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

FELIPE SABEDOT

SISTEMA PARA GERENCIAMENTO DE UMA METALÚRGICA DE PEQUENO PORTE

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO
2015

FELIPE SABEDOT

SISTEMA PARA GERENCIAMENTO DE UMA METALÚRGICA DE PEQUENO PORTE

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

Orientadora: Profa. Beatriz Terezinha Borsoi

PATO BRANCO
2015

ATA Nº: **263**

DEFESA PÚBLICA DO TRABALHO DE DIPLOMAÇÃO DO ALUNO **FELIPE SABEDOT**.

Às 19:40 hrs do dia 10 de junho de 2015, Bloco V da UTFPR, Câmpus Pato Branco, reuniu-se a banca avaliadora composta pelos professores Beatriz Terezinha Borsoi (Orientadora), Robison Cris Brito (Convidado) e Alessandro Graczyk Moraes (Convidado), para avaliar o Trabalho de Diplomação do aluno Felipe Sabedot, matrícula 1116681, sob o título **Sistema para gerenciamento de uma metalúrgica**; como requisito final para a conclusão da disciplina Trabalho de Diplomação do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, COADS. Após a apresentação o candidato foi entrevistado pela banca examinadora, e a palavra foi aberta ao público. Em seguida, a banca reuniu-se para deliberar considerando o trabalho **APROVADO**. Às 20:20 hrs foi encerrada a sessão.

Beatriz Borsoi

Profa. Beatriz Terezinha Borsoi, Dr.
Orientadora

Prof. Robison Cris Brito, M.Sc.
Convidado

Prof. Alessandro Graczyk Moraes,
Convidado

Coordenadora do Trabalho de Diplomação
Coordenador do Trabalho de Diplomação

Prof. Edilson Pontarolo, Dr.
Coordenador do Curso

RESUMO

SABEDOT, Felipe. Sistema para gerenciamento de uma metalúrgica de pequeno porte. 2015. 49 f. Monografia de Trabalho de Conclusão de Curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

O sistema implementado como resultado deste trabalho se destina a uma empresa específica que é uma metalúrgica de pequeno porte. Sistemas para esse tipo e porte de empresa são baseados em cadastros que permitem realizar o controle de entradas e saídas, o gerenciamento de contas a pagar e a receber, a realização de orçamentos e vendas e a geração de relatórios. Verificou-se que um sistema *desktop* poderia atender de maneira suficiente os requisitos de uma empresa com essas especificidades. Optou-se pela linguagem Java como tecnologia para a implementação do sistema pela possibilidade de uso de conceitos de orientação a objetos e pelo interesse no estudo da mesma. Como banco de dados foi utilizado o MySQL. A análise do sistema foi realizada utilizando a Linguagem de Modelagem Unificada, a UML.

Palavras-chave: Sistema para gerenciamento de metalúrgica. Tecnologia de Informação nas empresas. Linguagem Java.

ABSTRACT

SABEDOT, Felipe. Management system of a small metallurgical company. 2015. 49 f. Monografia de Trabalho de Conclusão de Curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

The system implemented as a result of this work is designed to a specific company, a small metallurgical company. A system for managing a small metallurgical company is based on records that allow you to perform the control of entries and exits, to manage accounts, to check outgoings and bills to receive, the realization of budgets and sales, and reporting. It was found that a desktop system could meet, in an adequate manner, the requirements of a company with these characteristics. It was decided to use the Java language by the possibility of using object-oriented concepts and the interest of studying it. The database used was MySQL and the system analysis was made using UML (Unified Modeling Language) concepts.

keywords: Metallurgical company management system. Information Tecnology in business. Java language.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

DAO	*Data Access Object*
GUI	*Grafical User Interface*
IDE	*Integrated Development Environment*
MVC	*Model-View-Controller*
TI	Tecnologia da Informação

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo são apresentadas as considerações iniciais que contêm o contexto no qual se insere a proposta do trabalho, que é um sistema para gerenciamento de uma indústria metalúrgica de pequeno porte. Também são apresentados os objetivos e a justificativa do trabalho. Por fim está a organização do texto por meio da apresentação dos seus capítulos subsequentes.

1.1 CONSIDERAÇÕES INICIAIS

A facilidade de acesso aos recursos computacionais, seja hardware ou software, e a necessidade de gerenciamento de informações que as empresas têm pelo volume de dados manipulados, independentemente do porte da empresa, tem feito com que o uso de sistemas computacionais passe a ser um item quase imprescindível na gestão dos negócios.

A facilidade de acesso é colocada pelo custo relativamente acessível de um computador. E é possível encontrar de forma gratuita sistemas operacionais e diversos aplicativos simples, mas funcionais, para uso na automação de escritório e mesmo para gestão de funcionalidades básicas de negócios de pequeno porte.

A quantidade de dados (agregados em informação quando fornecem suporte às atividades de negócio) manipulada pelas empresas é cada vez maior. É preciso controle adequado de contas, que estão vinculadas às receitas e às despesas, para que lucros possam ser maximizados e custos reduzidos. O gerenciamento da empresa baseado em dados reais do próprio negócio e do mercado é relevante para conquistar novos clientes e para manter os existentes. Um sistema computacional pode auxiliar na definição de margens adequadas de lucro, quantidades em estoque, realização de promoções e descontos, entre outros.

Apesar de existirem diversos sistemas para gerenciamento de negócios, verificou-se a possibilidade de desenvolver um sistema para uma indústria metalúrgica de pequeno porte. O desenvolvimento desse sistema é uma oportunidade de estudo para o autor deste trabalho (pela realização de atividades de todo o ciclo de vida de desenvolvimento de software) e de atender as necessidades específicas da empresa para a qual o sistema foi desenvolvido.

A linguagem Java foi escolhida para implementação pela possibilidade de emprego de conceitos de orientação a objetos e pelos recursos que a mesma oferece. O sistema desenvolvido é para ambiente *desktop*. Ressalta-se que diversas linguagens permitem o

desenvolvimento de sistemas utilizando o paradigma de orientação a objetos. Java foi escolhida pelo interesse do autor deste trabalho no aprendizado da mesma. A modelagem do sistema foi realizada como estágio curricular pelo autor deste trabalho. A documentação da modelagem consta neste texto para facilitar o entendimento do sistema desenvolvido.

1.2 OBJETIVOS

O objetivo geral apresenta o resultado principal obtido com o desenvolvimento deste trabalho e os objetivos específicos o complementam.

1.2.1 Objetivo Geral

Implementar um sistema para gerenciamento de uma metalúrgica de pequeno porte.

1.2.2 Objetivos Específicos

Facilitar a geração de pedidos e orçamentos de produtos vendidos e serviços realizados em uma metalúrgica de pequeno porte.

Fornecer uma ferramenta que facilite o controle de entradas e saídas da empresa e o gerenciamento de contas a pagar e a receber.

Disponibilizar relatórios que auxiliem no gerenciamento do negócio.

1.3 JUSTIFICATIVA

Um sistema para gerenciamento de uma metalúrgica é útil e mesmo necessário para auxiliar no controle de matéria-prima e na venda dos produtos manufaturados. O controle de contas e clientes e a possibilidade de realizar orçamentos também são requisitos importantes e que foram implementados no sistema desenvolvido.

A empresa para a qual o sistema foi desenvolvido já possui um sistema informatizado. A troca foi motivada pela sistema em uso ser um aplicativo de uso geral, ou seja, para funcionalidades básicas e genéricas de contas, estoque e cadastros, e que não atende

todas as necessidades específicas do negócio da empresa. Um software específico para metalúrgica pode reduzir custos, por trabalhar especificamente com as regras de negócio, interesses e necessidades da empresa. Além disso, o usuário principal do sistema é sócio da empresa e pela experiência e contato direto, pode determinar especificamente os requisitos e necessidades dos usuários. O entendimento das funcionalidades pretendidas para o sistema foi facilitado pelo fato de a empresa ser da família do autor deste trabalho.

A justificativa da escolha da linguagem Java e da implementação ser realizada para ambiente *desktop* decorre de a indústria que utilizará o sistema ser de pequeno porte. A linguagem Java possibilita o uso de conceitos de orientação a objetos. Orientação a objetos é um paradigma de programação que facilita o reuso e a organização das entidades que compõem o sistema.

1.4 ESTRUTURA DO TRABALHO

Este texto está organizado em capítulos. Neste Capítulo está a introdução que é composta pelas considerações iniciais e a apresentação dos objetivos e da justificativa do trabalho.

No Capítulo 2 está o referencial teórico que se refere ao uso de tecnologias de informação, com foco em sistemas, nas empresas.

O Capítulo 3 apresenta os materiais e o método utilizados na modelagem do sistema e na implementação do sistema.

No Capítulo 4 é apresentado o resultado obtido com a realização deste trabalho que é implementação de um sistema para gerenciamento de uma indústria metalúrgica de pequeno porte.

As considerações finais estão no Capítulo 5.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta o referencial teórico que fundamenta conceitualmente a proposta deste trabalho que se refere a um sistema para gestão de uma empresa de pequeno porte que atua no setor de metalurgia.

2.1 TECNOLOGIA DA INFORMAÇÃO NAS EMPRESAS

O estudo da influência do uso da Tecnologia da Informação (TI) na eficiência das empresas é justificado por várias razões (SOUZA: ARPINO, 2011), dentre as quais estão: a redução de custos, a melhoria no controle dos processos e a busca pela melhoria da competitividade da empresa (SOUZA; SZAFIR-GOLDSTEIN, 2005). Maçada (2000) reforça que várias pesquisas atestam que a literatura falha ao não mostrar de maneira conclusiva o impacto estratégico e econômico que os investimentos em TI têm sobre a produtividade organizacional. Sacilotti (2011), no entanto, ressalta que o sucesso empresarial está fortemente amparado na capacidade de perceber, organizar e administrar as informações, tirando proveito das ferramentas e dos recursos oferecidos pelas tecnologias de informação e comunicação.

Avaliar de forma adequada a aplicação dos recursos de TI tem sido uma preocupação constante para as empresas dos diversos setores e portes, com destaque para as micro, pequenas e médias (SOUZA *et al.*, 2005). Ressaltam-se essas categorias de empresas por diversas razões, incluindo o fato de elas, geralmente, possuírem menos recursos para investir e equipes menores e serem mais vulneráveis às variações da economia e do mercado.

Como base em um estudo realizado na Itália, Becchetti, Paganetto e Bedoya (2003) mostraram que o investimento em TI em pequenas e médias empresas influenciou positivamente na elaboração de novos produtos e processos, além de aumentar a capacidade produtiva e a produtividade. Essa conclusão é amparada por Davis que ainda em 1989 faz referência à TI como fornecedora do potencial para um aumento substancial nos ganhos com o desempenho nas empresas (DAVIS, 1989). Moraes, Bobsin e Lana (2006, p. 14) indicam que existe um efeito positivo que é significativo em termos de desempenho das organizações em relação aos investimentos realizados em TI.

Na perspectiva de processos, “os investimentos em TI criam vantagens competitivas no melhoramento da eficiência operacional de processos intermediários da organização, conduzindo a um melhor desempenho organizacional” (CANUTO; CHEROBIM, 2009, p. 3).

Albertin e Albertin (2007, p. 1) destacam que a TI tem sido considerada como um dos componentes mais importantes do ambiente empresarial e as organizações brasileiras têm utilizado a tecnologia em nível estratégico ou operacional. A significativa contribuição da TI com a estratégia da empresa está, também, estar relacionada ao grande volume de informação disponível e que as empresas manipulam e que os gestores necessitam para tomar decisões mais efetivas.

