

Tecnologia Java no desenvolvimento de Sistema de Vendas para Controle de Caixa e Estoque

Jean H. Zenzeluk¹, Wilian D. Pasternak², Elena M. Bini³

^{1,2,3}Tecnologia em Análise e Desenvolvimento de Sistemas
Faculdade Guairacá - Guarapuava - Paraná

jeanzenzeluk@hotmail.com, wilianpasternak@gmail.com,
elenamariielebini@gmail.com

Abstract. *In this article, are presented the steps taken to develop a desktop system that aims to manage sales, employees and products of a company. For the develop the software standards were used Model-view-controller (MVC) pattern and Data Access Object (DAO), and the Java programming language, coupled with JasperReports and Hibernate frameworks, and MySQL database.*

Resumo. *Neste artigo, são apresentadas as etapas realizadas para o desenvolvimento de um sistema desktop que tem como objetivo o gerenciamento de vendas, funcionários e produtos de uma empresa. Para o desenvolvimento do software, foram empregados os padrões Model-view-controller (MVC) e Data access object (DAO), e a linguagem de programação Java, aliada aos frameworks Hibernate e JasperReports, e o banco de dados MySQL.*

Introdução

Em empresas do ramo de vendas, muitos são seus colaboradores e os processos envolvidos. Neste ambiente, sistemas de informação são necessários para auxiliar na gestão dos dados para posterior tomada de decisão.

Por empresas deste ramo atuarem principalmente com movimentação de produtos e dinheiro, um sistema de informação gerencial torna-se um fator essencial para manter o controle e prevenir riscos referentes a falhas humanas e/ou perda de informações.

O objetivo deste trabalho é apresentar as etapas de desenvolvimento de um *software* para auxiliar nos processos rotineiros de empresas do ramo de vendas, e ainda explicar sobre as tecnologias utilizadas para a realização do mesmo.

O *software* foi desenvolvido, utilizando a linguagem de programação *Java*, com a implementação dos *frameworks* *Hibernate* e *JasperReports*. Sua programação foi realizada com a IDE *NetBeans 7.3*, juntamente do *plugin* do *iReport 5.0*. O Banco de dados utilizado foi o *MySQL*, junto da ferramenta *MySQL Workbench*. Para a elaboração dos diagramas propostos pela *UML*, foi empregada a ferramenta *Astah Community*.

Fundamentação Teórica

De acordo com Boghi e Shitsuka (2007), os negócios evoluem constantemente, tornando-se mais complexos. Nesse contexto, a tendência é que os dados aumentem. Ainda, segundo os autores, empresas desestruturadas acabam por revelar sua fragilidade e armazenar dados inúteis e sem aproveitamento, ou ainda perder dados e informações valiosas para o sucesso das mesmas.

Segundo Laudon e Laudon (2007), as empresas enfrentam muitos desafios, problemas e riscos de perdas de dados importantes, e uma das principais maneiras de resolvê-los é a implantação de sistemas de informação.

Batista (2004) afirma que uma característica dos sistemas de informação gerencial é que os mesmos sempre estão mudando e se adaptando com as exigências do negócio.

Para Boghi e Shitsuka (2007), informação é uma coisa útil, e para gerá-la é preciso de dados que possam ser trabalhados ou organizados para ser possível a geração das informações necessárias. Os dados necessitam ser armazenados e organizados para que possam ser recuperados a partir de um sistema. Nesse cenário, Cordeiro e Freitas(2011) afirma que o *software* destaca-se como um dos principais instrumentos estratégicos para as organizações, e vem sendo utilizado por estas como instrumento de apoio às diversas atividades e à tomada de decisões.

Uma tecnologia usada para o desenvolvimento de sistemas e aplicações é a linguagem de programação *Java*. Deitel (2005) afirma que esta linguagem é utilizada para desenvolver aplicativos corporativos de grande porte, aprimorar a funcionalidade de servidores *Web*, fornecer aplicativos para dispositivos voltados para o consumo popular (por exemplo, telefones celulares, *paggers* e PDAs) e para muitos outros propósitos.

Horstmann (2005) afirma que a linguagem *Java* é poderosa por possuir uma biblioteca padrão rica, com acesso ao banco de dados, gráficos, possível construção de interface com o usuário, multitarefa, programação em rede e acesso a banco de dados. E Orlandini e Delfino (2012) afirmam também que possui portabilidade com independência de plataforma, coletor de lixo com o processo automático de deslocação da memória, permissão para execução de programas com restrições, e ainda conta com a implementação de funcionalidades utilizando vários conceitos da orientação a objetos, como: classe, método, herança, polimorfismo, encapsulamento, mensagem, entre outros, dando maior qualidade à linguagem.

Ao se trabalhar com *Java*, para Matula (2003), uma boa ferramenta de auxílio à programação é o *NetBeans*, porque esta ferramenta fornece um conjunto de módulos para apoiar o desenvolvimento *Java*, e pode basicamente servir como uma plataforma para praticamente qualquer tipo de aplicação *Java*; e é mais especificamente útil para utilização de ferramentas de desenvolvimento integradas modulares.

Nos *softwares* empresariais, normalmente existem dados que precisam ser guardados e trabalhados; e estes são armazenados no banco de dados. Segundo Boghi e Shitsuka (2007), um banco de dados são grupos de arquivos relacionados entre si; e

estes possuem registros de vários tipos de dados, como informações sobre pessoas, lugares, dentre outras.

Segundo Batista (2004) basicamente as empresas utilizam bancos de dados para monitorar transações básicas, como pedidos, pagamento a funcionários e atendimento a clientes. Também precisam de bancos de dados para obter informações que auxiliem nas tomadas de decisão e na administração do negócio de uma maneira mais eficiente.

Para trabalhar com o gerenciamento de um grande volume de dados, segundo Junior e Almeida (2012), o responsável é o Sistema de Gerenciamento de Banco de Dados (SGBD); e este, além de gerenciar, possibilita o trabalho com múltiplos usuários e transações simultâneas.

Segundo Gilfillan (2003), um SGBD bastante utilizado é o servidor de banco de dados MySQL, que incorpora muitas das funções necessárias para outros ambientes, mantendo sua alta velocidade e ainda apresentando poderosas permissões do sistema.

Segundo o Manual de Referência do MySQL 5.7 (2014), o MySQL é um servidor robusto de banco de dados rápido, multitarefa e multiusuário, e pode ser usado em sistemas com alta carga de dados. Segundo o manual, também dentre as vantagens do MySQL estão portabilidade, funcionamento em diversas plataformas, dispõe várias APIs para algumas linguagens (incluindo *Java*), suporte total a *multi-threads*, usando *threads*, diretamente do *kernel*, podendo usar múltiplos CPUs, se disponíveis, um sistema de alocação de memória rápido e baseado em processo (*thread*), dentre outras vantagens.

Conforme Elliott, O'Brien e Fowler (2009), para facilitar o trabalho com banco de dados e desenvolvimento de *software* de pequeno e médio porte é bastante utilizado o *Hibernate*. Este é um *framework* de mapeamento objeto/relacional de peso leve para *Java*, que deixa mais fácil e eficiente o trabalho com informações de banco de dados relacional na forma de objetos *Java*.

Kraemer e Vogt (2004) afirmam que o *Hibernate* oferece suporte para relacionamentos e coleções de objetos, como polimorfismo, herança e composição; também é poderoso por sua linguagem de consultas orientada a objetos por possuir uma boa camada de cache, suporte para *Java Management Extensions* (JMX) e o *Hibernate Query Language* (HQL), que tem por objetivo a recuperação de objetos do banco de dados. Segundo Elliott, O'Brien e Fowler (2009) com a API *Criteria* do *Hibernate* é possível fazer consultas no banco de dados sem a necessidade de escrever códigos SQL, pois basta adicionar restrições (*Restrictions*) e escolher o método para realizar a consulta, depois os dados trazidos podem ser consultados a partir de uma lista.

Para Filho (2012), elaboração de relatórios são muito comuns em sistemas de informação, e esta forma de apresentação de dados é bastante solicitada pelos usuários. Para facilitar a organização de dados, formatação de conteúdos e permitir a saída de relatórios em diversos formatos, uma boa opção é a utilização do *framework Jasper Reports*, juntamente da ferramenta *iReport* para auxílio na elaboração do design dos relatórios empregando código XML.

