na Figura 31.

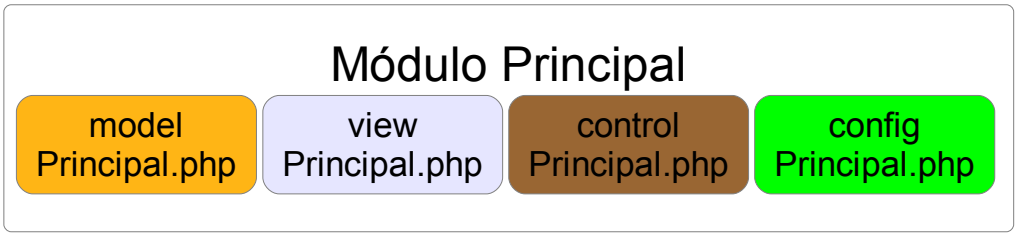


Figura 31: Estrutura de arquivos do módulo principal

O arquivo “modelPrincipal.php” contem a classe “Model()”, que seguindo o padrão MVC, representa a camada modelo neste módulo. Dessa classe descendem todas as outras classes modelo dos demais módulos. Esta classe estabelece as funções principais, que são usadas nas classes filhas, e inicializa o objeto “connector”, que estabelece a conexão com o banco de dados.

Dentre as propriedades e funções existentes nesta classe, destacam-se as seguintes:

1. Propriedade “connector”: estabelece conexão com o banco de dados;

2. Função “Model()”: estabelece a inicialização padrão de todas as classes modelo de todos os módulos do software;

3. Função “createTables()”: cria as tabelas do banco de dados referentes ao módulo a que pertence a classe, caso elas não existam.

4. Função “setLastError()”: é utilizada em funções internas da classe para informar algum possível erro, no caso desta função ter retorno booleano.

5. Função “getLastError()”: é utilizada em funções externas a classe, e retorna o último erro informado pela função “setLastError()”,.

O arquivo “viewPrincipal.php” representa a camada de apresentação. No caso deste módulo, esta

classe gera o menu principal e a tela principal do software.

O arquivo “controlPrincipal.php” representa a camada de regras de negócio, seguindo o padrão MVC. Neste módulo, este arquivo não contem classes de grande relevância, não sendo portanto, necessário que sejam aqui apresentadas.

O arquivo “configPrincipal.php” contem a classe “PrincipalConfig()” (que deriva da classe “Config()”, também definida neste arquivo), que possui funções que permitem a comunicação com os demais módulos do sistema, através da geração de código dinâmico (um recurso somente disponível em linguagens interpretadas ou semi-interpretadas, como é o caso do PHP).

A existência desta classe e de suas funcionalidades permite que o software seja estendido com a simples instalação de novos módulos no diretório “modulos”. Por exemplo, caso se algum usuário precisasse importar dados de um software externo, bastaria que o módulo para este fim existisse, e que o usuário instalasse este módulo no software, não sendo portanto, necessária a reinstalação do software por completo.

É importante ressaltar que esta é a exata finalidade deste mesmo arquivo nos demais módulos. Portanto, seria redundância descrevê-lo nos demais módulos. Doravante, serão apresentados apenas os arquivos “control+Nome do módulo.php”, “model+”Nome do módulo.php”, “view+”Nome do módulo.php” e qualquer outro arquivo relevante dos módulos restantes.

8.1.2 O módulo Servidores

A estrutura de arquivos deste módulo é similar a estrutura do módulo principal, contendo os arquivos “modelServidores.php”, “viewServidores.php”, “controlServidores.php” e “configServidores.php”, conforme pode ser visto na Figura 32.

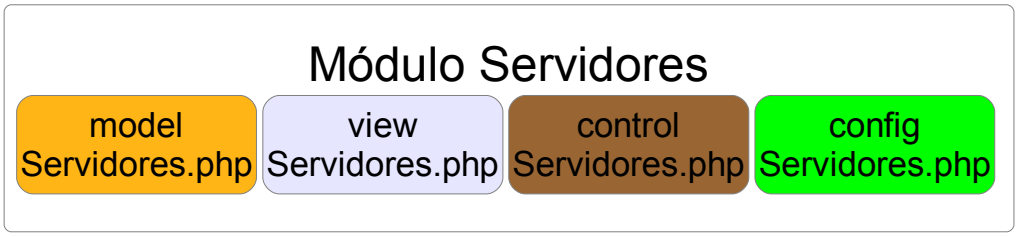


Figura 32: Estrutura de arquivos do módulo Servidores

O arquivo “modelServidores.php” contem a classe “ServidoresModel()”, que deriva da classe “Model()” definida no módulo principal. Ela herda as funções já estabelecidas na sua classe pai.

A classe “ServidoresModel()” foi desenvolvida para criar e manipular a tabela “servidores”, cuja função já foi explicada.

Dentre as propriedades e funções existentes nesta classe, destacam-se:

1. “cadastraServidorProprio()”: esta função é utilizada apenas na configuração inicial do sistema, para definir um nome para o servidor que executa o sistema.

2. “pesquisaServidores()”: esta função recebe uma cadeia de caracteres como parâmetro e pesquisa, dentro da tabela servidores, por servidores cujo nome que contenha a cadeia de caracteres passada por parâmetro.