Investimentos em TI podem gerar vantagens competitivas, bem como a necessidade de outros investimentos para manter e/ou ampliar essas vantagens. Para isso é importante direcionar os investimentos de forma eficiente, controlada e alinhada aos objetivos de negócios da organização (OLIVEIRA *et al.,* 2005). Porém, Kearns e Sabherwal (2006) ressaltam que parte significativa dos recursos em investimentos de TI são destinados a aspectos operacionais e não estratégicos e este deveria ser um dos componentes fundamentais das organizações (NEVO; WADE, 2010; RUGGIERO; GODOY, 2005).

Apesar da importância atribuída a TI nos negócios, ocorre a falta de investimento em aspectos estratégicos. Além disso, é observada certa relutância de funcionários operacionais e de gestores na utilização de recursos de TI. Essa resistência pode ocorrer por diversos motivos como a falta de conhecimento dos benefícios potenciais da tecnologia, a expectativa inicial de dificuldade de uso, aspectos cognitivos individuais e falta de treinamento, entre outros (MALAQUIAS; ALBERTIN, 2011; VENKATESH; BALA, 2008). A associação que frequentemente é realizada entre TI e necessidade de mudança no cenário organizacional pode ocasionar a resistência ao uso dos recursos da tecnologia (FETZNER; FREITAS, 2012).

Albertin (2001) destaca três categorias de variáveis que devem ser estudadas e tratadas para a criação de um ambiente adequado para que o uso de TI tenha sucesso nas organizações. Essas categorias são (ALBERTIN, 2001):

1) Cenário - relacionado à propensão ao uso de TI construída ao longo da história da organização, tendo de forma clara a estratégia de negócio da organização nos níveis estratégico, tático e operacional e como a TI contribuirá para a concretização dessa estratégia.

2) Atores - abrange o apoio da alta gerência ao projeto de TI e envolve um patrocinador do projeto, que acredite e argumente a importância do projeto e da própria TI; e a capacitação da equipe para o uso dos recursos.

3) Planejamento da intervenção - a organização deve estar ciente da relevância do projeto e participativa, por meio de um patrocinador, por exemplo, no desenvolvimento do projeto.

Para minimizar possíveis impactos negativos nas empresas quando da adoção de recursos de TI é necessário analisar os impactos decorrentes da implementação, eliminando dúvidas e receios, buscando facilitar o processo de mudança que, se não for bem compreendido e aceito pelos gestores, pode implicar a postergação de investimento em TI, mesmo que sua importância seja, no momento, evidente (SOUZA, ARPINO, 2011).

Albertin e Albertin (2009, p. 7) ressaltam que "o uso da TI por si só não determina o sucesso e o bom desempenho de uma organização". Canuto e Cherubim (2009) destacam que a relação entre investimento em TI e desempenho esperado do uso desses recursos deve considerar o uso que é (ou será) feito da TI como uma variável importante.

Lunardi, Dolci e Maçada (2009) também apontam que uma estrutura organizacional inapropriada para a implementação de TI pode contribuir para a baixa utilidade percebida (DAVIS, 1989) para o uso da TI. Essa baixa utilidade percebida pode fazer com que a decisão de investimento seja adiada (MALAQUIAS; ALBERTIN, 2011). O treinamento apresenta impacto significativo na utilidade percebida de sistemas de informação, sendo que tal impacto é afetado pela facilidade de uso percebida. Treinamentos podem ser utilizados para contornar o efeito negativo sobre os investimentos em TI em relação à baixa expectativa de seus potenciais benefícios (SILVA; DIAS, 2006).

3 MATERIAIS E MÉTODO

Este capítulo apresenta os materiais e o método utilizados. Os materiais se referem às tecnologias e ferramentas para modelar e implementar o sistema. O método reporta a sequência das principais atividades realizadas para desenvolver este trabalho.

3.1 MATERIAIS

No Quadro 1 estão as tecnologias e as ferramentas utilizadas para a modelagem e a implementação do sistema.

Ferramenta / Tecnologia	**Versão**	**Finalidade**	**Referência**
Cacoo	1.69 2015	Documentação da modelagem baseada na UML e ER do banco de dados	http://www.cacoo.com
Linguagem Java	JDK 8	Linguagem de programação	http://www.oracle.com
NetBeans	8.0	IDE (*Integrated Development Environment*) de desenvolvimento	https://netbeans.org
MySQL	5.7	Banco de dados	http://www.mysql.com
Sequel Pro	1.0.2	Administrador do banco de dados	http://www.sequelpro.com
iReport	5.1.0	Geração dos relatórios	http://sourceforge.net/projects/ireport/files/iReport/iReport-5.1.0/

Quadro 1 – Tecnologias e ferramentas

3.2 MÉTODO

As etapas para a modelagem e a implementação de funcionalidades de sistema seguiram o modelo sequencial linear proposto por Pressman (2005). O uso desse modelo é justificado porque o sistema é simples e a empresa é da família do autor deste trabalho, assim, foi possível obter uma visão completa e detalhada do sistema, antes mesmo da sua modelagem.

As etapas definidas foram:

a) **Requisitos**

O levantamento e a modelagem dos requisitos foram realizados como estágio curricular pelo autor deste trabalho. A modelagem realizada é apresentada neste texto para o entendimento dos requisitos do sistema.

b) Análise e projeto

Considerando que a modelagem do sistema e a elaboração do referencial teórico foram realizadas previamente (como estágio), na realização deste trabalho os requisitos foram revistos. O diagrama de classes foi redefinido e o diagrama de entidades e relacionamento do banco de dados foi revisto.

c) Implementação e testes

A implementação foi realizada utilizando a linguagem Java com o NetBeans como ambiente de desenvolvimento. Os testes de código foram realizados à medida que o aplicativo estava sendo implementado. Os testes foram realizados pelo autor deste trabalho e consistiram na verificação do código e no atendimento às regras de negócio e requisitos do sistema. Os testes de usuários foram realizados por pessoas da empresa, futuros usuários do sistema.

4 PROJETO DO SISTEMA

Este capítulo apresenta a modelagem de um sistema para gerenciamento das atividades de uma metalurgia e a implementação de funcionalidades básicas de um cadastro.

4.1 APRESENTAÇÃO DO SISTEMA

O sistema gerenciará as atividades de uma metalúrgica. O acesso dos vendedores, a princípio, será por um sistema *desktop*, implementado na linguagem Java. Em uma versão futura, o sistema poderá oferecer novos serviços, como nota fiscal eletrônica e controle de estoque. Em adição, o sistema pode ser expandido para serviços web e móvel.

O software tem como requisito principal o cadastro de orçamentos e vendas. Nesse cadastro o vendedor acrescentará uma série de informações que geram três relatórios:

a) Para o cliente – com a quantidade de aberturas e suas respectivas medidas e custos, seguido pelo valor total do orçamento;
b) Nota Promissória – com as informações da venda;
c) Para o estoque – com os materiais em unidades (podem ser separados por abertura).

Este relatório será utilizado para organizar a entrega e a instalação dos materiais.

A tela de cadastro de clientes mostra a lista com todos os clientes e botões para editar, cadastrar um novo cliente e excluir clientes cadastrados. Ao clicar nos botões “novo” ou “editar” uma nova janela é apresentada.

Nessas janelas serão informados/editados os dados do cliente. As telas de cadastro de fornecedores e funcionários seguem o mesmo padrão da tela de cadastro de clientes. Na tela de cadastro de produtos são listados os produtos com suas respectivas quantidades. Quando a venda é realizada, automaticamente é baixado do estoque, porém há a opção de realizar balanço manual de estoque. As demais telas de cadastro, como, por exemplo, a de matéria-prima, seguem o mesmo padrão da tela de cadastro de clientes.

4.2 MODELAGEM DO SISTEMA

Os Quadros a seguir apresentam os requisitos funcionais e os requisitos não funcionais relacionados. O Quadro 1 apresenta a descrição no requisito funcional "Cadastrar funcionários".

1 Requisito funcional: Cadastrar funcionários	
Descrição: O administrador do sistema deve cadastrar os funcionários. Funcionários são as pessoas que usarão o sistema (com limitações). O funcionário deve fornecer os dados: Nome, Setor, Senha, RG, CPF, Telefone, Endereço, Bairro, CEP, Cidade, Estado e País.	
Requisitos não funcionais:	
Nome	**Descrição**
NF 1.1 Acesso ao sistema	Cada funcionário deverá usar obrigatoriamente a sua própria "conta" com a respectiva senha, para o administrador ter um controle maior das vendas.
NF 1.2 Campos obrigatórios	Os campos Nome, Setor, Senha, RG, Telefone e Endereço, são obrigatórios para salvar o cadastro. Os outros dados podem ser complementados em outro momento, porém, são igualmente necessários.
NF 1.3 Máscara em campos	Os campos RG, CPF, CEP e telefone devem apresentar máscaras para entrada de dados.
NF 1.4 Cadastro	Nos campos cidade, estado e país, o sistema apresentará uma caixa de combinação com as cidades, estados e países cadastrados, respectivamente.
NF 1.5 Código funcionário	O código do funcionário deve ser gerado automaticamente.

Quadro 1 – Requisito cadastrar funcionários

O Quadro 2 apresenta a descrição no requisito funcional "Cadastrar clientes".

2 Requisito funcional: Cadastrar clientes	
Descrição: O funcionário do setor de vendas deve cadastrar o cliente antes de fazer o orçamento. O cliente informa o seu Nome, RG, CPF, Telefone, Endereço, Bairro, CEP, Cidade, Estado e País.	
Requisitos não funcionais:	
Nome	**Descrição**
NF 2.1 Pessoa física ou jurídica	O funcionário deve escolher se o cliente é pessoa física ou jurídica.
NF 2.2 Campos obrigatórios	Os campos Nome, RG/IE, CPF/CNPJ Telefone e Endereço, são obrigatórios para salvar o cadastro. Os outros dados podem ser complementados em outro momento, porém, são igualmente necessários. Se for cliente pessoa jurídica o campo obrigatório não será RG e sim Inscrição Estadual.
NF 2.3 Código	O código do cliente deve ser gerado automaticamente.

cliente	
NF 2.4 Nome cliente	O sistema deverá avisar se já existe um cadastro com o mesmo nome. Se o administrador permitir, o sistema deve liberar o cadastro mesmo assim.
NF 2.5 Máscaras de campos	Nos campos RG/IE, CPF/CNPJ, CEP e telefone, o sistema apresentará as máscaras para cada campo.
NF 2.6 Campos	Nos campos cidade, estado e país, o sistema apresentará uma caixa de combinação com todas as cidades, estados e países.

Quadro 2 – Requisito cadastrar clientes

No Quadro 3 está a descrição do requisito funcional "Cadastrar produtos" e dos requisitos não funcionais associados.

3 Requisito funcional: Cadastrar produtos	
Descrição: Os funcionários do setor de vendas devem cadastrar os produtos que serão vendidos pela empresa. Os dados solicitados serão: Nome, Quantidade Inicial, Altura, Comprimento, Espessura e Preço.	
Requisitos não funcionais:	
Nome	**Descrição**
NF 3.1 Código produto	O código do produto deve ser gerado automaticamente.
NF 3.2 Campos obrigatórios	Todos os campos são obrigatórios para salvar o cadastro.

Quadro 3 – Requisito cadastrar produtos

No Quadro 4 está a descrição do requisito funcional "Cadastrar fornecedores" e dos requisitos não funcionais associados.