Segundo Larman (2002), para evitar futuros problemas durante o desenvolvimento de sistemas e ainda obter melhor controle e organização é

recomendável a utilização do *Unified Modeling Language* (UML), porque esta é uma linguagem visual para especificar, construir e documentar as necessidades do sistema.

Dentro do conceito de UML, existem diferentes tipos de diagramas: e um destes é o diagrama de caso de uso. Para Bezerra (2007), este corresponde a um comportamento específico que é produzido por objetos que compõem o sistema. Basicamente, este diagrama é responsável por conter um conjunto de objetos que colabore na produção do resultado do respectivo caso de uso; e ainda com base no diagrama fica mais fácil identificar as classes necessárias para o sistema.

O outro diagrama proposto pela UML é o diagrama de classe. Neste, segundo Lima (2012), é mostrada a estrutura estática do modelo, em que seus elementos são representados por classes: estas possuem seus relacionamentos e estrutura interna. Lima (2012) afirma ainda que esse diagrama pode ser útil para revisar o que é armazenado no sistema e nas estruturas de armazenamento; mas seu principal propósito é obter as classes com seus relacionamentos de associação, agregação e generalização. As associações implicam, basicamente, que dois elementos do modelo têm um relacionamento colocado como uma instância de classe, ou seja, declara que pode haver ligações entre instâncias de tipos associados.

Dentro da UML, temos também o diagrama de sequência, que, como afirma Guedes (2007), trabalha com a ordem temporal da troca de mensagens entre objetos envolvidos de algum processo; basicamente esse diagrama identifica o evento que gerou o processo modelado e o ator responsável por este evento. Com isso é determinado como o processo deve trabalhar e ser concluído por meio do envio de mensagens que normalmente disparam métodos entre os objetos.

No desenvolvimento do *software*, existem vários padrões de projeto para melhor controle, organização e compreensão do sistema a ser desenvolvido; um deles bastante utilizado é o *Model View Controller* (MVC). Para Lima (2012), esse modelo é considerado uma arquitetura padrão utilizado na Engenharia de *Software*. Ele separa a lógica da aplicação da interface do usuário, permitindo testar e desenvolver cada parte separadamente.

Carvalho (2001) afirma que o padrão MVC possui três camadas. Sendo elas: camada de visualização, que possui a parte de interface visual que recebe os dados inseridos pelo usuário e retorna as mensagens do sistema; a camada de controle define a maneira que a interface reage a uma entrada feita pelo usuário; basicamente é responsável por controlar e mapear as ações, como pelo comportamento da aplicação, interpretando as solicitações feitas pelo usuário, comunicando-se com o modelo e atualizando a interface de usuário. A outra camada é a de modelo; esta consiste no objeto da aplicação; contém toda lógica da aplicação e representa os dados e as regras de negócio.

Uma opção para prover o isolamento da tecnologia de persistência no desenvolvimento do *software* é o padrão *Data Access Object* (DAO). Segundo Elliott, O'Brien e Fowler (2009), esse padrão isola o código da aplicação do código responsável pela manipulação de registros no banco de dados. Mais especificamente esse padrão coloca todas as operações de persistência ao banco de dados em uma única interface, e ainda fornece uma implementação dessa interface que pode utilizar qualquer quantidade de diferentes API's de persistência.

Para ter a possibilidade de realizar verificações e experimentos antes de o sistema ser finalizado definitivamente, o ciclo de desenvolvimento Prototipação pode ser empregado. Segundo Lessa & Junior (2006), prototipação é basicamente a montagem de protótipos, e pode ser classificada com uma variedade de dimensões. Para Lima (2012), a prototipação traz diversas vantagens, como não ter necessidade dos requisitos serem completamente determinados antes do início, e há maior participação e interação do usuário no processo de desenvolvimento, pois o mesmo sabe o que esperar do sistema tendo maior satisfação quando o projeto for finalizado.

Segundo Pressman (2011), o paradigma de prototipação começa com a comunicação, fazendo uma reunião com os envolvidos para definir os objetivos gerais do *software*. Depois é feito um projeto rápido que representa os aspectos do *software* que serão visíveis ao usuário final. Então esse projeto rápido leva à construção de um protótipo, que é avaliado pelos envolvidos; estes fornecerão um retorno que servirá para aprimorar os requisitos. Basicamente, para Lessa & Junior (2006), com a prototipação pode-se focar mais rapidamente e diretamente na parte de funcionalidade, performance, interface com banco de dados etc.

No desenvolvimento do *software*, é importante o uso do conceito de usabilidade. Para Barbosa e Silva (2010), esse tem por objetivo eficácia, eficiência, segurança, utilidade e satisfação na interação do usuário com o sistema. O autor complementa que a eficácia está relacionada com a capacidade e facilidade na interação do usuário com o sistema a fim de realizar e alcançar seus objetivos conforme o esperado.

NIELSEN (1994) propõe um conjunto de dez qualidades. Estas ele chamou de heurísticas de usabilidade. São elas:

Tabela 1. Heurísticas de usabilidade

1. Visibilidade de Status do Sistema	A interface sempre deve informar ao usuário o que está acontecendo
2. Relacionamento entre a interface do sistema e o mundo real	Na comunicação do sistema devem ser usadas palavras que façam sentido ao usuário
3. Liberdade e controle de usuário	Facilitar saídas para o usuário, permitindo desfazer ou refazer ações no sistema
4. Consistência	Facilitar a identificação do usuário, utilizando sempre os mesmos ícones e palavras nas ações similares.
5. Prevenção de erros	Evitar situações de erro, modificando a interface para que estes não ocorram
6. Reconhecimento ao invés de lembrança	Permitir que a interface dialogue com o usuário, evitando acionar a memória do usuário o tempo inteiro
7. Flexibilidade e eficiência de uso	O sistema deve ser fácil para usuários leigos, porém flexível o bastante para se tornar ágil a usuários avançados
8. Estética e design minimalista	Evitar textos e design desnecessários, os mesmos devem ser simples, diretos e naturais
9. Ajudar os usuários a reconhecer, diagnosticar e sanar erros	As mensagens de erro do sistema devem possuir uma redação simples e clara.
10. Ajuda e documentação	Um conjunto de documentação e ajuda devem ser visualmente e facilmente acessado quando necessário para orientar o usuário em caso de dúvida

Nielsen (1994) afirma que as dez heurísticas fornecem qualidade a qualquer interface. Preece, Rogers e Sharp (2005) afirmam que usabilidade é geralmente considerada como o fator que assegura que os produtos sejam eficientes, agradáveis e de fácil utilização da perspectiva do usuário, gerando satisfação do mesmo.

A próxima seção apresenta as etapas do desenvolvimento do trabalho e as tecnologias e/ou ferramentas empregadas.

Etapas do Desenvolvimento do trabalho

O ciclo de vida que orientou o desenvolvimento do software apresentado neste artigo foi a prototipação. Assim, o primeiro passo realizado foi a coleta dos requisitos. Estes foram levantados através de diálogos com o proprietário de uma empresa da área de vendas, que se tornou parceira no processo de desenvolvimento do *software*. Durante conversas informais, foram levantados os requisitos necessários para um sistema de gerenciamento de vendas. Os requisitos levantados foram: (i) realizar vendas, (ii) cadastrar caixas, vendedores, auxiliares de estoque, clientes, produtos, departamentos, setores, seções, fornecedores e transportadoras. Outro fator foi a necessidade de geração de relatórios informativos.

Após a coleta de requisitos, foi criado um protótipo rápido, seguindo as premissas do ciclo de vida de prototipação. A ficha contendo os requisitos encontra-se no apêndice A, a tela do protótipo rápido está no apêndice B. Este protótipo rápido foi avaliado pelo usuário, e a ficha de avaliação está no apêndice C.