O arquivo “controlServidores.php”, seguindo o padrão MVC, define as regras de negócio referentes a seguinte operação:

1. frmCadastraServidorProprio: Esta operação refere-se ao formulário apresentado nas configurações iniciais do sistema, para definição do nome do servidor que executa o software. A tela referente a este formulário pode ser vista na Figura 6.

O arquivo “viewServidores.php”, neste módulo, é responsável apenas por apresentar o formulário para definição do nome do servidor que roda o software nas configurações iniciais do software.

8.1.3 O módulo Usuários

A estrutura de arquivos deste módulo é similar a estrutura do módulo principal, contendo os arquivos “modelUsuarios.php”, “viewUsuaios.php”, “controlUsuarios.php” e “configUsuarios.php”, e está exibida na Figura 33.

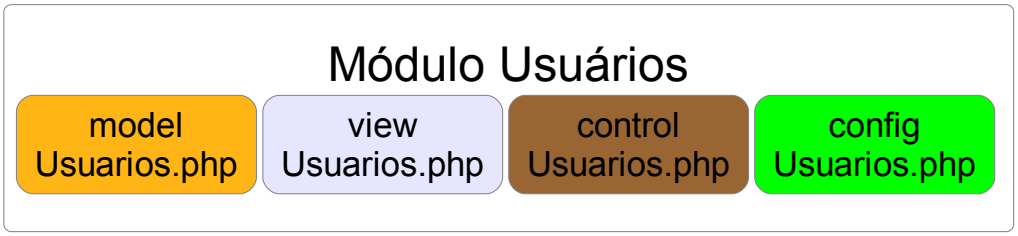


Figura 33: Estrutura de arquivos do módulo Usuários

O arquivo “modelUsuarios.php” contem a classe “UsuariosModel()”, que deriva da classe “Model()” definida no módulo principal. Ela herda as funções já estabelecidas na sua classe pai.

A classe “UsuariosModel()” foi desenvolvida para criar e manipular a tabela “grupos_usuarios”, “permicoes_grupos_usuarios” e a tabela “usuarios”.

A tabela “grupos_usuarios” armazena dados dos grupos de usuários cadastrados, e contém os seguintes campos:

1. servidor: Este campo é uma chave estrangeira que referencia o servidor no qual o grupo foi cadastrado.

2. id: Este campo é um código numérico que identifica cada grupo dentro de um dado servidor.

Ele é preenchido automaticamente pelo software, não sendo em situações normais, necessário que o usuário se preocupe ele.

3. nome: Este campo registra o nome do grupo.

4. administrador: Este campo armazena um valor verdadeiro quando o grupo cadastrado é administrador, e um valor falso quando não o é. Um usuário que pertença ao grupo administrador possui permissão para realizar qualquer tipo de operação dentro do sistema, exceto excluir a si mesmo.

A tabela “permicoes_grupos_usuarios” armazena as permissões dada a determinado grupo de usuários, e contém os seguintes campos:

1. servidor: Referencia o servidor no qual a permissão foi dada.

2. grupo_servidor: Juntamente com o campo “grupo_id”, referencia o grupo ao qual q permissão foi dada.

3. grupo_id: Juntamente com o campo “grupo_servidor”, referencia o grupo ao qual q permissão foi dada.

4. modulo: Armazena o nome do módulo ao qual a permissão garante acesso a determinada função.

5. nome_permissao: Armazena o nome da permissão.

6. valor_permissao: Armazena o valor da permissão, que pode ser um número, uma cadeia de caracteres e mesmo valores booleanos.

A tabela “usuarios” possui os seguintes campos:

1. servidor: Referencia o servidor no qual o usuário foi cadastrado.

2. id: Armazena o identificador do usuário dentro do servidor no qual ele foi cadastrado.

3. nome: Armazena o nome do usuário.

4. login: Armazena o login do usuário, que deve ser único.

5. senha: Armazena um hash de 64 bytes contendo a senha criptografada do usuário. O método de criptografia utilizado neste caso é o SHA256, utilizando-se um salt de 3 bytes aleatoriamente gerado pelo software.

6. salt: Armazena o salt de 3 bytes gerado aleatoriamente para aumentar a segurança da criptografia.

Dentre as propriedades e funções existentes na classe “UsuariosModel()”, destacam-se:

1. “cadastraAdministrador()”: esta função é utilizada apenas na configuração inicial do sistema, para cadastrar um usuário para acesso inicial ao sistema. Também nesta função é cadastrado um grupo administrador, caso não exista um.

2. “cadastraGrupo()”: esta função é utilizada para cadastrar grupos de usuários. É importante

que todos os grupos cadastrados tenham nomes diferentes. O módulo permite que haja mais de um grupo administrador, mas isto não é aconselhável, pois na maioria dos casos poderia gerar redundância.

3. “editaGrupo()”: esta função é utilizada para realizar a edição dos dados de um grupo cadastrado. É possível alterar o nome do grupo e redefinir se o grupo é ou não administrador.

4. “editaUsuario()”: esta função é utilizada para realizar a edição dos dados de um usuário cadastrado.

5. “excluiGrupo()”: esta função é utilizada para excluir grupos de usuários. Se o existirem usuários cadastrados que pertençam ao grupo em exclusão, o software informa esta existência e não permite a exclusão do grupo.