4 Requisito funcional: Cadastrar fornecedores	
Descrição: Antes de cadastrar a Matéria-Prima, os funcionários do setor de compras devem cadastrar o fornecedor, inserindo os dados da empresa: Razão Social, Inscrição Estadual, CNPJ, Telefone, Endereço, Bairro, CEP, Cidade, Estado e País.	
Requisitos não funcionais:	
Nome	**Descrição**
NF 4.1 Código fornecedor	O código do fornecedor deve ser gerado automaticamente.
NF 4.2 Campos obrigatórios	Os campos Razão Social, Telefone e Endereço são obrigatórios para salvar o cadastro. Os outros dados podem ser complementados em outro momento, porém, são igualmente necessários.
NF 4.3 Máscara de campo	Nos campos Razão Social, CNPJ, CEP e telefone, o sistema apresentará as mascaras para cada campo.
NF 4.4 Campos	Nos campos cidade, estado e país, o sistema apresentará uma caixa de combinação com todas as cidades, estados e países.

Quadro 4 – Requisito cadastrar fornecedores

No Quadro 5 está a descrição do requisito funcional “Cadastrar matéria-prima” e dos requisitos não funcionais associados.

5 Requisito funcional: Cadastrar matéria-prima	
Descrição: Os funcionários do setor de compras devem cadastrar a matéria-prima utilizada pela empresa para instalação das aberturas. Os dados solicitados serão: Nome, Quantidade Inicial, Comprimento, Largura e Espessura.	
Requisitos não funcionais:	
Nome	**Descrição**
NF 5.1 Código fornecedor	O código do produto deve ser gerado automaticamente.
NF 5.2 Campos obrigatórios	Os campos Nome, Quantidade Inicial (esta, mesmo se for zero), Comprimento, Espessura e Preço, são obrigatórios para salvar o cadastro.

Quadro 5 – Requisito cadastrar matéria-prima

O Quadro 6 apresenta a descrição do requisito funcional “Cadastrar orçamentos” e dos requisitos não funcionais associados.

6 Requisito funcional: Cadastrar orçamento	
Descrição: Os funcionários do setor de vendas devem cadastrar um novo orçamento ao negociar com o cliente. A tela de orçamento será da seguinte forma: na parte superior haverá o campo para informar o cliente; abaixo haverá um campo para selecionar os produtos que serão orçados; juntamente com os produtos, será mostrado o preço e solicitado para informar a quantidade de cada produto que será orçado.	
Requisitos não funcionais:	
Nome	**Descrição**
NF 6.1 Código orçamento	O código do orçamento deve ser gerado automaticamente.
NF 6.2 Apresentação preço	Ao selecionar o produto, o preço será mostrado automaticamente ao lado. Será possível alterar manualmente o preço, se necessário.
NF 6.3 Busca em cadastro	O funcionário poderá buscar o cadastro do cliente pelo nome, mas mesmo assim terá a opção de colocar diretamente o código do cliente que retornará automaticamente o seu nome ao lado. O mesmo acontece na busca dos produtos.

Quadro 6 – Requisito cadastrar orçamentos

O Quadro 7 apresenta a descrição do requisito funcional “Cadastrar orçamentos” e dos requisitos não funcionais associados.

7 Requisito funcional: Cadastrar conta	
Descrição: Os funcionários do setor financeiro devem cadastrar uma nova conta receber a conta da mercadoria comprada. A tela de contas será da seguinte forma: na parte superior haverá os filtros para busca de contas; abaixo haverá uma tabela onde os registros serão mostrados. Na parte inferior da tela serão mostradas as informações sobre as contas.	
Requisitos não funcionais:	
Nome	**Descrição**
NF 7.1 Código conta	O código da conta deve ser gerado automaticamente.
NF 7.2 Apresentação conta	Quando o usuário mudar o filtro, a tabela se organizara automaticamente, de acordo com o filtro selecionado.
NF 7.3 Edição e pagamento	O usuário pode pagar ou alterar uma conta.

No Quadro 8 está a relação dos atores administrador, vendedor e financeiro e dos requisitos associados aos mesmos.

Ação	Descrição	Requisitos relacionados
Gerenciar funcionários	Permitir que atores adicionem, consultem, alterem ou removam funcionários.	Cadastrar funcionários Alterar funcionários Consultar funcionários Excluir funcionários
Gerenciar clientes	Permitir que atores adicionem, consultem, alterem ou removam clientes.	Cadastrar clientes Alterar clientes Consultar clientes Excluir clientes
Gerenciar produtos	Permitir que atores adicionem, consultem, alterem ou removam produtos.	Cadastrar produtos Alterar produtos Consultar produtos Excluir produtos
Gerenciar fornecedores	Permitir que atores adicionem, consultem, alterem ou removam fornecedores.	Cadastrar fornecedores Alterar fornecedores Consultar fornecedores Excluir fornecedores
Gerenciar matéria-prima	Permitir que atores adicionem, consultem, alterem ou removam matéria-prima.	Cadastrar matéria-prima Alterar matéria-prima Consultar matéria-prima Excluir matéria-prima
Gerenciar orçamento	Permitir que atores adicionem, consultem, alterem ou removam orçamentos.	Cadastrar orçamento Alterar orçamento Consultar orçamento Excluir orçamento

Gerenciar contas	Permitir que atores adicionem, consultem, alterem ou removam contas.	Cadastrar conta Alterar conta Consultar conta Excluir conta

Quadro 8 – Atores e requisitos

A Figura 1 apresenta o diagrama de casos de uso definidos para o sistema.

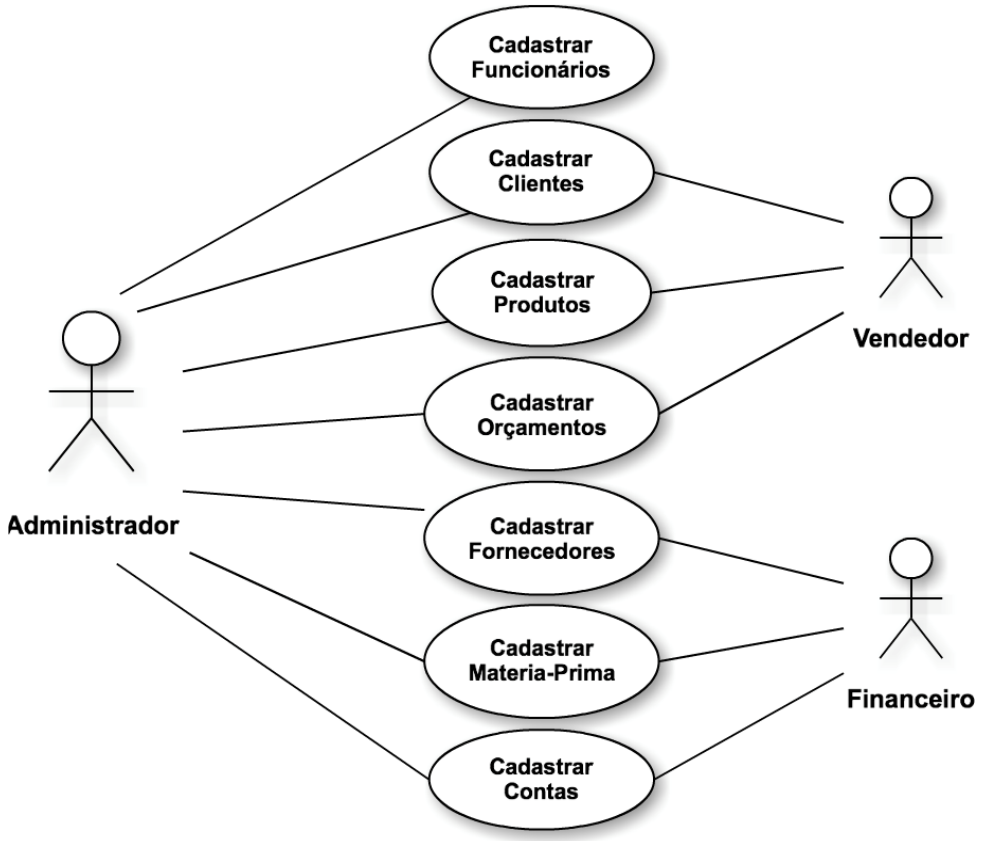


Figura 1 – Diagrama de casos de uso

Nos Quadros a seguir estão descritas as operações realizadas com os casos de uso apresentados no Diagrama da Figura 6. O Quadro 8 apresenta a operação cadastrar de todos os casos de uso.

Caso de uso: Cadastrar. **Descrição:** Este caso de uso permite cadastrar. **Evento Iniciador:** Tela de cadastros disponíveis. **Atores:** Administrador, vendedor, financeiro (cada um com suas limitações, exceto o administrador) **Pré-condição:** Os campos obrigatórios devem estar preenchidos. **Seqüência de Eventos:** 1 – [IN] Usuário informa dados. 2 – [OUT] Sistema valida esses dados e inclui no banco de dados. 3 – [OUT] Sistema retorna mensagem de cadastro efetuado com sucesso.

Pós-Condições: O cadastro é salvo no banco de dados.
Requisitos correlacionados: F01, F02, F03, F04, F05 e F06.
Tratamento de exceções:
1 – Os campos obrigatórios não foram preenchidos
1.1 [IN] O usuário preenche os campos obrigatórios
1.2 Retorna ao passo dois.

Quadro 8 – Operação cadastrar

No Quadro 9 está a descrição da operação alterar realizada pelos casos de uso definidos para o sistema.

Caso de uso: Alterar.
Descrição: Este caso de uso permite alterar dados.
Evento Iniciador: Tela de alterações disponíveis.
Atores: Administrador, vendedor, financeiro (cada um com suas limitações, exceto o administrador)
Pré-condição: Não há.
Seqüência de Eventos:
1- Usuário analisa os dados que devem ser alterados.
2- [IN] Usuário altera os dados do cadastro.
3- [OUT] Sistema valida se é possível a alteração.
4- [OUT] Sistema retorna mensagem de cadastro efetuado com sucesso.
Pós-Condições: As alterações no cadastro são salvas no banco de dados.
Requisitos correlacionados: F01, F02, F03, F04, F05 e F06.
Tratamento de exceções:
1 – Faltaram dados.
1.1 – [IN] O usuário preenche os campos que restaram.
2.1 Retorna ao passo três.

Quadro 9 – Operação alterar

No Quadro 10 está a descrição da operação consultar realizada pelos casos de uso definidos para o sistema.

Caso de uso: Consultar.
Descrição: Este caso de uso permite consultar dados.
Evento Iniciador: Tela de consultas disponíveis.
Atores: Administrador, vendedor, financeiro (cada um com suas limitações, exceto o administrador)
Pré-condição: Não há.
Seqüência de Eventos:
1- [IN] Usuário busca o cadastro para fazer a consulta.
2- [OUT] Sistema valida se esses dados estão disponíveis para consulta.
3- [OUT] Sistema mostra os dados para consulta ou retorna mensagem de que os dados não podem ser consultados.
Pós-Condições: A consulta é salva no banco de dados.
Requisitos correlacionados: F01, F02, F03, F04, F05, F06, F07.
Tratamento de exceções:

Quadro 10 – Operação consultar

No Quadro 11 está a descrição da operação excluir realizada pelos casos de uso definidos para o sistema.