Após a avaliação do protótipo rápido e já com os requisitos iniciais levantados, passou-se para a criação dos diagramas propostos pela UML. Para essa modelagem, foi utilizada a ferramenta Astah e foram elaborados os seguintes diagramas: de caso de uso, o diagrama de classe e o diagrama de sequência. Os diagramas elaborados encontram-se nos apêndices D, E, F, G e H. Em seguida, com a utilização da ferramenta MySQL *Workbench*, foi iniciada a modelagem lógica do banco de dados. A modelagem encontra-se no apêndice I.

Iniciou-se a próxima fase, com a codificação do *software*, seguindo as normas do padrão MVC para o projeto e o padrão DAO no quesito de trabalho com banco de dados. Com o banco de dados modelado, foi feito o mapeamento das classes com o uso das *Annotations* do *Hibernate* para que o mesmo identifique as tabelas e seus respectivos atributos que serão persistidos no banco de dados. As *Annotations* pertencem ao pacote *Modelo*.

Depois de criadas as classes do pacote *Modelo*, nas mesmas foram implementados e substituídos alguns métodos, como métodos construtores e *toString*. Com as classes de modelo terminadas, foram criadas as classes DAO, pertencentes ao pacote *Controle*, responsáveis pelos métodos salvar, excluir, alterar e listar. Para as consultas no banco de dados, foi bastante utilizada a API Criteria e o Hibernate Query Language (HQL) por facilitar as consultas no banco de dados.

Feito isso, iniciou-se o desenvolvimento das telas do sistema. Estas ficam no pacote *Visão* e foram desenvolvidas com a ferramenta *NetBeans* trabalhando na maioria dos casos com *layouts* de conjunto de grade. As telas foram padronizadas para maior facilidade de uso e menos memorização por força bruta do usuário. Basicamente as telas

de cadastro possuem na parte superior as informações gerais; no meio, uma tabela, contendo os registros do banco de dados e, no canto inferior direito da tela, encontram-se os botões. Depois da avaliação de um protótipo foram feitas telas de pesquisa de acesso rápido onde apenas são listados os dados para o usuário. Nessa tela, os campos de busca e o botão pesquisar ficam na parte superior da tela; e abaixo fica uma tabela responsável por apresentar os registros. Ainda foram disponibilizados temas para escolha do usuário. O padrão seguido pelas telas de cadastro pode ser visualizado na figura 1.

id	Nome	CPF/CNPJ	RG	Telefone 1	Data Nascimento	Data Cadastro
1	Cliente Teste	080 080 080-88	126889968	(44) 8888-8888	10/10/1994	01/03/2014
2	Cliente Teste	080 080 080-98	126856894	(22) 4444-5555	10/10/1994	01/01/2014
3	Cliente Teste	546 546 545-64	125789654	(51) 5645-4444	10/10/1994	03/03/2015
4	Cliente Teste	408 488 804-84	115254878	(58) 5849-84894	10/10/1994	11/03/2014
5	Cliente Teste	846 846 846-46	125566898	(84) 8464-6464	10/10/1994	01/03/2014
6	Cliente Teste	850 850 505-05	125632244	(85) 0858-50858	10/10/1994	01/03/2014
7	Cliente Teste	213 123 123/1232-11	115789556	(41) 6516-16516	10/10/1994	02/03/2014
8	Cliente Teste	850 085 085-08	126588999	(85) 0850-8888	10/10/1994	20/03/2014
9	Cliente Teste	408 468 486-40	112547778	(84) 6840-64846	10/10/1994	02/03/2014
10	Cliente Teste	414 004 468/4684-68	126889968	(84) 8489-01681	10/10/1994	16/03/2014

Figura 1. Tela de clientes.

Dentre as Heurísticas de usabilidade estão presentes a visibilidade de Status do Sistema, o relacionamento entre a interface do sistema e o mundo real, a consistência utilizando os mesmos ícones e palavras nas ações similares, a prevenção de erros, o reconhecimento ao invés de lembrança, flexibilidade e eficiência de uso, estética e design minimalista e, por último, ajuda para o usuário na compreensão dos erros. As telas, comprovando a implementação dessas heurísticas, encontram-se nos apêndices I e J.

Depois foram desenvolvidos os relatórios, por estes normalmente serem essenciais para as empresas que utilizam softwares empresariais. Relatórios são o resultado das informações trabalhadas no sistema, e para auxiliar no desenvolvimento destes foi empregado o uso da biblioteca *Jasper Reports* junto da ferramenta *iReport*, que serve para automatizar o processo da elaboração do desenho do relatório empregando código XML. Basicamente esta ferramenta permite a definição do design do relatório de um modo gráfico e depois gera o código XML automaticamente, sem a necessidade de utilizar diretamente estes códigos.

Após a finalização da codificação do sistema, foi realizado uma nova avaliação. Neste momento, quatro usuários participaram dos testes e das validações. Erros foram diagnosticados e listados pelos usuários que realizaram os testes; e o proprietário da

empresa citou as alterações desejadas no sistema. A quatro ficha de Avaliação dos erros encontrados estão nos apêndices L,M, N e O, e a ficha de avaliação contendo as alterações desejadas encontra-se no apêndice P. Tais erros foram corrigidos, e as alterações desejadas foram implementadas para a versão final do sistema. A ficha de avaliação final do sistema encontra-se no apêndice Q.

Resultados

Os processos das empresas que atuam na área de vendas devem ser cuidadosamente organizados e seguros. Ao desenvolver um sistema para empresas desse ramo, é necessário implementar validações e permissões de acesso para que não ocorram futuros problemas que possam prejudicar a empresa.

As avaliações com os usuários do sistema mostraram que as funcionalidades implementadas no *software* indicam a capacidade do mesmo em agilizar o gerenciamento de vendas, minimizando o risco de perda de informações. Tais avaliações também mostraram facilidade de uso. O controle das vendas realizadas pode ser feito através dos relatórios. Estes se mostraram satisfatórios por apresentarem dados importantes como o movimento que a empresa teve, produtos e clientes cadastrados, saldo dos caixas e ainda vendedores com seus respectivos lucros percentuais.

Com o ciclo de vida prototipação, foi possível fazer alterações e implementações no sistema de acordo com as necessidades, e com o uso dos diagramas da UML foi possível ter uma maior compreensão e entendimento das funções necessárias para o *software*.

Como o sistema foi implementado, utilizando a linguagem de programação *Java*, seguindo o paradigma de orientação a objetos, o desenvolvimento se tornou mais simples e organizado. Ainda o período de desenvolvimento se tornou mais curto pelo reaproveitamento de código. Futuras manutenções e implementações também serão simplificados pela presença do paradigma orientado a objetos.

Com a implementação do *Hibernate*, o desenvolvimento dos relatórios se tornou mais simples, assim como as classes responsáveis pela manipulação dos dados e as operações de persistência de dados (como inclusão, alteração, exclusão e busca), reduzindo o tempo de trabalho no desenvolvimento do sistema.

Com a utilização da ferramenta *NetBeans* e a implementação dos *plugins* do *iReport*, juntamente da biblioteca Jasper Reports a criação de relatórios foi simplificada e rápida.

O desenvolvimento deste trabalho proporcionou um ganho de conhecimento das tecnologias e padrões utilizados, e ainda colocou-se na prática um passo a passo do início até a finalização do desenvolvimento de um *software*. Também foi possível conhecer os processos de trabalho de empresas que trabalham com vendas.

Considerações Finais

O objetivo do presente estudo consistiu no desenvolvimento de um sistema, com a utilização da linguagem *Java*, orientado a objetos, banco de dados MySQL e

frameworks JasperReports e Hibernate, a fim de controlar o gerenciamento de empresas que trabalham com vendas.

No futuro, este trabalho terá continuidade, e será desenvolvida uma nova versão do *software* em que o mesmo permitirá o gerenciamento de condicionais, nas quais os clientes podem levar os produtos para casa antes de realmente comprá-los; o *software* também conterà vendas com parcelamento, pagamentos com cheque ou cartão e ainda os produtos poderão trabalhar com código de barras.