6. “excluiUsuario()”: esta função exclui um ou mais usuários de uma só vez. Portanto, recebe como parâmetro uma lista (array) contendo as chaves primárias dos usuários para exclusão. Para realizar a exclusão, o módulo Usuários percorre a lista de módulos instalados no software, procurando por restrições que impeçam a exclusão de um dado usuário. Caso estas restrições não existam, o usuário é excluído.

7. “adicionaPermissao()”: esta função adiciona uma nova permissão para determinado grupo de usuários. É importante que se note que só faria sentido adicionar determinada permissão para um grupo de usuários se este não for administrador. Do contrário, o grupo já teria todas as permissões possíveis, como já foi explicado.

8. “removePermissao()”: esta função remove determinada permissão dada a um determinado grupo de usuários.

O arquivo “controlUsuarios.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraUsuarioAdministrador: Esta operação somente é requisitada nas configurações iniciais do sistema, onde é apresentado um formulário para que se efetue o cadastramento de um usuário para acesso inicial ao sistema. Esta operação chama a função “cadastraAdministrador()” da classe “UsuariosModel()” para realizar o cadastramento do usuário no banco de dados..

2. usuarioLogin: Esta é a operação que permite que os usuários façam login no sistema..

3. visualizadorUsuariosSearch: Esta operação é requisitada na tela de visualização de usuários, quando se pesquisa por determinado usuário. O resultado da pesquisa é dividido em páginas, para evitar que grandes quantidades de dados sejam enviados ao navegador.

4. frmCadastraUsuario: Esta operação permite que novos usuários sejam cadastrados no sistema.

5. frmCadastraGrupoUsuario: Esta operação permite que novos grupos de usuários sejam

cadastrados, podendo ou não ser grupos administradores.

6. frmCadastraPermissaoGrupoUsuario: Esta operação permite que se possa dar permissões para um grupo de usuários possa realizar determinado tipo de operação em determinado módulo do software.

A existência de operações tais como “frmCadastraGrupoUsuario”, que permite o cadastramento de novos grupos de usuários, denota a existência de operações que permitam que um determinado grupo possa também ter seus dados alterados ou mesmo que seja excluído. Portanto, estas operações podem e serão omitidas, de acordo com o escopo do presente trabalho. Somente serão apresentadas as operações de maior relevância.

O arquivo “viewUsuarios.php”, neste módulo, é responsável apenas por apresentar formulários que permitam o cadastramento do usuário inicial do sistema (nas configurações iniciais), formulários de cadastramento de usuários, grupos de usuários, e formulários de adição de permissões para usuários. Também este arquivo é responsável por apresentar o formulário de login.

8.1.4 O módulo Cadastros Geográficos

O módulo Cadastros Geográficos tem por finalidades :

- Permitir que se possa cadastrar países;
- Permitir que se possa editar dados de países cadastrados;
- Permitir a exclusão de países cadastrados;
- Permitir que se possa cadastrar estados;
- Permitir que se possa editar dados de estados cadastrados;
- Permitir a exclusão de estados cadastrados;
- Permitir que se possa cadastrar Cidades;
- Permitir que se possa editar cidades cadastradas;
- Permitir a exclusão de cidades cadastradas.

O desenvolvimento deste módulo foi de grande importância, pois existem outros módulos que dependem dos cadastros geográficos. O módulo Pessoas é um deles (pois uma pessoa cadastrada pode pertencer a uma cidade, que deve ser cadastrada previamente no módulo Cadastros Geográficos).

A estrutura de arquivos do módulo Cadastros Geográficos é similar a do módulo principal, conforme pode ser verificado na Figura 34:

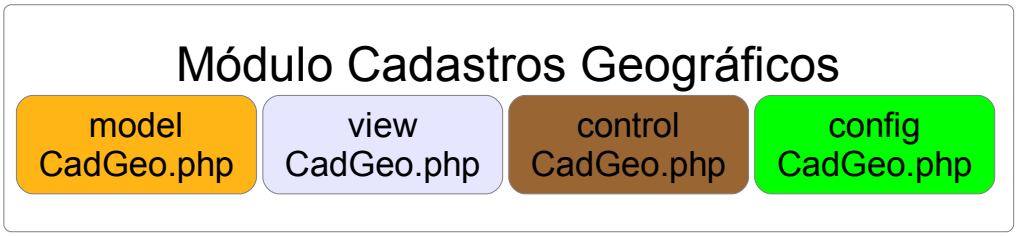


Figura 34: Estrutura de arquivos do módulo Cadastros Geográficos

O arquivo “modelCadGeo.php” contem a classe “CadastrosGeograficosModel()”, que deriva da classe “Model()” definida no módulo principal.

Essa classe é responsável por criar e manipular as tabelas “paises”, “estados” e “cidades”. A tabela “paises” armazena os países cadastrados no software, e contém os seguintes campos:

1. servidor: Este campo referencia o servidor no qual o país foi cadastrado.

2. id: É um código numérico que identifica o país dentro do servidor no qual ele foi cadastrado.

3. nome: Armazena o nome do país cadastrado.

4. sigla: Armazena a sigla do país cadastrado.

5. A tabela “estados” armazena os estados cadastrados no software, e possui os seguintes campos:

1. servidor: Este campo referencia o servidor no qual o estado foi cadastrado.

2. id: É um código numérico que identifica o estado dentro do servidor no qual ele foi cadastrado.

3. pais_servidor: Juntamente com o campo “pais_id”, referencia o país ao qual pertence o estado cadastrado.