Caso de uso: Excluir. **Descrição:** Este caso de uso permite excluir dados. **Evento Iniciador:** Tela de exclusões disponíveis. **Atores:** Administrador, vendedor, financeiro (cada um com suas limitações, exceto o administrador) **Pré-condição:** Não há. **Seqüência de Eventos:** 1- [IN] Usuário busca o cadastro para fazer a exclusão. 2- [OUT] Sistema valida se o cadastro pode ser excluído. 3- [OUT] Sistema retorna mensagem de exclusão não disponível ou exclusão efetuada com sucesso. **Pós-Condições:** Os cadastros excluídos são eliminados do banco de dados. **Requisitos correlacionados:** F01, F02, F03, F04, F05, F06, F07. **Tratamento de exceções:** 2 – O cadastro não pode ser excluído por algum motivo. 2.1 Retorna ao passo 1.

Quadro 11 – Operação excluir

A Figura 2 apresenta o diagrama de classes definido para o sistema.

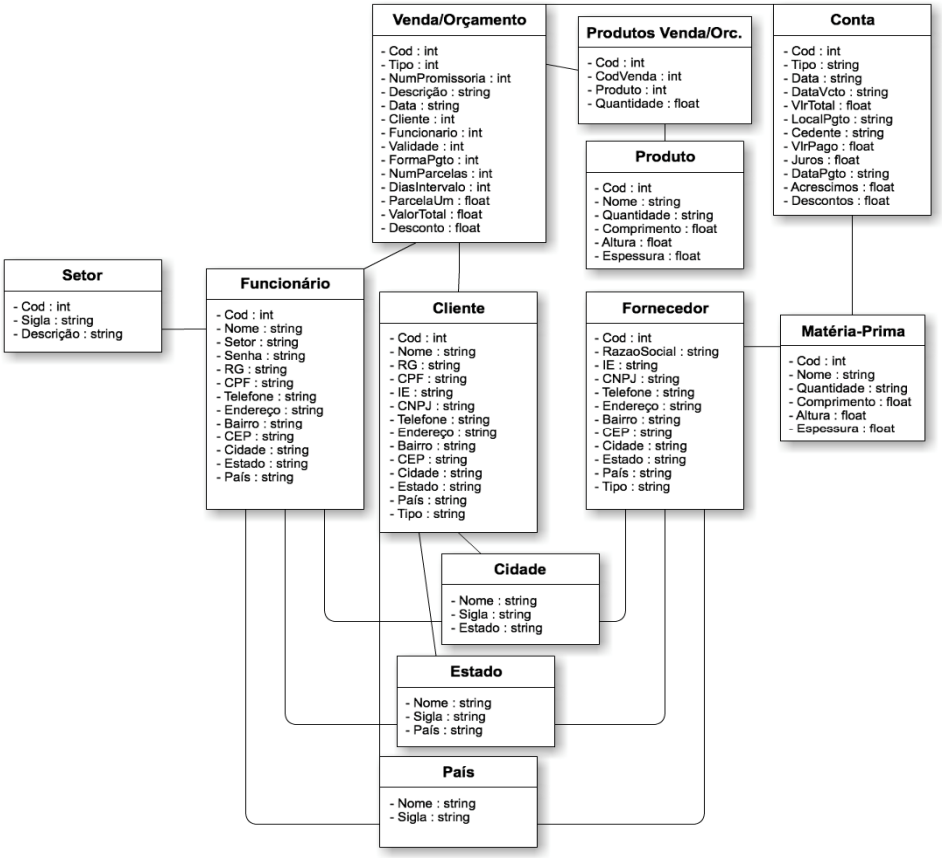


Figura 2 – Diagrama de classes

Nos Quadros 12 a 17 está descrição de cada uma das classes do diagrama de classes apresentado na Figura 2.

Identificação: Funcionário
Descrição: Nesta classe os dados do funcionário e o setor de trabalho serão requisitados, para ser gerada uma senha do sistema e então o cadastro é efetuado.
Requisitos: F01
Atributos:
NomeFunc (String): Nome do funcionário;
FuncaoFunc(String}): Função do funcionário;
SenhaFunc(String): Senha do funcionário;
RGFunc (String): RG do funcionário;
CPFFunc (String): CPF do funcionário;
TelefoneFunc (String): Telefone do funcionário;
EndereçoFunc (String): Endereço do funcionário (Rua e número);
BairroFunc (String): Bairro do funcionário;
CEPFunc (String): CEP do funcionário;
CidadeFunc (String): Cidade do funcionário;
EstadoFunc (String): Estado do funcionário;
PaisFunc (String): País do funcionário;

Quadro 12 – Descrição da classe funcionário

No Quadro 13 é apresentada a descrição da classe cliente.

Identificação: Cliente
Descrição: Nesta classe os dados do cliente serão requisitados e então o cadastro é efetuado.
Requisitos: F02
Atributos:
NomeCliente(String): Nome do cliente;
RGCliente (String): RG do cliente;
CPFCliente (String): CPF do cliente;
TelefoneCliente (String): Telefone do cliente;
EndereçoCliente (String): Endereço do cliente (Rua e número);
BairroCliente (String): Bairro do cliente;
CEPCliente (String): CEP do cliente;
CidadeCliente (String): Cidade do cliente;
EstadoCliente (String): Estado do cliente;
PaisCliente (String): País do cliente.

Quadro 13 – Classe cliente

O Quadro 14 apresenta a descrição da classe produto.

Identificação: Produto
Descrição: Nesta classe os dados do produto serão requisitados e então o cadastro é efetuado.
Requisitos: F03
Atributos:
NomeProduto (String): Nome do produto;
QuantInicial (Integer): Quantidade inicial do produto;
Comprimento (Float): Comprimento do produto;

Altura (Float): Altura do produto; Espessura (Float): Espessura do produto; Preco (Float): Preço do produto;

Quadro 14 – Classe produto

A descrição da classe fornecedor é apresentada no Quadro 15.

Identificação: Fornecedor **Descrição:** Nesta classe os dados do fornecedor serão requisitados e então o cadastro é efetuado. **Requisitos:** F04 **Atributos:** NomeFornecedor (String): nome do fornecedor; RGFornecedor (String): RG do fornecedor; CPFFornecedor (String): CPF do fornecedor; TelefoneFornecedor (String): telefone do fornecedor; EndereçoFornecedor (String): endereço do fornecedor (Rua e número); BairroFornecedor (String): Bairro do fornecedor; CEPFornecedor (String): CEP do fornecedor; CidadeFornecedor (String): Cidade do fornecedor; EstadoFornecedor (String): Estado do fornecedor; PaisFornecedor (String): País do fornecedor.

Quadro 15 – Classe fornecedor

O Quadro 16 apresenta a descrição da classe matéria-prima.

Identificação: Matéria-Prima **Descrição:** Nesta classe os dados da matéria-prima que será usada para as aberturas de alumínio serão requisitados, e então o cadastro é efetuado. **Requisitos:** F05 **Atributos:** NomeMateriaPrima (String): Nome da matéria-prima; QuantInicial (Integer): Quantidade inicial da matéria-prima; Comprimento (Float): Comprimento da matéria-prima; Altura (Float): Altura da matéria-prima; Espessura (Float): Espessura da matéria-prima; Preco (Float): Preço da matéria-prima;

Quadro 16 – Classe matéria-prima

No Quadro 17 está a descrição da classe orçamento.

Identificação: Orçamento **Descrição:** Nesta classe os dados do orçamento serão requisitados, e então o cadastro é efetuado. **Requisitos:** F06 **Atributos:** Cliente (String): Nome do cliente que será feito o orçamento; Produto (Float): Produtos que serão orçados; Quantidade (Integer): Quantidade de cada produto que será orçado;

Preco (Float): Preço de cada produto que será orçado; ValorTotal (Float): Valor Total do orçamento;

Quadro 17 – Classe orçamento

O diagrama de entidades e relacionamentos do banco de dados do sistema é apresentado na Figura 3.

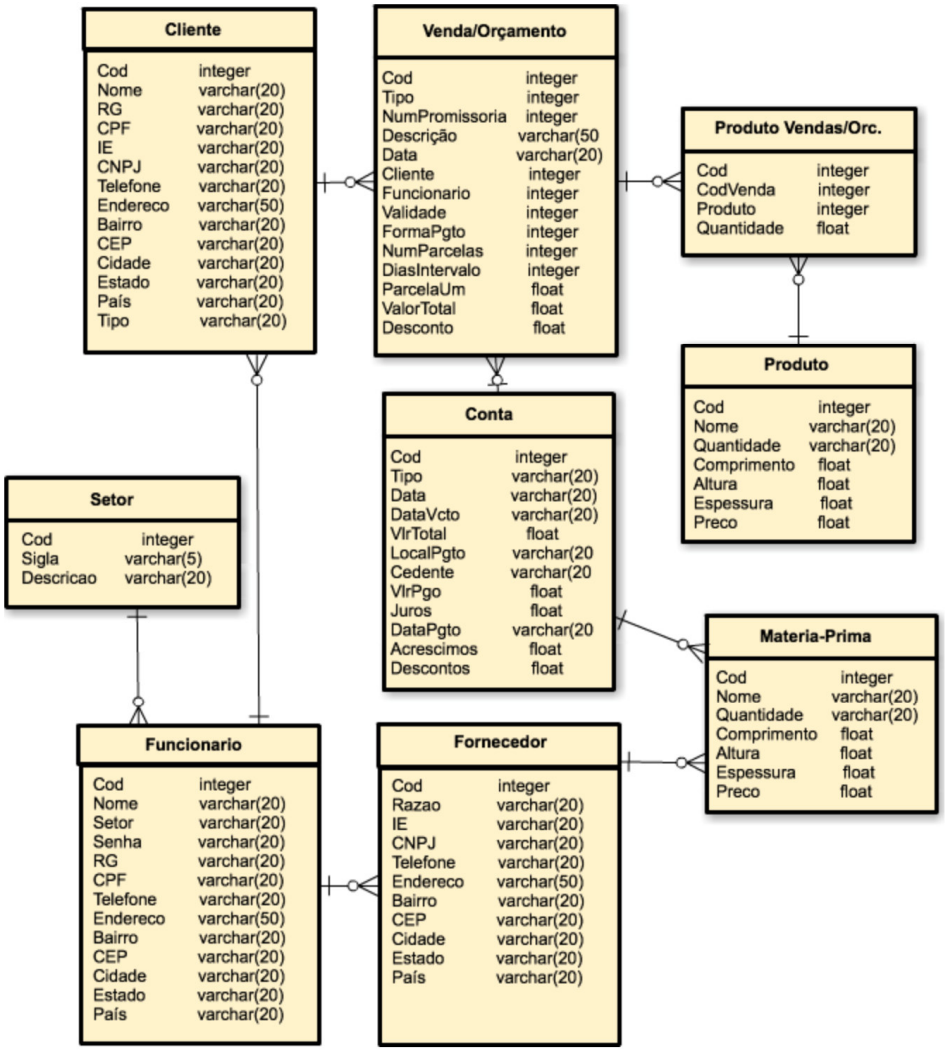


Figura 3 – Diagrama de entidades e relacionamentos

Os Quadros 18 a 26 apresentam a descrição das tabelas do diagrama de entidade e relacionamento constante na Figura 3.

No Quadro 18 está a descrição da tabela para o cadastro de funcionários.