Referências

- BARBOSA, Simone Diniz Junqueira. SILVA, Bruno Santana. Interação Humano-Computador. Rio de Janeiro: Elsevier, 2010.
- BATISTA, Emerson de Oliveira. Sistemas de Informação: o uso consciente da tecnologia para o gerenciamento. São Paulo: Saraiva, 2004.
- BEZERRA, Eduardo. Princípios de análise e projeto de sistemas com UML. 2a ed. Rio de Janeiro: Elsevier, 2007.
- BOGHI, Cláudio; SHITSUKA, Ricardo. Sistemas de informação: um enfoque dinâmico. 3a ed. São Paulo: Érica, 2007.
- CARVALHO, Sérgio Teixeira. Um design pattern para a configuração de arquiteturas de software. Niterói: UFF/CTC, 2001.
- CORDEIRO, A. G. FREITAS, A. L. P. Priorização de requisitos e avaliação da qualidade de software segundo a percepção dos usuários. - Ci. Inf., Brasília, v.40 n.2, 2011.
- DEITEL, Harvey M.; DEITEL, Paul J. Java, como programar. 6a ed. São Paulo: Pearson Prentice Hall, 2005.
- ELLIOTT, James; O'BRIEN, Tim; FOWLER, Ryan. Dominando hibernate. Rio de Janeiro: Alta Books, 2009.
- FILHO, Elio Lovisi. Estratégias para o desenvolvimento de relatórios utilizando o JasperReports com iReport. Revista eletrônica da Faculdade Metodista Granbery, 2012.
- GUEDES, Thorwald Araujo. UML 2: guia prático. São Paulo: Novatec, 2007.
- GILFILLAN, Ian. Bíblia de MySQL. Espanha - Madrid: Anaya Multimedia, 2003.
- HORSTMANN, Cay. Conceitos de computação com o essencial de Java. 3ª ed. Porto Alegre: Bookman, 2005.
- JUNIOR, Julio Cesar do Carmo; ALMEIDA, Osvaldo Cesar Pinheiro. Sistema de informação geográfica aplicada à gestão pública. Ia. Jornada Científica da Fatec de Botucatu , Botucatu, São Paulo: 2012.

- KRAEMER, Fabiano; VOGT, Jerônimo Jardel. Hibernate, um robusto framework de persistência objeto-relacional. Porto Alegre: 2004.
- LARMAN, Craig. Utilizando UML e padrões. 3a ed. Rio Grande do Sul: Bookman, 2002.
- LAUDON, Kenneth C.; LAUDON, Jane P. Sistemas de informação gerenciais. 7a ed. São Paulo: Prentice Hall Brasil, 2007.
- LESSA, Rafael Orivaldo; JUNIOR, Edson Orivaldo Lessa. Modelos de processos de engenharia de software. Santa Catarina, 2006.
- LIMA, Adilson da Silva. Especificações técnicas de software. São Paulo: Érica Ltda, 2012.
- LIMA, Adilson da Silva. UML 2.3: do requisito à solução. São Paulo: Érica Ltda, 2012.
- Manual de Referência do MySQL 5.7 (2014) "MySQL 5.7 Reference Manual". <<http://dev.mysql.com/doc/refman/5.7/en/index.html>>
- MATULA, Martin. Netbeans metadata repository, 2003.
- NIELSEN, Jakob. Usability engineering. New York: Catherine Plaisant, 1994.
- Oracle (2014) "MySQL Workbench". <<http://www.oracle.com/us/products/mysql/mysql-workbench-066221.html>>
- ORLANDINI, Guilherme; DELFINO, Sérgio Roberto. Introdução ao desenvolvimento de aplicações Java desktop com netbeans 6.7. Águas de São Pedro: Livronovo, 2012.
- PREECE, Jennifer; ROGERS, Yvonne; SHARP, Helen. Design de interação: além da interação homem-computador. Porto Alegre: Bookman, 2005.
- PRESSMAN, Roger S. Engenharia de software: uma abordagem profissional. Porto Alegre: AMGH, 2011.

Apêndices

Apêndice A: Requisitos.

Definição do Contexto

O Sistema terá de efetuar variados cadastros e também efetuar vendas. Ainda deverá permitir a organização dos produtos com sua derivada posição no estoque.

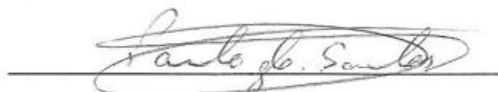
Requisitos:

Efetuar vendas, onde devem ser guardados o vendedor e caixa responsáveis pela venda, os produtos e seu preço, o nome do cliente e o total da venda.

Emissão de relatórios de movimento.

Cadastros com inclusão, alteração, remoção, listagem e pesquisa:

Cadastro de	Dados Necessários
Clientes	Nome, data de nascimento, CPF ou CNPJ, RG, data do cadastro, telefone 1, telefone 2, naturalidade, estado, cidade, CEP, bairro, rua, numero e proximidades.
Fornecedores	Razão Social, nome fantasia, CPF ou CNPJ, RG, conta corrente, representante, inscrição, data do cadastro, fax, telefone 1, telefone 2, data de fundação, observações, estado, cidade, CEP, Bairro, Rua, Número e Proximidades.
Transportadoras	Nome, CPF ou CNPJ, telefone 1, telefone 2, placa, inscrição, data do cadastro, estado, cidade, CEP, Bairro, Rua, Número e Proximidades.
Caixas	Nome, data de nascimento, data de contratação, cart. de Trabalho, CPF, RG, telefone 1, telefone 2, email, saldo, login, senha, estado, cidade, CEP, Bairro, Rua, Número e Proximidades.
Auxiliares	Nome, data de nascimento, data de contratação, cart. de Trabalho, CPF, RG, telefone 1, telefone 2, email, login, senha, estado, cidade, CEP, Bairro, Rua, Número e Proximidades.
Vendedores	Nome, data de nascimento, data de contratação, cart. de Trabalho, CPF, RG, telefone 1, telefone 2, email, percentual de venda, observações, estado, cidade, CEP, Bairro, Rua, Número e Proximidades.
Produtos	Nome, marca, departamento, setor, seção, peso, data da compra, quantidade, quantidade mínima, fornecedor, transportadora, preço de custo, preço de venda.
Seções	Nome, departamento, setor.
Setores	Nome, departamento.
Departamentos	Nome.



Assinatura do Responsável

Data: 05/02/14

Requisitos

Apêndice B: Protótipo Rápido.

Clientes

Nome: Data Nascimento:

Tipo: Físico Jurídico CPF: CNPJ: RG: Data Cadastro:

Estado: Cidade: CEP: Bairro:

Rua: Numero: Proximidades:

Clientes cadastrados

Protótipo rápido

Apêndice C: Ficha de Avaliação do protótipo rápido.

Clientes

Nome: Data Nascimento:

Tipo: Físico Jurídico CPF: CNPJ: RG: Data Cadastro:

Estado: Cidade: CEP: Bairro:

Rua: Numero: Proximidades:

Clientes cadastrados

A posição dos campos, tabela e botões está agradável?	<input checked="" type="checkbox"/> SIM () NÃO
A interface está informativa, de simples compreensão?	<input checked="" type="checkbox"/> SIM () NÃO
Aceita este padrão de tela para as outras telas de cadastros do sistema?	<input checked="" type="checkbox"/> SIM () NÃO



Assinatura do responsável

Data: 05/02/14

Ficha de avaliação do protótipo rápido

Apêndice D: Diagrama de casos de uso.