4. pais_id: Juntamente com o campo “pais_servidor”, referencia o país ao qual pertence o estado cadastrado.

5. nome: Armazena o nome do estado cadastrado.

6. sigla: Armazena a sigla do estado cadastrado.

A tabela “cidades” armazena as cidades cadastradas no software, e possui os seguintes campos:

1. servidor: Este campo referencia o servidor no qual a cidade foi cadastrada.

2. id: É um código numérico que identifica a cidade dentro do servidor no qual ele foi cadastrada.

3. estado_servidor: Juntamente com o campo “estado_id”, referencia o estado ao qual pertence a cidade cadastrada.

4. estado_id: Juntamente com o campo “estado_servidor”, referencia o estado ao qual pertence a cidade cadastrada.

5. nome: Armazena o nome da cidade cadastrada.

Dentre as propriedades e funções existentes nesta classe, destacam-se as seguintes:

1. Função “cadastraPais()”: permite o cadastramento de países;
2. Função “pesquisaPaises()”: pesquisa dentre os países cadastrados no software;
3. Função “cadastraEstado()”: permite o cadastramento de estados;
4. Função “pesquisaEstados()”: pesquisa dentre os estados cadastrados no software;
5. Função “cadastraCidade()”: permite o cadastramento de cidades;
6. Função “pesquisaCidades()”: pesquisa dentre as cidades cadastradas no software.

O arquivo “controlCadGeo.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraCidade: Esta operação é requisitada através do formulário de cadastramento de cidades.
2. frmCadastraEstado: Esta operação é requisitada através do formulário de cadastramento de estados.
3. frmCadastraPais: Esta operação é requisitada através do formulário de cadastramento de países..

O arquivo “viewCadGeo.php” representa a camada de apresentação, e tem por finalidade apresentar formulários que permitam que o usuário possa realizar as diversas operações oferecidas por esse módulo, que foram descritas no arquivo “controlCadGeo.php”.

8.1.5 O módulo Pessoas

A estrutura de arquivos do módulo Pessoas é similar a do módulo principal, conforme pode ser visto na Figura 35:

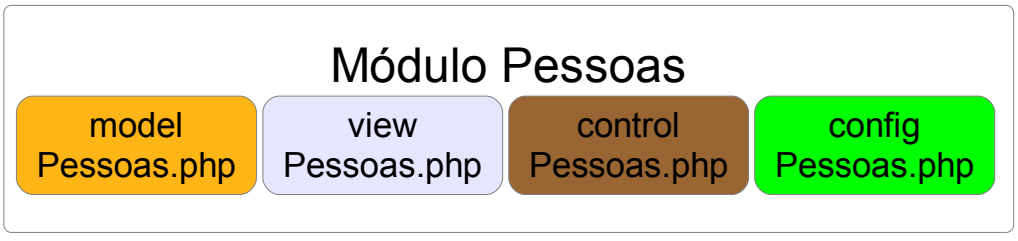


Figura 35: Estrutura de arquivos do módulo Pessoas

O arquivo “modelPessoas.php” contem a classe “PessoasModel()”, que deriva da classe “Model()” definida no módulo principal.

Essa classe é responsável por criar e manipular as tabelas “pessoas”, “telefones_pessoa”, “emails_pessoa” e “sites_pessoa”. A tabela “pessoas” armazena as pessoas cadastradas no software,

dentre os diversos campos que possui, devem ser aqui listados:

1. servidor: Este campo referencia o servidor no qual a pessoa foi cadastrada.

2. id: É um código numérico que identifica a pessoa dentro do servidor no qual ela foi cadastrada.

3. tipo: Armazena a informação que permite distinguir se a pessoa é física ou jurídica.

4. razao_social: Armazena a razão social da pessoa.

5. nome_fantasia: Armazena o nome fantasia da pessoa, caso essa seja jurídica.

6. cnpj_cpf: Armazena o CNPJ, caso a pessoa seja jurídica, ou o CPF caso a pessoa seja física.

7. inscricao_estadual: Armazena a inscrição estadual caso a pessoa seja jurídica.

8. data_cadastro: Armazena a data, hora, minuto e segundo em que a pessoa foi cadastrada no software.

Existem nessa tabela outros campos, tais como endereço, bairro, CEP, etc.

A tabela “telefones_pessoa” armazena os telefones que uma determinada pessoa possa ter, assim como a tabela “emails_pessoa” armazena os e-mails e a tabela “sites_pessoa” armazena os sites.

Dentre as propriedades e funções existentes na classe “PessoasModel()”, destacam-se as seguintes:

1. Função “cadastraPessoa()”: permite o cadastramento de pessoas;

2. Função “cadastraTelefonePessoa()”: permite o cadastramento de telefones para determinada pessoa;

3. Função “cadastraEmailPessoa()”: permite o cadastramento de e-mails para determinada pessoa;

4. Função “cadastraSitePessoa()”: permite o cadastramento de sites para determinada pessoa;

5. Função “pesquisaPessoas()”: permite que se pesquisa entre as pessoas cadastradas no software.

O arquivo “controlPessoas.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraPessoa: Esta operação é requisitada através do formulário de cadastramento de pessoas.

2. visualizadorPessoasSearch: Esta operação é requisitada através da tela de visualização e pesquisa de pessoas.