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
FuncID	Integer	Não	Sim	Não	Auto-Denominado
NomeFunc	String	Não	Não	Não	
Setor	String	Sim	Não	Não	
Senha	String	Sim	Não	Não	
RGFunc	String	Não	Não	Não	
CPFFunc	String	Sim	Não	Não	
TelFunc	String	Não	Não	Não	
EndFunc	String	Não	Não	Não	
BairroFunc	String	Sim	Não	Não	
CEPFunc	String	Sim	Não	Não	
CidadeFunc	String	Sim	Não	Não	
EstadoFunc	String	Sim	Não	Não	
PaisFunc	String	Sim	Não	Não	

Quadro 18 – Tabela funcionários

A descrição da tabela setor é apresentada no Quadro 19.

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
SetorID	Integer	Não	Sim	Não	Auto-Denominado
Sigla	String	Não	Não	Não	
Descricao	String	Não	Não	Não	

Quadro 19 – Tabela setores

No Quadro 20 está a descrição da tabela para o cadastro de clientes.

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
ClienteID	Integer	Não	Sim	Não	Auto-Denominado
NomeCliente	String	Sim	Não	Não	
RazaoCliente	String	Não	Não	Não	
RGCliente	String	Sim	Não	Não	
InscEstCliente	String	Sim	Não	Não	
CPFCliente	String	Não	Não	Não	
CNPJCliente	String	Não	Não	Não	
TelCliente	String	Sim	Não	Não	
EndCliente	String	Sim	Não	Não	
BairroCliente	String	Não	Não	Não	
CEPCliente	String	Não	Não	Não	
CidadeCliente	String	Não	Não	Não	
EstadoCliente	String	Não	Não	Não	
PaisCliente	String	Não	Não	Não	
Tipo	String	Não	Não	Não	

Quadro 20 – Tabela clientes

A descrição dos campos da tabela para o cadastro de produtos é apresentada no Quadro 21.

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
ProdutoID	Integer	Não	Sim	Não	Auto-Denominado
NomeProd	String	Não	Não	Não	
QntProd	String	Não	Não	Não	
AlturaProd	Float	Não	Não	Não	
CompProd	Float	Não	Não	Não	
EspProd	Float	Não	Não	Não	
PrecoProd	Float	Não	Não	Não	

Quadro 21 – Tabela produtos

O Quadro 22 apresenta a descrição dos campos da tabela para o cadastro de matéria-prima.

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
MatPrimaID	Integer	Não	Sim	Não	Auto-Denominado
NomeMP	String	Não	Não	Não	
QntMP	String	Não	Não	Não	
AlturaProd	Float	Não	Não	Não	
CompProd	Float	Não	Não	Não	
EspProd	Float	Não	Não	Não	
PrecoProd	Float	Não	Não	Não	

Quadro 22 – Tabela matéria-prima

No Quadro 23 está a descrição da tabela para o cadastro de fornecedores.

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
FornecedorID	Integer	Não	Sim	Não	Auto-Denominado
RazaoForn	String	Sim	Não	Não	
InscEstForn	String	Sim	Não	Não	
CNPJForn	String	Sim	Não	Não	
TelForn	String	Não	Não	Não	
EndForn	String	Não	Não	Não	
BairroForn	String	Sim	Não	Não	
CEPForn	String	Sim	Não	Não	
CidadeForn	String	Sim	Não	Não	
EstadoForn	String	Sim	Não	Não	
PaisForn	String	Sim	Não	Não	

Quadro 23 – Tabela fornecedores

No Quadro 24 está a descrição da tabela para o cadastro de orçamentos.

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
OrcamentoID	Integer	Não	Sim	Não	Auto-Denominado
Tipo	Integer	Não	Não	Não	
NumPromissoria	Integer	Não	Não	Não	
Descricao	String	Sim	Não	Não	
Data	String	Não	Não	Não	
CodCliente	Integer	Não	Não	Sim	ClienteID
CodFuncionario	Integer	Não	Não	Sim	FuncID

Validade	Integer	Não	Não	Não	
FormaPgto	Integer	Não	Não	Não	
NumParcelas	Integer	Sim	Não	Não	
DiasIntervalo	Integer	Sim	Não	Não	
ParcelaUm	Integer	Sim	Não	Não	
ValorTotal	Float	Não	Não	Não	
Desconto	Float	Não	Não	Não	

Quadro 24 – Tabela orçamentos

No Quadro 25 está a descrição da tabela Produto Vendas/Orc.

Campo	**Tipo**	**Nulo**	**Chave Primária**	**Chave Estrangeira**	**Observações**
Est_ProdID	Integer	Não	Sim	Não	Auto-Denominado
OrcamentoID	Integer	Não	Não	Sim	OrcamentoID
ProdutoID	Integer	Não	Não	Sim	ProdutoID
Quantidade	Float	Não	Não	Não	

Quadro 25 – Tabela Produto Vendas/Orc

No Quadro 26 está a descrição da tabela Conta.

Campo	**Tipo**	**Nulo**	**Chave Primária**	**Chave Estrangeira**	**Observações**
ContaID	Integer	Não	Sim	Não	Auto-Denominado
Tipo	String	Não	Não	Não	
Data	String	Não	Não	Não	
DataVcto	String	Não	Não	Não	
VlrTotal	Float	Não	Não	Não	
LocalPgto	String	Não	Não	Não	
Cedente	String	Não	Não	Não	
VlrPgo	Float	Sim	Não	Não	
Juros	Float	Sim	Não	Não	
DataPgto	String	Sim	Não	Não	
Acrescimos	Float	Sim	Não	Não	
Descontos	Float	Sim	Não	Não	

Quadro 26 – Tabela Conta

4.3 APRESENTAÇÃO DO SISTEMA

O sistema é iniciado com uma tela de *login* para o funcionário entrar com seu nome de usuário e senha. Após a validação dos dados do funcionário, o sistema apresenta o formulário principal. Nessa tela, um menu de botões para o usuário escolher a opção desejada é disponibilizado. A Figura 4 apresenta a tela principal.

Figura 4 – Tela inicial

As telas de cadastro funcionam de maneira semelhante: o usuário preenche o formulário de cima para baixo, campo após campo. Nos campos de cidade, estado e país, o Brasil já vem selecionado como país por padrão. O usuário precisa informar primeiramente o estado (para que o sistema habilite o campo cidade) e depois escolher a cidade desejada.

As Figuras 5 e 6 apresentam os formulários para cadastro de cliente pessoa física e pessoa jurídica, respectivamente.

Figura 5 – Formulário de registro tipo alterar

O cadastro de pessoa física, Figura 6, em edição para inserção de um novo cliente.

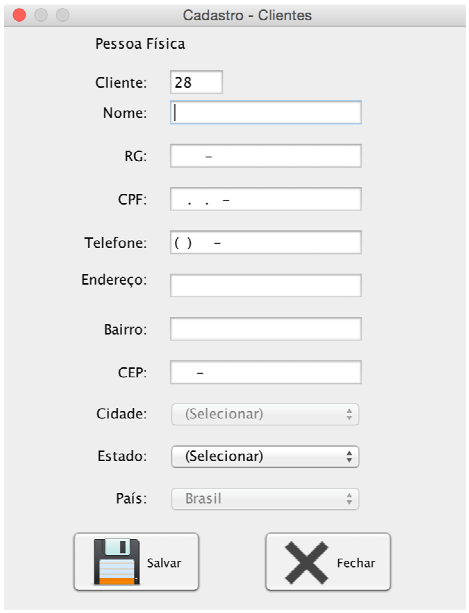


Figura 6 – Formulário de registro tipo novo

A tela de Orçamento/Venda é a principal funcionalidade do sistema. Nela o usuário informa os dados que serão necessários para gerar um novo orçamento ou uma nova venda.

Primeiramente o usuário seleciona qual o tipo do formulário. Se o tipo selecionado for "Venda com Promissória", o formulário gera um novo número de promissória à partir dos já existentes no banco de dados. Depois disso, o usuário deve preencher os demais campos do formulário: observação, forma de pagamento (se for "a prazo" informar data da primeira parcela e número de parcelas), validade, cliente, funcionário e produto. Para adicionar os produtos existe uma tabela na qual, depois de pesquisar o produto desejado, o usuário clica no botão "Adicionar". Para remover produtos cadastrados, basta selecionar o produto e clicar no botão "Remover". Por fim, ainda há a opção de reajustar o valor total com o desconto. A Figura 7 mostra a tela de orçamentos/vendas.

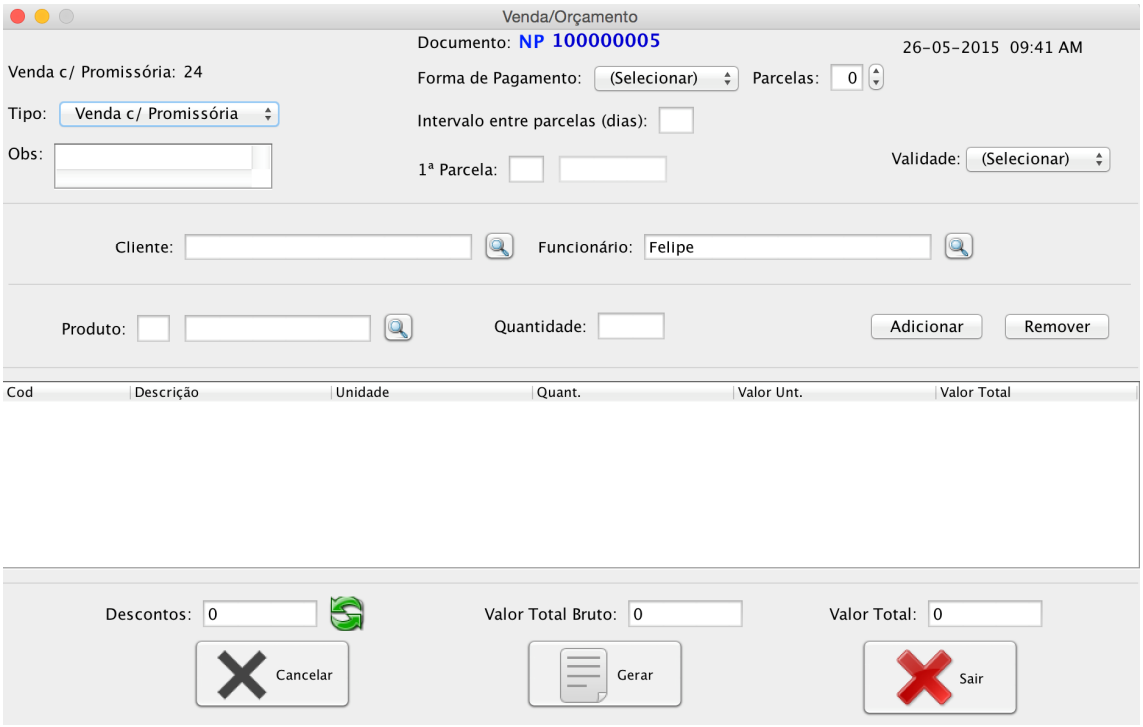


Figura 7 – Formulário de orçamentos e vendas

Quando o formulário estiver preenchido, o usuário pode finalizar a venda ou orçamento clicando no botão "Gerar". Deste modo, o formulário de impressões é aberto e o usuário seleciona as opções desejadas. A Figura 8 mostra a tela de impressões.