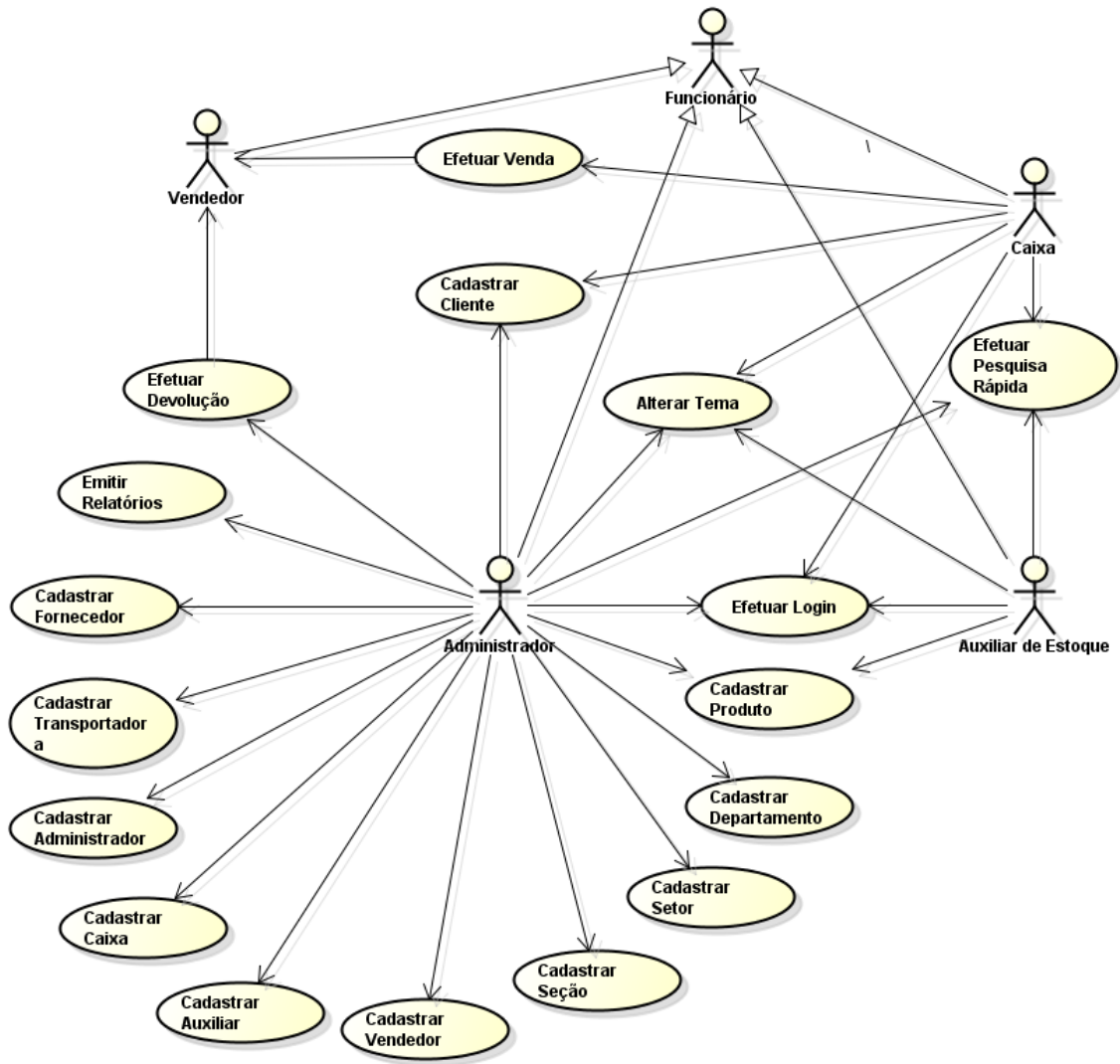


Diagrama de casos de uso

Apêndice E: Diagrama de classes.

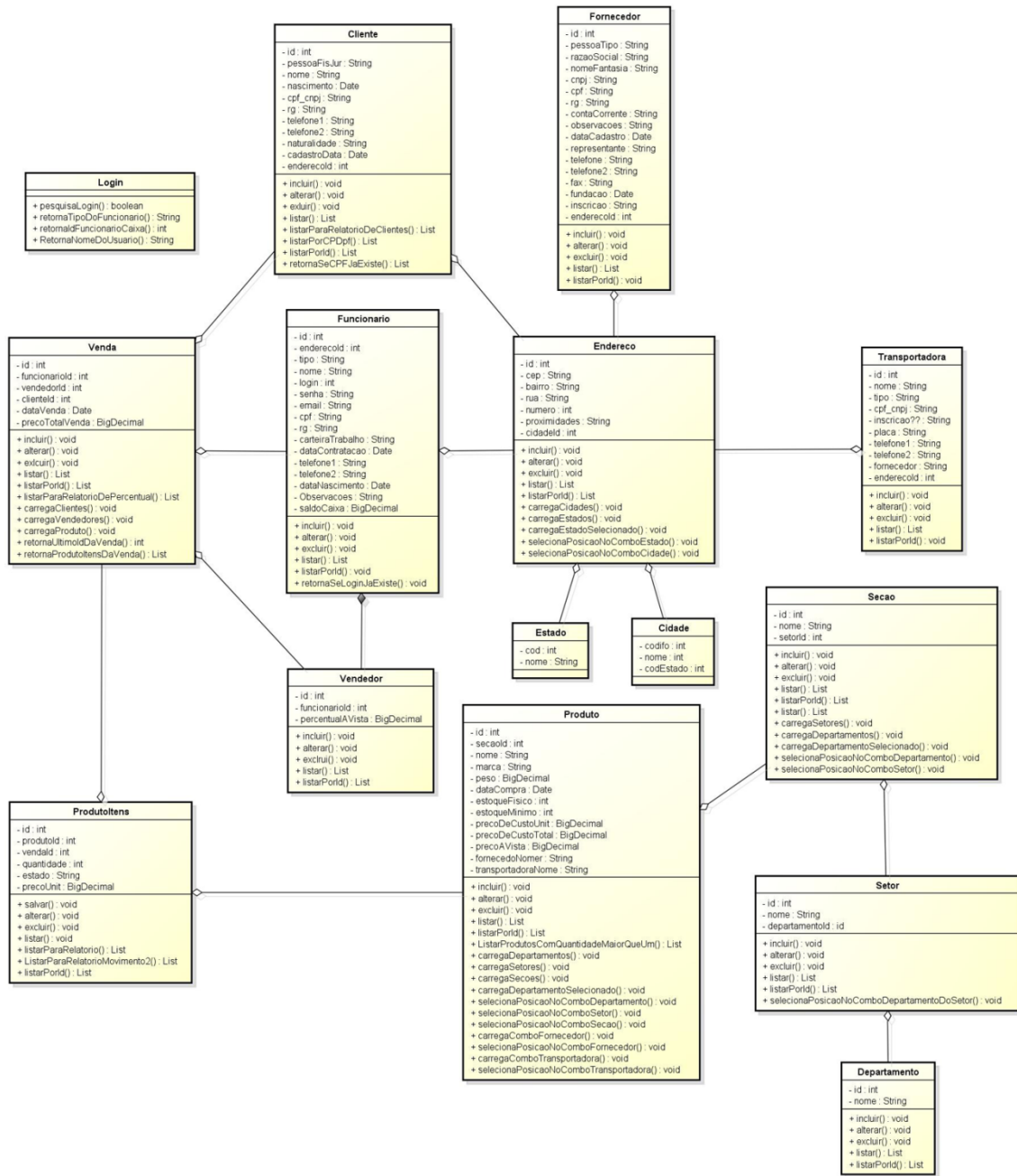


Diagrama de classes

Apêndice F. Diagrama de Sequência do caso de uso "Cadastrar Cliente"