O arquivo “viewPessoas.php” representa a camada de apresentação, e tem por finalidade apresentar formulários que permitam que o usuário possa realizar as diversas operações oferecidas por esse módulo, que foram descritas no arquivo “controlPessoas.php”.

8.1.6 O módulo Empresas

A estrutura de arquivos do módulo Empresas é similar a do módulo principal, e pode ser vista na Figura 36:

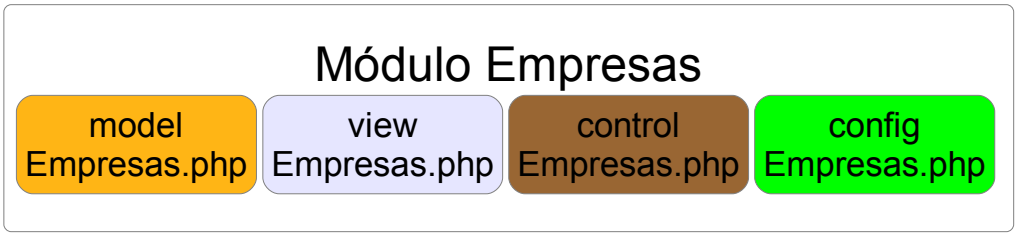


Figura 36: Estrutura de arquivos do módulo Empresas

O arquivo “modelEmpresas.php” contem a classe “EmpresasModel()”, que deriva da classe “Model()” definida no módulo principal, e dela herda propriedades e funções.

Essa classe é responsável por criar e manipular a tabela “empresa”, que armazena informações referentes a empresa que utiliza o software. Esta tabela contém apenas os campos que são suficientes para fazer referências a determinado registro da tabela pessoas, permitindo assim que se evite redundâncias no banco de dados. Eis os campos dessa tabela:

1. servidor: Este campo referencia o servidor no qual a empresa foi cadastrada.

2. id: É um código numérico que identifica a empresa dentro do servidor no qual ela foi cadastrada.

3. pessoa_servidor: Referencia, juntamente com o campo “pessoa_id”, a pessoa correspondente a empresa cadastrada.

4. pessoa_id: Referencia, juntamente com o campo “pessoa_servidor”, a pessoa correspondente a empresa cadastrada.

O arquivo “controlEmpresas.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraEmpresa: Esta operação é requisitada nas configurações iniciais do software, através do formulário de cadastramento da empresa usuária.

O arquivo “viewEmpresas.php” foi desenvolvido para representar a camada de apresentação, e tem por finalidade apresentar o formulário que permite o cadastramento da empresa usuária nas configurações iniciais do sistema.

8.1.7 O módulo Clientes

A estrutura de arquivos do módulo Clientes é similar a do módulo principal, conforme pode ser

visto na Figura 37:

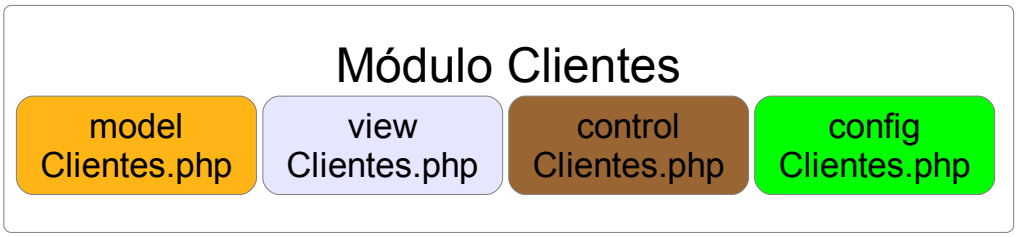


Figura 37: Estrutura de arquivos do módulo Clientes

O arquivo “modelClientes.php” contem a classe “ClientesModel()”, que deriva da classe “Model()” definida no módulo principal, e dela herda propriedades e funções.

Essa classe é responsável por criar e manipular a tabela “clientes”. Esta tabela contém apenas os campos que são suficientes para fazer referências a determinado registro da tabela pessoas. Eis os campos dessa tabela:

1. servidor: Este campo referencia o servidor no qual o cliente foi cadastrado.

2. id: É um código numérico que identifica o cliente dentro do servidor no qual ele foi cadastrado.

3. pessoa_servidor: Referencia, juntamente com o campo “pessoa_id”, a pessoa correspondente ao cliente cadastrado.

4. pessoa_id: Referencia, juntamente com o campo “pessoa_servidor”, a pessoa correspondente ao cliente cadastrado.

O arquivo “controlClientes.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraClientes: Esta operação é requisitada no formulário de cadastramento de clientes.

2. visualizadorClientesSearch: Esta operação é requisitada na tela de visualização e pesquisa de clientes.

O arquivo “viewClientes.php” foi desenvolvido para representar a camada de apresentação, e tem por finalidade apresentar a tela de visualização e pesquisa de clientes, o formulário de cadastramento de clientes, o formulário de edição de dados de clientes e a tela de importação de pessoas.

8.1.8 O módulo Fornecedores

A estrutura de arquivos do módulo Fornecedores é similar a do módulo principal, conforme pode ser verificado na Figura 38.