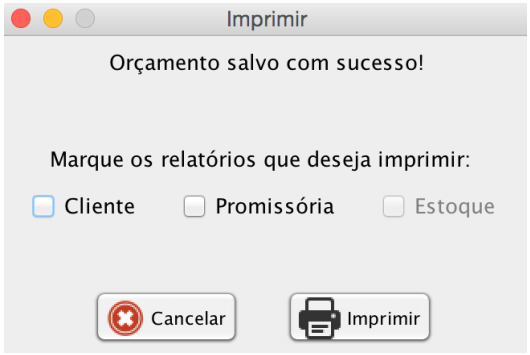
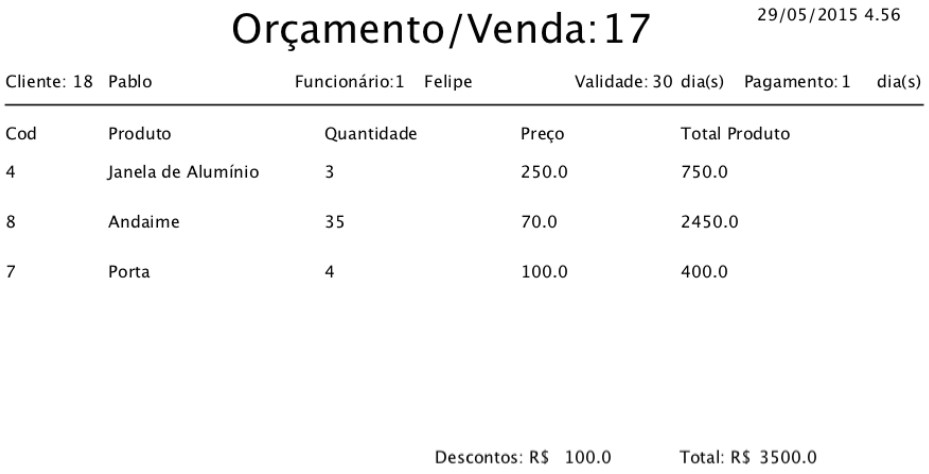


Figura 8 – Tela de impressões

Na tela de impressão, o usuário pode escolher o tipo de relatório que deseja imprimir. Assim que o botão Imprimir é clicado, os relatórios escolhidos são mostrados na tela, no formato pdf. O usuário tem a opção de imprimir ou salvar o relatório sendo visualizado. Os relatórios foram feitos com o uso do *framework* iReport. A Figura 9 mostra um relatório de venda pronto.



Orçamento/Venda: 17 29/05/2015 4.56

Cliente: 18 Pablo Funcionário: 1 Felipe Validade: 30 dia(s) Pagamento: 1 dia(s)

Cod	Produto	Quantidade	Preço	Total Produto
4	Janela de Alumínio	3	250.0	750.0
8	Andaime	35	70.0	2450.0
7	Porta	4	100.0	400.0

Descontos: R$ 100.0 Total: R$ 3500.0

Figura 9 – Relatório de venda de produto

A tela de relatórios apresenta um menu para o usuário selecionar o relatório desejado. Ao escolher uma opção no menu, um arquivo pdf é gerado e automaticamente mostrado na tela. A Figura 10 mostra a tela de relatórios.

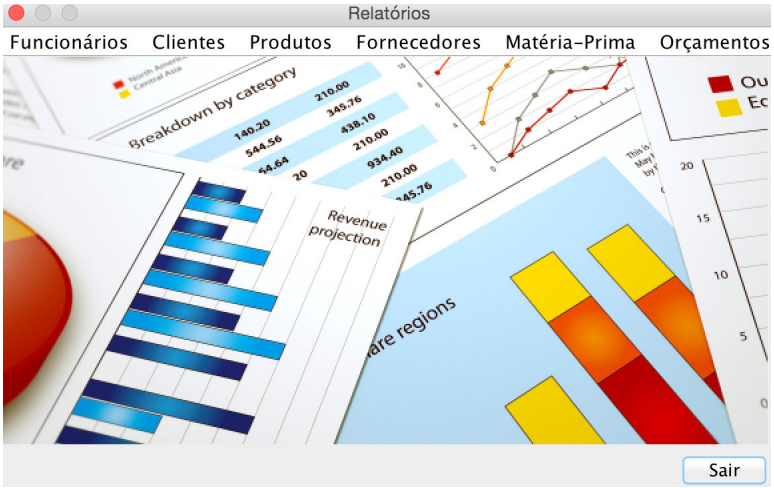


Figura 10 – Tela de relatórios

A operação de Pesquisar Orçamento/Vendas é composta por duas tabelas, bem como botões de edição, exclusão, impressão e fechar (voltar ao menu principal). Quando o usuário clica no botão “Todos”, as tabelas de vendas e orçamentos são preenchidas com todos os itens já cadastrados, em suas respectivas tabelas (Como mostra a Figura 11). O usuário também tem a opção de pesquisar por nome ou código do cliente, ou ainda, por data do orçamento/venda.

Figura 11 – Tela de pesquisa

Na mesma tela, o usuário tem a possibilidade de editar o registro selecionado, o que faz o sistema abrir a tela de Orçamento/Venda com os campos preenchidos. O usuário tem, ainda, as opções de imprimir os relatórios do registro selecionado ou excluir algum registro. A Figura 11 mostra a tela de opções.

A tela de Contas a Pagar é composta por uma tabela, bem como filtros de pesquisa, botões para criar uma nova conta ou pagar uma conta já existente e informações gerais. Quando o usuário seleciona a opção “Pendentes”, a tabela é preenchida com todos os registros de contas ainda não pagas. Quando a opção “Pagas” é selecionada, a tabela é preenchida com os registros contendo data de pagamento. Além disso, o usuário pode ordenar a tabela de diversas formas, como, por exemplo, por data ou valor.

Na parte inferior da tela ficam as informações dos dados mostrados na tabela. A Figura 12 mostra a tela de Contas a Pagar.

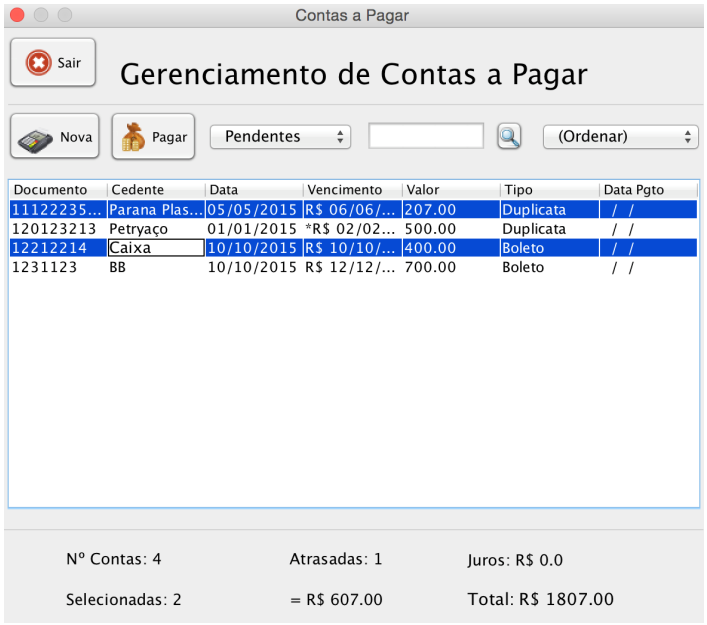


Figura 12 – Tela de contas a pagar

Quando o botão "Nova" é clicado, o sistema abre um novo formulário para cadastro de uma nova conta. Para pagar ou editar uma conta já existente, o usuário seleciona o item desejado na tabela e clica em "Pagar". O mesmo formulário é mostrado, porém desta forma o formulário é preenchido com as informações cadastradas anteriormente. A Figura 13 mostra o formulário Conta.

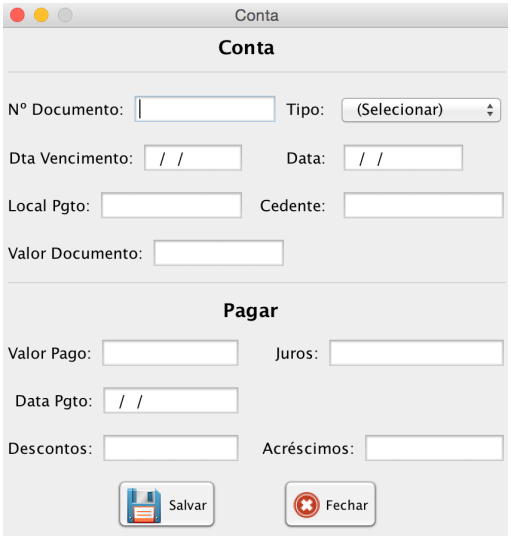


Figura 13 – Formulário de contas

4.4 IMPLEMENTAÇÃO DO SISTEMA

A organização da implementação do sistema foi baseada no padrão MVC (*Model-View-Controller* ou Modelo-Visão-Controlador). Esse padrão de arquitetura de software divide o programa em camadas, fazendo com que o código fique estruturado de maneira a facilitar o entendimento e o reaproveitamento. O *Model* representa os dados da aplicação, ou seja, os dados de um cliente, por exemplo. A *View* é a interface gráfica por meio da qual o usuário interagirá com o sistema. O *Controller* faz a lógica do sistema, recebe os dados e também os devolve para o usuário. É o *Controller* que faz a intermediação da *View* com o *Model*.

Para o sistema proposto, o *Model* é representado pelas classes *bean*, a *View* é representada pelos formulários e telas e o *Controller* é representado pelas classes DAO. A Figura 14 mostra a apresenta do projeto.

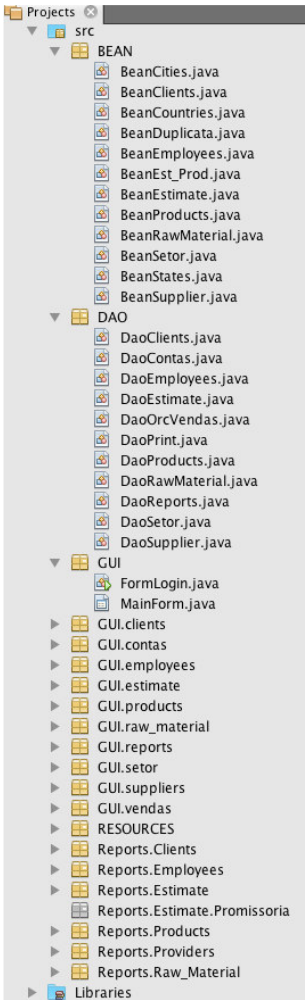


Figura 14 – Estrutura do projeto

A seguir é apresentada a implementação de algumas das funcionalidades do sistema, com o objetivo de exemplificar o uso das tecnologias. Primeiramente será apresentado o classe *bean* dos clientes, que representa o *Model* desta operação. Depois a classe DAO dos clientes é apresentada, seguida da apresentação de alguns de seus métodos. A classe DAO representa o *Controller* desta operação. A *View* desta operação foi apresentada anteriormente, nas Figuras 5 e 6.

a) Estrutura de um cadastro

Inicialmente é criada uma classe no formato *bean* com os campos que definem a classe encapsulados e somente sendo acessados por meio dos métodos disponíveis. Ou seja, os serviços providos pela classe definem o seu comportamento. Desta forma, quando um objeto daquela classe é instanciado, a única forma de acessar os campos que contém as informações da classe é pelos seus próprios métodos, o que os torna protegidos de alterações indevidas. Figura 14 mostra a criação da classe Clientes.

```
1   package BEAN;
2
3   public class BeanClients
4   {
5       private int cod;
6       private String nome;
7       private String ie;
8       private String cnpj;
9       private String rg;
10      private String cpf;
11      private String telefone;
12      private String endereco;
13      private String bairro;
14      private String cep;
15      private String cidade;
16      private String estado;
17      private String pais;
18      private int funcID;
19      private String tipo;
20
21      public int getCod() {
22          return cod;
23      }
24
25      public void setCod(int cod) {
26          this.cod = cod;
27      }
28
29      public String getNome() {
30          return nome;
31      }
32
33      public void setNome(String nome) {
34          this.nome = nome;
35      }
36
37      //...
```