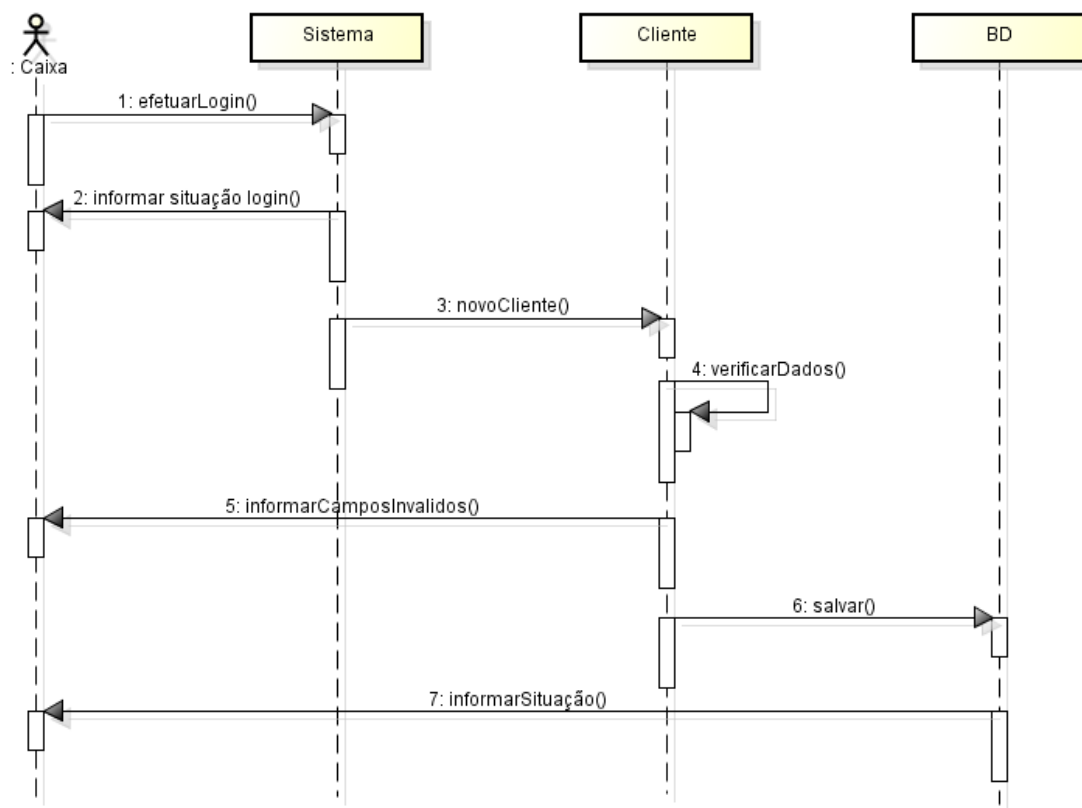


Diagrama de Sequência do caso de uso "Cadastrar cliente"

Apêndice G. Diagrama de Sequência do caso de uso "Cadastrar Produto"

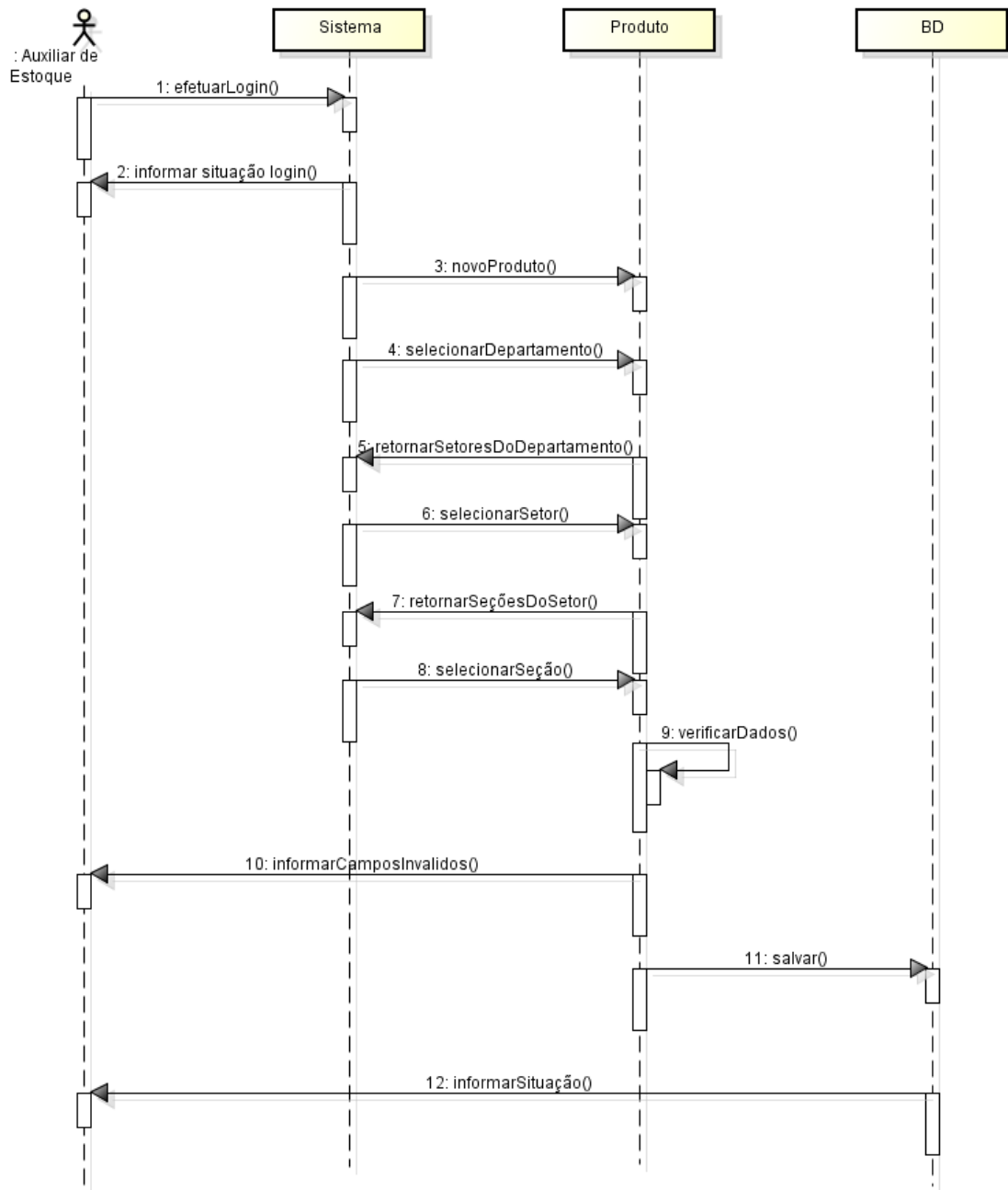


Diagrama de Sequência do caso de uso "Cadastrar produto"

Apêndice H. Diagrama de Sequência do caso de uso "Efetuar Venda"