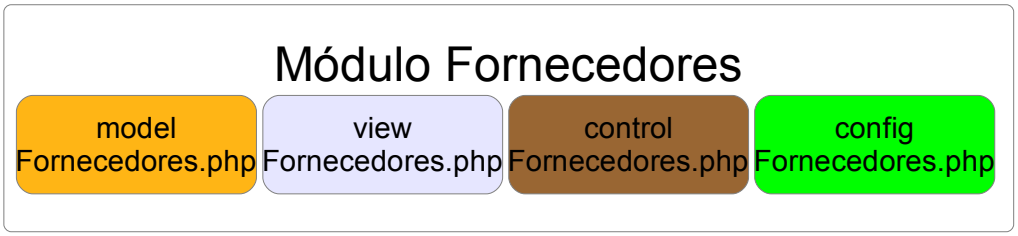


Figura 38: Estrutura de arquivos do módulo Fornecedores

O arquivo “modelFornecedores.php” contem a classe “FornecedoresModel()”. Essa classe é responsável por criar e manipular a tabela “fornecedores”. Essa tabela contém apenas os campos que são suficientes para fazer referências a determinado registro da tabela pessoas. Eis os campos dessa tabela:

1. servidor: Este campo referencia o servidor no qual o fornecedor foi cadastrado.

2. id: É um código numérico que identifica o fornecedor dentro do servidor no qual ele foi cadastrado.

3. pessoa_servidor: Referencia, juntamente com o campo “pessoa_id”, a pessoa correspondente ao fornecedor cadastrado.

4. pessoa_id: Referencia, juntamente com o campo “pessoa_servidor”, a pessoa correspondente ao fornecedor cadastrado.

O arquivo “controlFrnecedores.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraFornecedores: Esta operação é requisitada no formulário de cadastramento de fornecedores.

2. visualizadorFornecedoresSearch: Esta operação é requisitada na tela de visualização e pesquisa de fornecedores.

3. frmImportaPessoa: Esta operação é requisitada na tela de importação de pessoas, e permite fazer a importação de uma determinada pessoa que ainda não esteja associada a nenhum fornecedor cadastrado, isto é, uma pessoa que ainda não tenha sido cadastrada como fornecedor.

O arquivo “viewFornecedores.php” foi desenvolvido para representar a camada de apresentação, e tem por finalidade apresentar a tela de visualização e pesquisa de fornecedores, o formulário de cadastramento de fornecedores, o formulário de edição de dados de fornecedores e a tela de importação de pessoas.

8.1.9 O módulo Estoque

A estrutura de arquivos do módulo Estoque é similar a do módulo principal, conforme pode ser

visto na Figura 39.

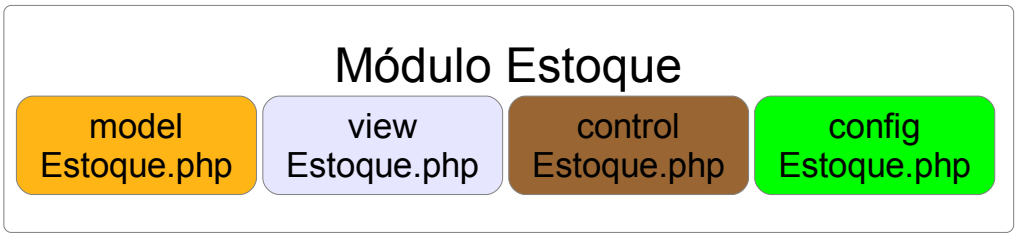


Figura 39: Estrutura de arquivos do módulo Estoque

O arquivo “modelEstoque.php” contem a classe “EstoqueModel()”. Essa classe é responsável por criar e manipular a tabela “produtos” e a tabela “movimentos_estoque”.

A tabela “produtos” foi desenvolvida para armazenar os produtos cadastrados no software. Eis seus campos:

- servidor: Este campo referencia o servidor no qual o produto foi cadastrado.
- id: É um código numérico que identifica o produto dentro do servidor no qual ele foi cadastrado.
- fornecedor_servidor: Referencia, juntamente com o campo “fornecedor_id”, o fornecedor associado ao produto.
- fornecedor_id: Referencia, juntamente com o campo “fornecedor_servidor”, o fornecedor associado ao produto.
- nome: Armazena o nome do produto.
- custo: Preço de custo do produto.
- lucro: Margem percentual de lucro incidente sobre o valor de custo.
- valor_venda: Valor de venda do produto, que é calculado a partir do valor de custo mais a margem percentual de lucro.
- saldo: Quantidade existente do produto no estoque.
- data_cadastro: Data em que foi cadastrado o produto.

O arquivo “controlEstoque.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraProdutos: Esta operação é requisitada no formulário de cadastramento de produtos.

2. visualizadorProdutosSearch: Esta operação é requisitada na tela de visualização e pesquisa de produtos.

3. frmAjustaEstoque: Esta operação é requisitada na tela de ajuste de estoque, onde se pode registrar entradas e saídas de produtos.

O arquivo “viewEstoque.php” foi desenvolvido para representar a camada de apresentação, e tem por finalidade apresentar a tela de visualização e pesquisa de produtos, o formulário de

cadastramento de produtos, o formulário de edição de dados de produtos e a tela de ajuste de estoque.

8.1.10 O módulo Funcionários

A estrutura de arquivos do módulo Funcionários é similar a do módulo principal, como pode ser visto na Figura 40.