Figura 14 – Classe Clientes

b) Data Access Object (DAO)

Em seguida, uma classe DAO é criada para cada *bean*. Os métodos da classe DAO trabalham com o acesso de dados, ou seja, fazem as consultas, inclusões, exclusões e todas as outras funções ligadas ao Banco de Dados. A Figura 15 mostra o cabeçalho dos métodos da classe DAO dos clientes e em seguida eles são explicados.

```
31   //**********************************connect()**********************************
32       public void connect()
33   ⊞   {...44 lines }
77
78   //****************************println(Object)*****************************
79   ⊞   public static void println(Object o) {...3 lines }
82
83   //************************insertRecord(BeanClients)************************
84       public boolean insertRecord(BeanClients client)
85   ⊞   {...31 lines }
116
117  //***************************deleteRecord(int)*****************************
118      public boolean deleteRecord(int cod)
119  ⊞   {...17 lines }
136
137  //************************updateRecord(BeanClients)************************
138      public boolean updateRecord(BeanClients client)
139  ⊞   {...32 lines }
171
172      //************************populateCountries()*************************
173      public ArrayList populateCountries()
174  ⊞   {...26 lines }
200
201  //***************************populateStates(String)************************
202      public ArrayList populateStates(String pais)
203  ⊞   {...29 lines }
232
233  //**************************populateCities(String)*************************
234      public ArrayList populateCities(String state)
235  ⊞   {...31 lines }
266
267  //*********************************list()*********************************
268      public ArrayList list()
269  ⊞   {...25 lines }
294
295  //**************************findNameByCod(int)*****************************
296      public String findNameByCod(int tfCod)
297  ⊞   {...27 lines }
324
325  //***************************findRecordModify(int)*************************
326      public BeanClients findRecordModify(int tfCod)
327  ⊞   {...41 lines }
368  }
369
```

Figura 15 – Métodos da classe DaoClientes

Explicação dos métodos da classe DaoClientes, constantes na Figura 15:

connect( ) - faz a conexão do sistema com o banco de dados e cria os objetos PreparedStatement que serão usados por outros métodos para executar as *queries* no banco de dados.

Para fazer a conexão com o banco de dados, primeiramente o sistema instancia a classe do Driver jdbc:

```
Class.forName("com.mysql.jdbc.Driver");
```

Em seguida uma nova conexão é criada, passando a url, usuário e senha do banco de dados como parâmetros:

```
con = DriverManager.getConnection(url, user, password);
```

Depois da conexão ser realizada o sistema esta apto a executar comandos SQL. Neste sistema os comandos SQL são executados através de instâncias criadas da classe PreparedStatement. Os '?' são os parâmetros que serão passados para a query, neste caso, os valores à serem inseridos:

```
pstmtInsert = con.prepareStatement("INSERT INTO client VALUES (?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");
```

insertRecord(BeanClients) - recebe um objeto do tipo BeanClients e executa as *queries* de inserção no banco de dados, usando PreparedStatement.

Esta função primeiramente passa os parâmetros para a instancia de PreparedStatement. Serão passados 14 valores para formar a query:

```
pstmtInsert.setInt(1, client.getCod());
pstmtInsert.setString(2, client.getNome());
//...demais parâmetros ocultados
```

Depois disso a *query* é executada. Se todos os campos não nulos forem repassados, o cliente é salvo no banco de dados e o valor booleano *true* é retornado. Se não, a exceção é tratada e o sistema continua trabalhando normalmente.

```
pstmtInsert.executeUpdate();
return true;
```

deleteRecord(int) - recebe o código do elemento e o exclui do banco de dados. Também é usado PreparedStatement. Mesmo procedimento do método anterior, porém para excluir um registro.

updateRecord(BeanClients) - recebe um objeto do tipo BeanClients e executa as *queries* de atualização no banco de dados, também usando PreparedStatement.

populateCountries( ), populateStates( ) e **populateCities( )** - os três métodos também utilizam PreparedStatement para fazer consultas no banco de dados, mas neste caso somente consultam as tabelas de país, estado e cidade. Depois que as consultas são feitas, os dados são salvos em um objeto de ArrayList e retornados pelo método.

list() - retorna um ArrayList com todos os dados cadastrados na tabela específica (neste caso na tabela de Clientes). Este método é utilizado para formar as tabelas nos formulários de listar cadastros. Também usa PreparedStatement para fazer a consulta.

findNameByCod(int) - recebe um inteiro com o código do registro e retorna uma *string* com o nome do cadastro que possui aquele código. Também usa PreparedStatement para fazer a consulta.

findRecordModify(int) - seleciona um cliente por meio do código para modificá-lo. Também usa PreparedStatement para fazer a consulta. Os dados são passados para uma instancia de ResultSet,. Retorna um objeto com os dados do cliente:

```
pstmtSelectModify.setInt(1, tfCod);   //parâmetro passado para a query
ResultSet rs = pstmtSelectModify.executeQuery();    //O ResultSet armazena os dados
                                                    //recuperados
if (rs.next()) //Posiciona o ponteiro no primeiro elemento do ResultSet se existir.
{
   BeanClients client = new BeanClients(); //Cria uma instancia do bean cliente
   client.setCod(rs.getInt("CLIENTEID"));
   client.setNome(rs.getString("NOME"));
   //...Campos ocultados. O mesmo ocorre para todos os outros setters do bean client.
   return client;    //Retorna o bean do cliente preenchido.
}
else
{
   JOptionPane.showMessageDialog(null, "Registro não encontrado!");
   return null;
}
```

println(Object) – recebe um objeto de qualquer tipo, pois toda classe tem a classe Object como super classe e o imprime no console. Para uma *string*, funciona como o System.out.println(). Para qualquer outro objeto é o mesmo que chamar o método toString().

Todas as exceções são tratadas pelos blocos try-catch.

c) Formulários Grafical User Interface (GUI)

A maioria dos formulários gráficos de interface visual foram feitos utilizando o editor visual de interfaces do NetBeans. Alguns foram codificados manualmente, para intuito de

aprendizado, porém a ferramenta do NetBeans oferece mais agilidade por gerar a maior parte do código automaticamente. Os componentes utilizados foram os da biblioteca *javax.swing*.

Quando o usuário clica nos botões no menu iniciar, um novo formulário é apresentado.

No caso dos cadastros, o formulário é composto por uma tabela com algumas informações de todos os cadastros existentes no banco de dados (do respectivo tipo - neste caso tipo Clientes), bem como botões de pesquisa, edição, exclusão, cadastro e fechar (voltar ao menu principal). A Figura 16 mostra a tela de opções.

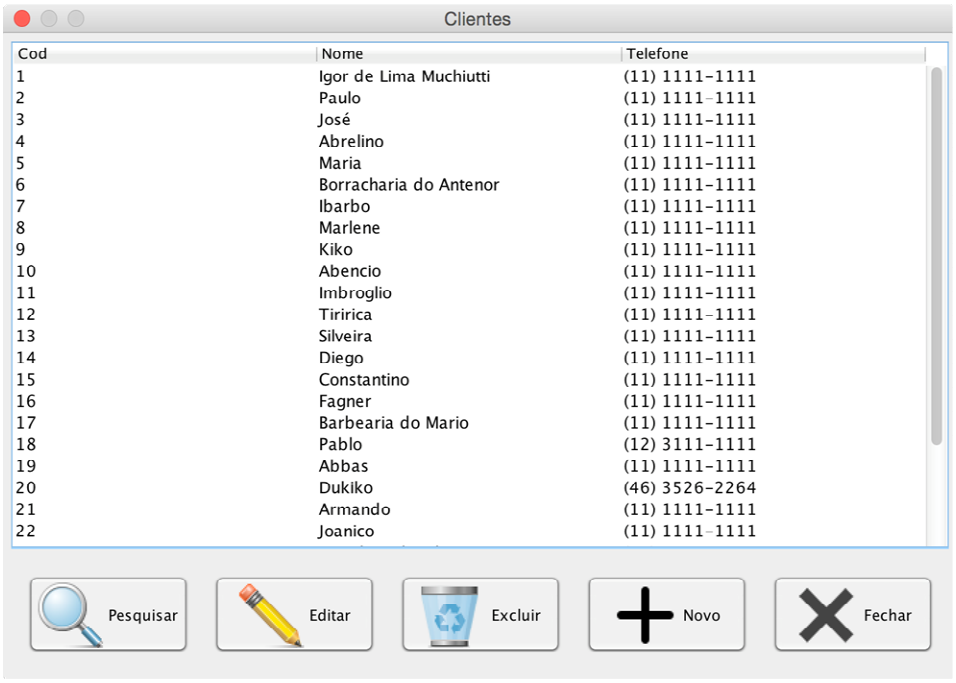


Figura 16 – Tela principal cadastro de clientes

A Figura 17 mostra o método vinculado ao botão ‘Novo’. Inicialmente uma caixa de opções pede que o usuário informe se o cliente é pessoa física ou jurídica (linhas 279-281). Logo após, o sistema cria uma nova instância do formulário de registro (linha 283), passando por parâmetro que ele é um formulário do tipo ‘New’. No caso do botão ‘Alterar’, o sistema passaria por parâmetro ‘Update’ e então montaria o novo formulário com todos os dados do cliente preenchidos. Em seguida é chamado o método para organizar o formulário (linha 285), passando por parâmetro a opção escolhida pelo usuário – pessoa física ou jurídica. Por fim, o valor ‘true’ é passado por parâmetro no método setVisible (linha 287), o que torna visível o formulário de registro.

```
276     private void btNewActionPerformed(java.awt.event.ActionEvent evt) {
277         String arr[] = { "Pessoa Física", "Pessoa Jurídica" };
278 
279         option = JOptionPane.showOptionDialog(rootPane,
280                 "Qual tipo de cliente?",
281                 "Selecionar", WIDTH, WIDTH, null, arr , "");
282 
283         FormRegisterClients formRc = new FormRegisterClients(this, "new");
284 
285         formRcSetUpNew(formRc, option);
286 
287         formRc.setVisible( true );
288     }
```

Figura 17 – Código botão novo

Como nos formulários de cadastro, os métodos do formulário Orçamento/Venda também trabalham com o acesso de dados por meio de uma classe DAO. Esta classe faz as consultas, inclusões, exclusões e todas as outras funções ligadas ao Banco de Dados. Para trabalhar com os dados de um Orçamento ou Venda, o sistema cria uma objeto da classe Orc_Ven no formato *bean*, com os campos que definem a classe encapsulados e somente sendo acessados por meio dos métodos disponíveis (*getters e setters*). A seguir é apresentado o cabeçalho dos principais métodos do formulário Orçamento/Venda.

public FormEstimate(int cod) – É um dos métodos construtores da classe deste formulário. Recebe o código de um orçamento por parâmetro e chama o método fillFormUp(int cod) para ajustar o formulário. Além disso, ele também cria os componentes do formulário. É acionado quando o usuário edita algum item na tela Pesquisar Orçamentos/Vendas.

```
public FormEstimate(int cod)
   {
      initComponents();   //cria todos os compontentes do formulário.
      dao = new DaoEstimate();   // cria uma nova instância do método Dao Orçamento
      daoP = new DaoProducts();   // cria uma nova instância do método Dao Produtos
      dao.connect(); //chama o método connect para fazer a conexão com o banco.
//Centraliza a tela
      Toolkit toolkit = Toolkit.getDefaultToolkit();
      Dimension tela = toolkit.getScreenSize();
      this.setLocation(tela.width/2 - this.getWidth()/2,
            tela.height/2 - this.getHeight()/2);
```

```
        fillFormUp(cod); //Preenche o formulário com o cadastro recuperado.
    }
```

public FormEstimate(MainForm mainForm) – É outro método construtor da classe deste formulário. Cria os componentes do formulário e chama o método makeForm( ) para ajustar o formulário. É acionado quando o usuário clica no botão Orçamento/ Venda do menu principal. O código é praticamente o mesmo do método anterior.

public void fillFormUp(int cod) – Recebe o código de um orçamento por parâmetro, que será buscado no banco de dados para assim o formulário ser preenchido com as informações do orçamento ou venda que tenha aquele código.

public void makeForm() – Configura um novo formulário.

int findUnusedCod () – Busca por algum código ainda não utilizado.

public void addAtTable() – Adiciona um produto na tabela.

public boolean updateEst_Prod() – Salva os produtos da tabela na tabela Est_Prod do banco de dados. Retorna um valor booleano.

public boolean setUpBeanEst_Prod() – Configura o *bean* dos produtos adicionados a tabela para passá-lo para algum método da classe DAO e, assim, salvar no banco de dados. Retorna um valor booleano.

public BeanEstimate setUpBeanEstimate() – Configura o *bean* do Orçamento/Venda e o retorna.