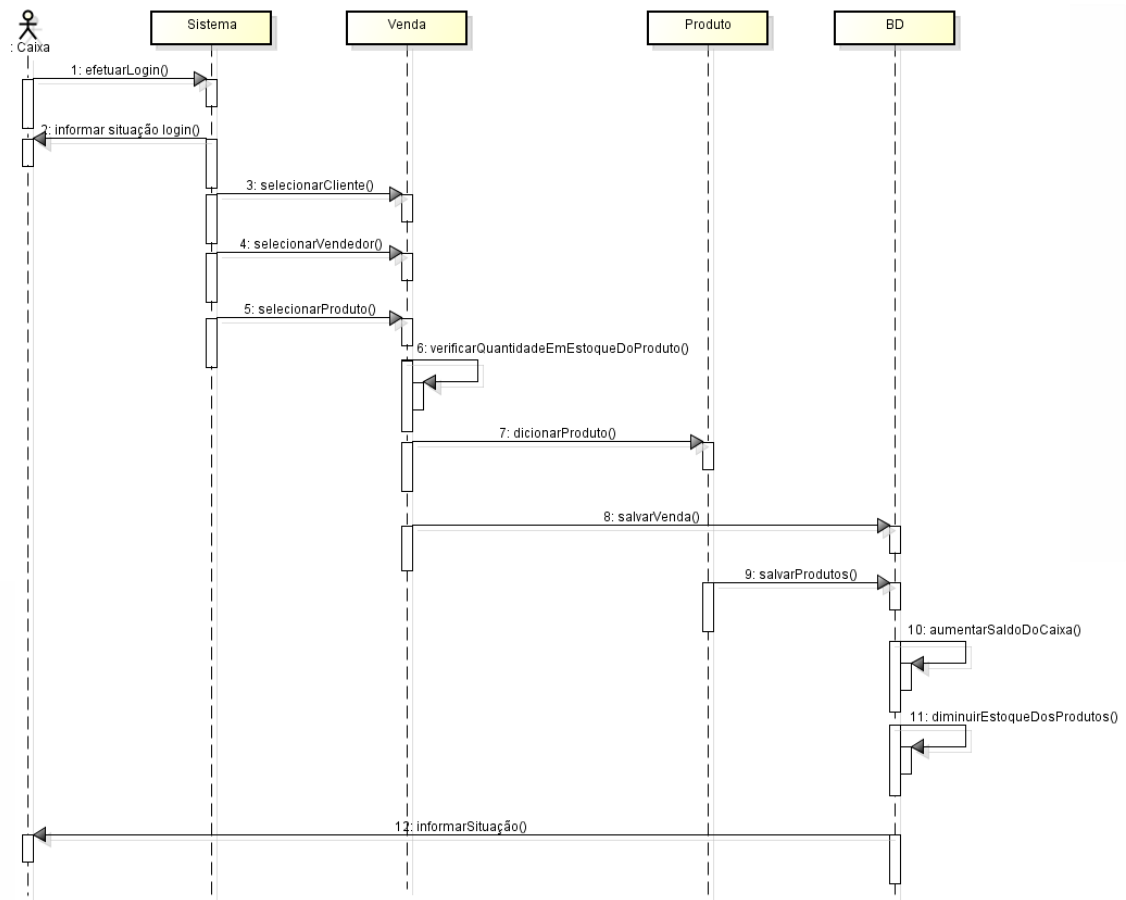
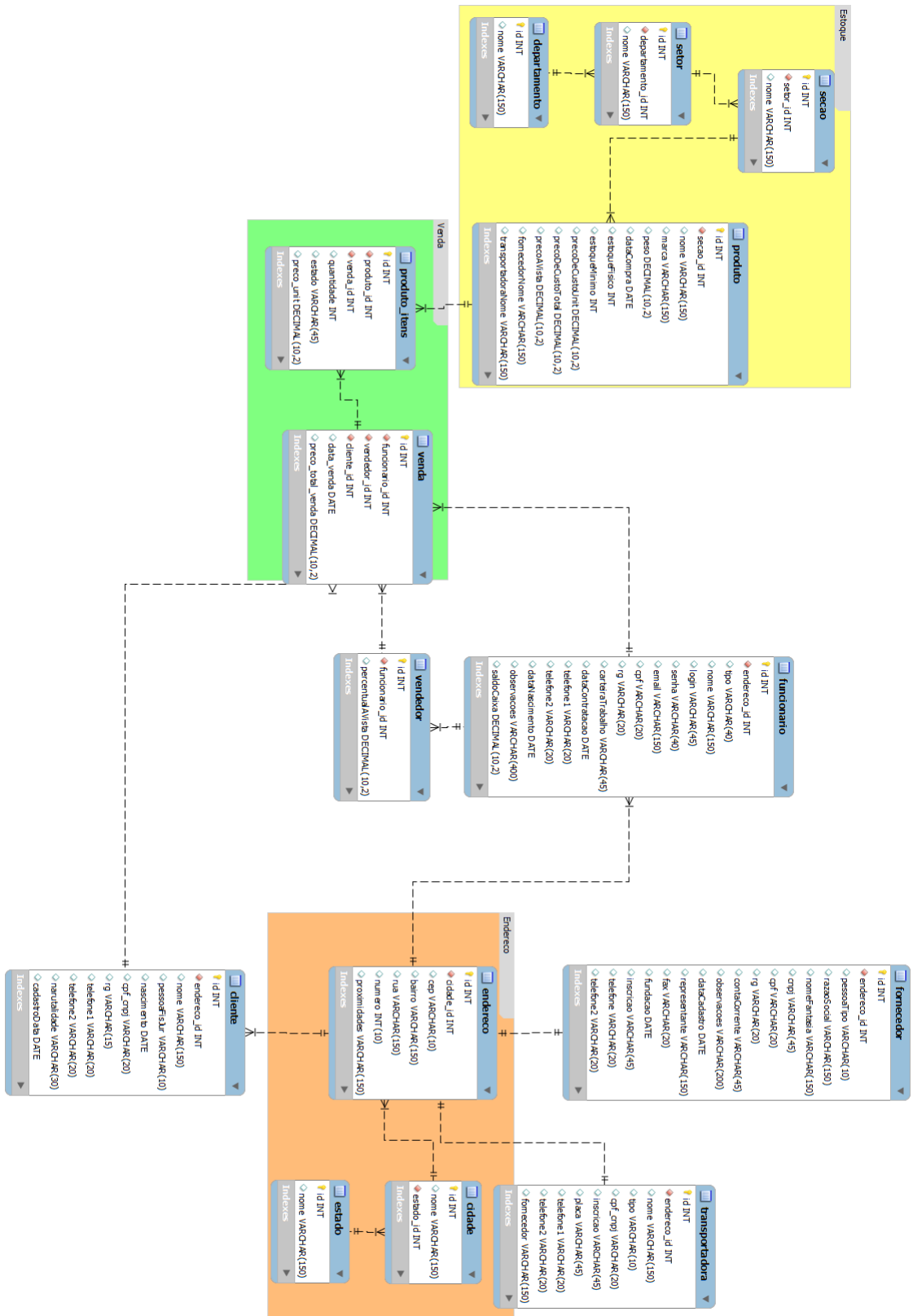


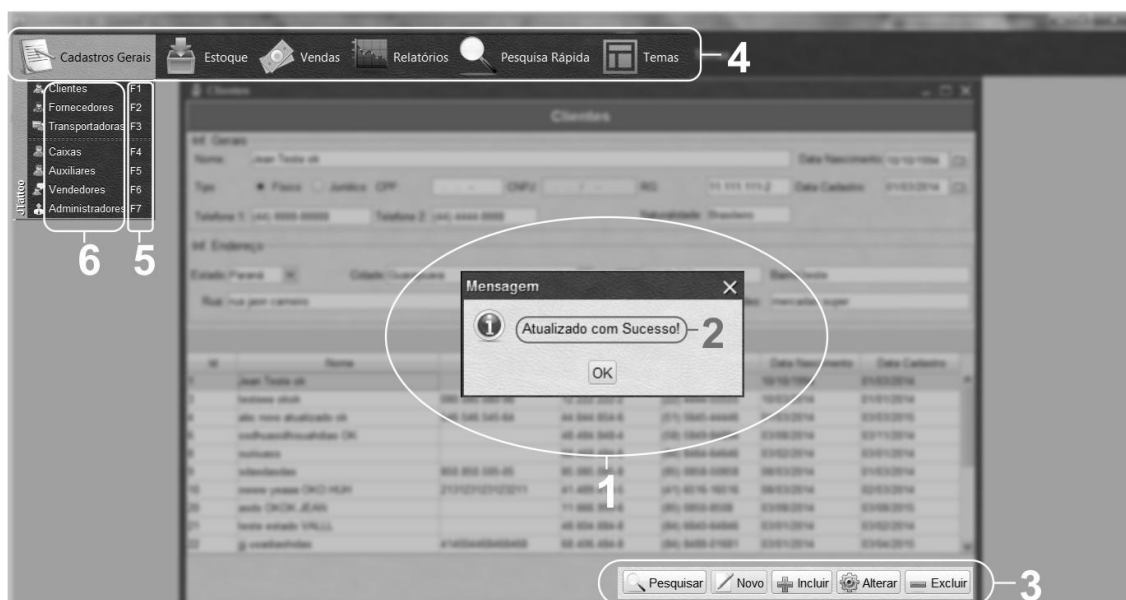
Diagrama de Sequência do caso de uso "Efetuar venda"

Apêndice I: Modelagem lógica do sistema



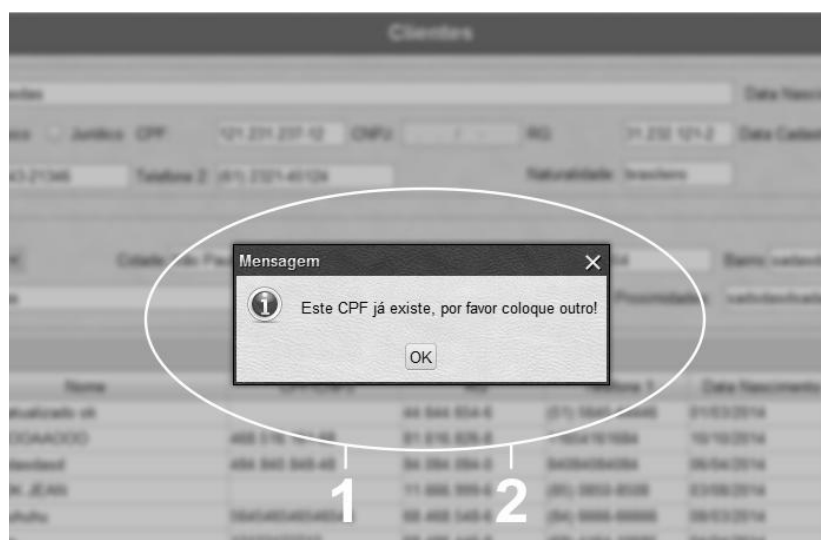
Modelagem lógica

Apêndice J. Tela com as heurísticas de usabilidade: 1. Visibilidade de Status do Sistema; 2. Relacionamento entre a interface do sistema e o mundo real; 3. Consistência; 4. Reconhecimento ao invés de lembrança; 5. Flexibilidade e eficiência de uso; 6. Estética e design minimalista.