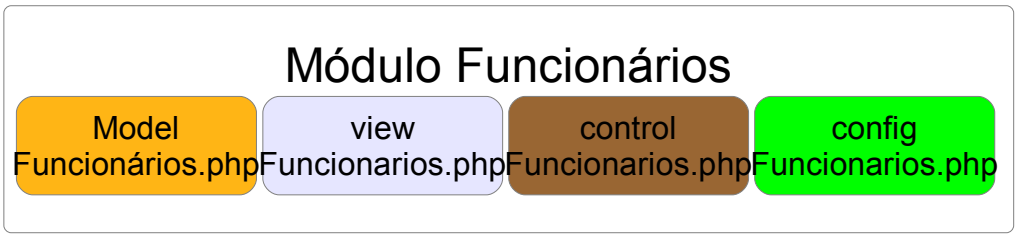


Figura 40: Estrutura de arquivos do módulo Funcionários

O arquivo “modelFuncionarios.php” contem a classe “FucionariosModel()”. Este módulo é responsável pela manipulação da tabela “funcionarios”. Esta tabela possui apenas os campos que são necessários para a referência a tabela “pessoas”, não sendo portanto, necessário que se apresente aqui seus campos.

O arquivo “controlFuncionarios.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

1. frmCadastraFuncionarios: Esta operação é requisitada no formulário de cadastramento de funcionários.

2. visualizadorFuncionariosSearch: Esta operação é requisitada na tela de visualização e pesquisa de funcionários.

3. frmImportaPessoa: Esta operação é requisitada na tela de importação de pessoas, onde se pode importar uma pessoa que ainda não tenha sido associada a nenhum funcionário.

O arquivo “viewFuncionarios.php” foi desenvolvido para representar a camada de apresentação, e tem por finalidade apresentar a tela de visualização e pesquisa de funcionários, o formulário de cadastramento de funcionários, o formulário de edição de dados de funcionários e a tela de importação de pessoas.

8.1.11 O módulo Ordens de Serviço

Este módulo possui uma estrutura de arquivos mais complexa que os demais módulos, que pode

ser vista na Figura 41:

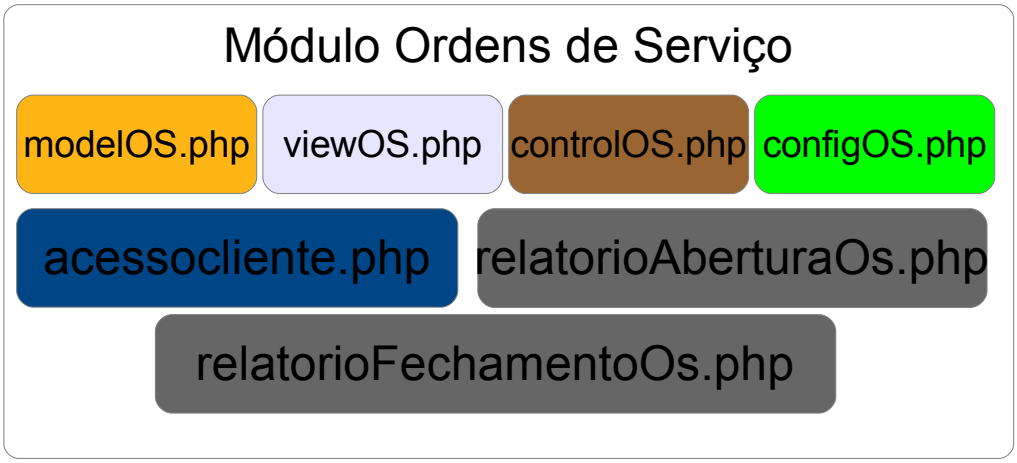


Figura 41: Estrutura de arquivos do módulo Ordens de Serviço

O arquivo “modelOS.php” contem a classe “OrdemServicoModel()”. Essa classe é responsável por criar e manipular as tabelas “ordens_servico”, “tecnicos_os”, “atendentes_os”, “equipamentos_os”, “procedimentos_os” e “procedimentos_os_executados”.

A tabela “ordens_servico” é responsável por armazenar todas as ordens de serviço. Eis seus campos:

1. servidor: Este campo referencia o servidor no qual a ordem de serviço foi aberta.

2. id: É um código numérico que identifica a ordem de serviço dentro do servidor no qual ela foi aberta.

3. atendente_servidor: Referencia, juntamente com o campo “atendente_id”, o atendente responsável pela abertura da ordem de serviço.

4. atendente_id: Referencia, juntamente com o campo “atendente_servidor”, o atendente responsável pela abertura da ordem de serviço.

5. cliente_servidor: Referencia, juntamente com o campo “cliente_id”, o cliente que requisitou a ordem de serviço.

6. cliente_id: Referencia, juntamente com o campo “cliente_servidor”, o cliente que requisitou a ordem de serviço.

7. tecnico_servidor: Referencia, juntamente com o campo “tecnico_id”, o técnico que assumiu e executou os serviços de manutenção referentes a ordem de serviço.

8. tecnico_id: Referencia, juntamente com o campo “tecnico_servidor”, o técnico que assumiu e executou os serviços de manutenção referentes a ordem de serviço.

9. equipamento_servidor: Referencia, juntamente com o campo “equipamento_id”, o equipamento relativo a ordem de serviço.

10. equipamento_id: Referencia, juntamente com o campo “equipamento_servidor”, o

equipamento relativo a ordem de serviço.

11. marca: Armazena informações a respeito da marca do equipamento relativo a ordem de serviço.

12. modelo: Armazena informações a respeito do modelo do equipamento relativo a ordem de serviço.