O formulário de Contas a Pagar também trabalha com o acesso de dados por meio de uma classe DAO e com o auxilio de classes no formato *bean* para trabalhar com os objetos. Os cabeçalhos dos principais métodos do formulário Contas a Pagar são apresentados a seguir.

public FormCP() – Método construtor do formulário. Cria e ajusta os componentes do formulário.

public BeanDuplicata fillFieldsUp() – Busca o registro pelo número e o retorna para a tela de Pagamento/Edição ser preenchida.

public void searchCP() throws ParseException – Procura o registro pelo número ou nome do cedente digitado no campo de texto e o mostra na tabela.

public void updateLabels() – Atualiza as informações da parte inferior da tela.

public void resetForm() – Reseta os componentes da tela.

public void fillTableUp(int filtro) throws ParseException – Preenche a tabela. É acionado quando o usuário muda os filtro do formulário.

O formulário de Pesquisar Orçamento/Vendas também trabalha com o acesso de dados por meio de uma classe DAO e com o auxilio de classes no formato *bean* para trabalhar com os objetos. A seguir está o cabeçalho dos principais métodos do formulário Pesquisar Orçamento/Vendas:

public FormOrcamentosVendas(MainForm mainForm) – Método construtor do formulário. Cria e ajusta os componentes do formulário.

public static void makeTable(int tipo) – Preenche a tabela. O tipo passado por parâmetro depende de qual filtro o usuário esta solicitando a informação.

public void print() – Abre a tela do orçamento ou venda selecionada, no formato pdf.

public void editOrc_Venda() – Abre a tela de Orçamento/Venda com o registro selecionado preparado para edição.

public void deletaOrc_Venda() – Exclui o registro selecionado.

public static void resetComponents(int count1, int count2) – Atualiza os componentes para que fiquem preparados para receber os novos registros.

5 CONCLUSÃO

Os objetivos pretendidos com este trabalho foram alcançados. Um sistema para gerenciamento de uma empresa metalúrgica de pequeno porte foi desenvolvido. Os principais processos de negócio dessa empresa foram automatizados visando facilitar, entre outras, as atividades de geração de orçamento e vendas, bem como de compras.

A realização deste trabalho possibilitou aprendizado por meio das dificuldades encontradas com o desenvolvimento de atividades que abrangem todo o ciclo de vida de software.

Desenvolver um sistema que será utilizado por uma empresa é diferente do desenvolvimento com objetivos didáticos. Um sistema para ser utilizado tem um cliente com necessidades, interesses e expectativas reais. Isso no sentido que eles estão vinculados aos requisitos e regras de negócio da empresa. E, assim, os programas deixam de ser um conjunto de estruturas de decisão e repetição para uma composição de instruções que atendam necessidades da empresa.

O desenvolvimento do sistema também auxiliou a evidenciar a importância da análise e a respectiva modelagem. E permitiu visualizar a utilização dos diagramas produzidos na fase de análise e projeto.

Em termos de banco de dados, o trabalho realizado permitiu ressaltar a importância da correta e adequada definição das tabelas e dos seus campos. E, que isso pode não ser uma tarefa realizada em uma única iteração. É preciso complementar, acrescentar e, por vezes, refazer tabelas inteiras.

As tecnologias utilizadas permitiram uma visão bastante ampla do desenvolvimento de software. Contudo, o desenvolvimento em si permitiu implementar as funcionalidades tendo em mente prover facilidades para os usuários. Assim, além dos aspectos técnicos, o desenvolvimento deve considerar a interação do usuário com o sistema e o atendimento dos requisitos de negócio por meio das funcionalidades do sistema.

Como trabalhos futuros destacam-se o desenvolvimento de um aplicativo *web* com acesso diferenciado pela empresa e por clientes. E, ainda, uma versão para dispositivos móveis também poderia ser implementada facilitando, assim, o acesso aos dados de gestão.

REFERÊNCIAS

ALBERTIN, Alberto L. Valor estratégico dos projetos de tecnologia de informação. **Revista de Administração de Empresas,** *v. 41, n.* 3, p. 42-50, 2001.

ALBERTIN, Alberto L.; ALBERTIN, Rosa Maria M. **Tecnologia de informação e desempenho empresarial no gerenciamento de projetos de TI.** Anais do Encontro Nacional da Associação Nacional de Pós-Graduação e Pesquisa em Administração, Rio de Janeiro, RJ, Brasil, 31, 2007.

ALBERTIN, Alberto L; ALBERTIN, Rosa Maria M. **Tecnologia de informação e desempenho empresarial: as dimensões de seu uso e sua relação com os benefícios de negócio.** 2 ed. São Paulo: Atlas, 2009.

BECCHETTI, Leonardo; PAGANETTO, Luigi; BEDOYA, David A. L. **ICT investment, productivity, and efficiency: evidence at firm level using a stochastic frontier approach**. Journal of Productivity Analysis, n. 20, p. 143 - 167, 2003.

CANUTO, Kleber C.; CHEROBIM, Ana Paula M. S. **Grau de informatização e desempenho: um estudo em organizações brasileiras de capital aberto**. In: 33 Encontro Nacional da Associação Nacional de Pós-Graduação e Pesquisa em Administração, São Paulo, SP, Brasil, p. 1-16, 2009.

DAVIS, Fred D. Perceived usefulness, perceived ease of use, and user acceptance of information technology. **MIS Quarterly**, v. 13, n. 3, p. 319-340, 1989.

FETZNER, Maria Amélia M.; FREITAS, Henrique M. R. **Repensando questões sobre mudança, afeto e resistência na implementação de SI.** In: Encontro Nacional da Associação Nacional de Pós-Graduação e Pesquisa em Administração, São Paulo, SP, Brasil, v. 18, n. 1, 2012.

KEARNS, Grover S.; SABHERWAL, Rajiv Strategic alignment between business and information technology: a knowledge-based view of behaviors, outcome, and consequences. **Journal of Management Information Systems***,* v. 23, n. 3, p. 129-162, 2006.

LUNARDI, Guilherme L.; DOLCI, Pietro C.; MAÇADA, Antônio Carlos G.. Adoção de tecnologia de informação (TI) e seu impacto no desempenho organizacional: um estudo realizado com micro e pequenas empresas. **R. Adm.**, São Paulo, v.45, n.1, p. 5-17, jan./fev./mar. 2015.

MAÇADA, Antonio Carlos G. **Impacto dos investimentos em tecnologia da informação nas variáveis estratégicas e na eficiência dos bancos brasileiros**. Tese de Doutorado. PPGA/EA/UFRGS, 2000.

MALAQUIAS, Rodrigo F.; ALBERTIN, Alberto L. Por que os gestores postergam investimentos em tecnologia da informação? Um Estudo de Caso. **RAC**, Curitiba, v. 15, n. 6, art. 8, p. 1120-1136, Nov./Dez. 2011, p. 1120-1136.

MORAES, Giancarlo M.; BOBSIN, Débora; LANA, Francielle V. D. **Investimentos em tecnologia da informação e desempenho organizacional: uma busca do estado da arte.** In: 30 Encontro Nacional da Associação Nacional de Pós-Graduação e Pesquisa em Administração, Salvador, BA, Brasil, 2006, p. 1-16.

NEVO, Saggi; WADE, Michael R. The formation and value of IT – enabled resources: antecedents and consequences of synergistic relationships. **MIS Quarterly**, v. 34, n. 1, p. 163-183, 2010.

PRESSMAN, Roger. **Engenharia de software**. Rio de Janeiro: McGraw-Hill, 2005.

RUGGIERO, Alberto P.; Godoy, Arilda S. A influência da tecnologia de informação no trabalho gerencial: um estudo com gestores de recursos humanos. **REAd**. v. 12, n. 1, 2006, p. 160-181.

OLIVEIRA, Mírian; OLIVEIRA, Leonardo Rocha de; HANSEN, Peter Bent; GASPAROTE, Maurício. **Governança em TI e Competitividade do Arranjo Produtivo Local Coureiro-Calçadista do Rio Grande do Sul**. 2005. Disponível em: <http://www.anpad.org.br/diversos/trabalhos/EnANPAD/enanpad_2005/ADI/2005_ADID2673.pdf>. Acesso em: 15 fev. 2015.

SACILOTTI, Adaní C. **A importância da tecnologia da informação nas micro e pequenas empresas: um estudo exploratório na região de Jundiaí**. Dissertação (mestrado), Faculdade Campo Limpo Paulista – FACCAMP, 2011.

SILVA, André Luiz M. R.; DIAS, Donaldo S. **Influência do treinamento de usuários na aceitação de sistemas ERP no Brasil**. In: 30 Encontro Nacional da Associação Nacional de Pós-Graduação e Pesquisa em Administração, Salvador, BA, Brasil, 2006, p. 1-17.

SOUZA, Cesar. A.; ZWICKER, Ronaldo; VIDAL, Antonio Geraldo da R.; SIQUEIRA, José de O. **Avaliação do grau de informatização de empresas: um estudo em indústrias brasileiras**. In: 29 Encontro Nacional da Associação Nacional de Pós-Graduação e Pesquisa em Administração, Salvador, BA, Brasil, 2005, p. 1-16.

SOUZA, Cesar Alexandre de; ARPINO, Giuseppe. **TI e eficiência organizacional: um estudo no setor brasileiro de bens de capital mecânicos com foco em micro, pequenas e médias empresas**. Prod. v. 21, n. 4 , 2011. Disponível em: < http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-65132011000400016>. Acesso em: 09 fev. 2015.

SOUZA; Cesar A. de; SZAFIR-GOLDSTEIN, Cláudia. Tecnologia da informação aplicada à gestão empresarial: um modelo para a empresa digital. **VI SEMEAD /FEA-USP**, v. 4, n. 22,

2005. Disponível em: <http://www.cyta.com.ar/ta0404/v4n4a1.htm>. Acesso em: 25 fev. 2015.

VENKATESH, Viswanath; BALA, Hillol. **Technology acceptance model 3 and a research agenda on interventions**. Decision Science, v. 39, n. 2, p. 273-312, may. 2008.