Tela 1 do Sistema com implementação das Heurísticas de Usabilidade

Apêndice K. Tela 2 com as heurísticas de usabilidade: 1. Prevenção de erros; 2. Ajudar os usuários a reconhecer, diagnosticar e sanar erros.



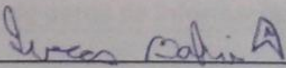
Tela 2 do Sistema com implementação das Heurísticas de Usabilidade

Apêndice L: Ficha 1 de Avaliação do protótipo com os erros encontrados

Você encontrou algum erro no sistema?	<input checked="" type="checkbox"/> SIM () NÃO
Se sim, quais erros?	
<p>Se eu logar com o usuário saizo ou auxilizar, ainda assim posso abrir e utilizar as telas que somente são permitidas ao administrador.</p>	
O tempo de resposta do sistema é satisfatório?	<input checked="" type="checkbox"/> SIM () NÃO
O sistema é de fácil manuseio?	<input checked="" type="checkbox"/> SIM () NÃO
A interface do sistema é agradável?	<input checked="" type="checkbox"/> SIM () NÃO
Você acha que as funcionalidades implementadas no software possuem capacidade agilizar o gerenciamento de vendas minimizando o risco de perda de informações?	<input checked="" type="checkbox"/> SIM () NÃO
<p><u>Ederino Soares Machado</u></p> <p>Assinatura do responsável</p> <p>Data: <u>20/04/14</u></p>	

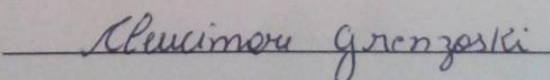
Ficha 1 de avaliação do protótipo com os erros encontrados

Apêndice M: Ficha 2 de Avaliação do protótipo com os erros encontrados

Você encontrou algum erro no sistema?	<input checked="" type="checkbox"/> SIM () NÃO
Se sim, quais erros?	
<p>As credenciais em produtos listos até os departamentos que sem possuem setores e regiões credenciados, ai da um erro ao tentar entrar.</p>	
O tempo de resposta do sistema é satisfatório?	<input checked="" type="checkbox"/> SIM () NÃO
O sistema é de fácil manuseio?	<input checked="" type="checkbox"/> SIM () NÃO
A interface do sistema é agradável?	<input checked="" type="checkbox"/> SIM () NÃO
Você acha que as funcionalidades implementadas no software possuem capacidade agilizar o gerenciamento de vendas minimizando o risco de perda de informações?	<input checked="" type="checkbox"/> SIM () NÃO
 _____ Assinatura do responsável	
Data: <u>20/04/14</u>	

Ficha 2 de avaliação do protótipo com os erros encontrados

Apêndice N: Ficha 3 de Avaliação do protótipo com os erros encontrados

Você encontrou algum erro no sistema?	<input checked="" type="checkbox"/> SIM () NÃO
Se sim, quais erros?	
* Ao atualizar um produto, não atualiza as informações referentes a seqçõ, setor e departamentos que ele se encontra.	
O tempo de resposta do sistema é satisfatório?	<input checked="" type="checkbox"/> SIM () NÃO
O sistema é de fácil manuseio?	<input checked="" type="checkbox"/> SIM () NÃO
A interface do sistema é agradável?	<input checked="" type="checkbox"/> SIM () NÃO
Você acha que as funcionalidades implementadas no software possuem capacidade agilizar o gerenciamento de vendas minimizando o risco de perda de informações?	<input checked="" type="checkbox"/> SIM () NÃO
 Assinatura do responsável Data: <u>22/04/14</u>	

Ficha 3 de avaliação do protótipo com os erros encontrados

Apêndice O: Ficha 4 de Avaliação do protótipo com os erros encontrados

Você encontrou algum erro no sistema?	() SIM (<input checked="" type="checkbox"/>) NÃO
Se sim, quais erros?	

O tempo de resposta do sistema é satisfatório?	(<input checked="" type="checkbox"/>) SIM () NÃO
O sistema é de fácil manuseio?	(<input checked="" type="checkbox"/>) SIM () NÃO
A interface do sistema é agradável?	(<input checked="" type="checkbox"/>) SIM () NÃO
Você acha que as funcionalidades implementadas no software possuem capacidade agilizar o gerenciamento de vendas minimizando o risco de perda de informações?	(<input checked="" type="checkbox"/>) SIM () NÃO



Assinatura do responsável

Data: 22/04/14

Ficha 4 de avaliação do protótipo com os erros encontrados

Apêndice P. Ficha de avaliação do protótipo com as alterações necessárias

Deseja alguma alteração no sistema?	<input checked="" type="checkbox"/> SIM <input type="checkbox"/> NÃO
Se sim, quais são estas alterações necessárias para a versão final do sistema?	
<p>Telas de pesquisa rápida onde se possa pesquisar os principais dados referentes a Clientes, Transportadoras e fornecedores, e ainda possa também verificar a seção, setor e departamento do produto desejado. OBS: A Tela de pesquisa rápida do cliente deve permitir busca por CPF.</p>	



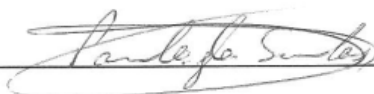
Assinatura do responsável

Data: 22/04/14

Ficha de avaliação do protótipo com as alterações necessárias

Apêndice Q. Ficha de avaliação final

1) Você considera o uso do sistema:	<input type="checkbox"/> Difícil <input checked="" type="checkbox"/> Fácil
2) Durante a execução das tarefas ocorreu algum tipo de falha por parte do sistema?	<input type="checkbox"/> SIM <input checked="" type="checkbox"/> NÃO
3) O sistema atende aos principais processos de venda necessários para o funcionamento de empresas do ramo?	<input checked="" type="checkbox"/> SIM <input type="checkbox"/> NÃO
4) O sistema atende aos requisitos levantados no início do projeto?	<input checked="" type="checkbox"/> SIM <input type="checkbox"/> NÃO
5) Em relação ao tempo de resposta do sistema você considera:	<input checked="" type="checkbox"/> Rápido <input type="checkbox"/> Lento
6) Em relação a facilidade de uso do sistema você considera:	<input checked="" type="checkbox"/> Bom <input type="checkbox"/> Ruim
7) Ao trabalhar em uma tela que ainda não tinha utilizado, a dificuldade é:	<input type="checkbox"/> Grande <input checked="" type="checkbox"/> Pequena
8) Em relação as vendas, o sistema permite que sejam efetuadas de um modo:	<input checked="" type="checkbox"/> Rápido <input type="checkbox"/> Lento
9) Em relação ao estoque no sistema, os produtos podem ser bem organizados e controlados?	<input checked="" type="checkbox"/> SIM <input type="checkbox"/> NÃO
10) No sistema foram corrigidas as alterações desejadas?	<input checked="" type="checkbox"/> SIM <input type="checkbox"/> NÃO
11) Os relatórios atendem aos objetivos?	<input checked="" type="checkbox"/> SIM <input type="checkbox"/> NÃO
12) Com a implementação deste sistema seria possível agilizar os processos e minimizar os riscos de perda de informações?	<input checked="" type="checkbox"/> SIM <input type="checkbox"/> NÃO



Assinatura do responsável

Data: 29/05/14

Ficha de avaliação final