13. numero_de_serie: Armazena informações a respeito do número de série do equipamento relativo a ordem de serviço.

14. dispositivos: Armazena informações a respeito dos dispositivos que possam estar acompanhando o equipamento relativo a ordem de serviço.

15. relato: Armazena dados informados pelo cliente a respeito do defeito apresentado pelo equipamento relativo a ordem de serviço.

16. estado_conservacao: Armazena informações referentes ao estado de conservação apresentado pelo equipamento ao ser entregue pelo cliente.

17. status: Armazena informações que permitem ao software identificar e definir o status da ordem de serviço.

18. senha: Armazena a senha gerada aleatoriamente que permite ao cliente acompanhar, on-line, o status da ordem de serviço por ele requisitada.

19. data_cadastro: Armazena a data, hora, minuto, segundo em que foi aberta a ordem de serviço.

A tabela “tecnicos_os” é responsável por armazenar os dados dos técnicos cadastrados no software. Ela contém apenas os campos que lhe permitem referenciar funcionários, pois um técnico é um funcionário, não sendo portanto, necessário que sejam aqui apresentados seus campos.

A tabela “atendentes_os” é responsável por armazenar os dados dos atendentes cadastrados no software, e assim como a tabela “tecnicos_os”,possui apenas os campos que lhe permitem referenciar os registros da tabela funcionários, não sendo portanto, necessário que sejam aqui apresentados seus campos.

A tabela “equipamentos_os” é responsável por armazenar os dados dos tipos de equipamentos para os quais são oferecidos serviços de manutenção, na empresa que utiliza o software. Eis seus campos:

1. servidor: Este campo referencia o servidor no qual o equipamento foi cadastrado.

2. id: É um código numérico que identifica o equipamento dentro do servidor no qual ele foi cadastrado.

3. nome: Texto que representa o nome do equipamento.

4. data_cadastro: Armazena a data, hora, minuto, segundo em que foi cadastrado o

equipamento.

A tabela “procedimentos_os” armazena os tipos e valores de procedimentos de manutenção que podem ser realizados pelos técnicos. Eis seus campos:

1. servidor: Este campo referencia o servidor no qual a ordem de serviço foi aberta.

2. id: É um código numérico que identifica a ordem de serviço dentro do servidor no qual ela foi aberta.

3. descricao: Texto que representa a descrição do procedimento e permita identificá-lo.

4. produto_servidor: Este campo, juntamente com o campo “produto_id”, permite identificar o produto que pode estar envolvido com este procedimento.

5. produto_id: Este campo, juntamente com o campo “produto_servidor”, permite identificar o produto que pode estar envolvido com o procedimento.

6. quantidade: Identifica o número de unidades do produto que pode estar relacionado com o procedimento.

7. taxa_de_servico: É um valor que pode ser cobrado pela mão de obra do técnico referente a este procedimento.

8. valor_final: É o valor total do procedimento de reparo que deve ser cobrado do cliente. Este valor total é calculado a partir do valor do produto que pode estar associado ao procedimento multiplicado pela quantidade deste produto

9. desconto: Este campo permite que se faça um desconto no valor total do procedimeto de reparo.

A tabela “procedimentos_os_executados” armazena os procedimentos executados em determinada ordem de serviço, e possui apenas os campos que permitem associar um procedimento executável a ordem de serviço correspondente. Deste modo, não é necessário que sejam aqui apresentados os campos desta tabela.

O arquivo “controlOS.php”, seguindo o padrão MVC, define as regras de negócio referentes a diversas operações, dentre as quais devem ser destacadas:

4. frmCadastraTecnicos: Esta operação é requisitada no formulário de cadastramento de técnicos.

5. frmCadastraAtendentes: Esta operação é requisitada no formulário de cadastramento de atendentes.

6. frmCadastraEquipamentos: Esta operação é requisitada no formulário de cadastramento de equipamentos.

7. frmCadastraProcedimentos: Esta operação é requisitada no formulário de cadastramento de procedimentos de reparos executáveis.

8. frmAberturaOrdemServico: Esta operação é requisitada na tela de abertura de ordens de serviço

9. alteraStatusOrdemServico: Esta operação é requisitada nas diversas situações onde é necessária a alteração do status da ordem de serviço.

O arquivo “viewOS.php” foi desenvolvido para representar a camada de apresentação, e tem por finalidade apresentar a tela de visualização e pesquisa de técnicos, o formulário de cadastramento de técnicos, a tela de visualização e pesquisa de atendentes, o formulário de cadastramento de atendentes, a tela de visualização e pesquisa de procedimentos de reparo executáveis, o formulário de cadastramento de procedimentos de reparo executáveis, a tela de visualização e pesquisa de equipamentos, o formulário de cadastramento de equipamentos, a tela de abertura de ordem de serviço, a tela de visualização de ordens de serviço da bancada (tela do técnico) e a tela de visualização de ordens de serviço dos atendentes.

O arquivo “relatorioAberturaOs.php” foi desenvolvido para que o software pudesse gerar o relatório de abertura de ordem de serviço.

O arquivo “relatorioFechamentoOs.php” foi desenvolvido para que o software pudesse gerar o relatório de fechamento de ordem de serviço.

O arquivo “acessocliente.php” foi desenvolvido para que o cliente pudesse acompanhar on-line a ordem de serviço por ele contratada.