Utilizando Tecnologias Gratuitas para o Desenvolvimento de um Sistema para Gerenciamento de Açougues

Willian Ricardo Fialka Agner[1], Rodrigo Souza D'avila[1]

[1],[1]Tecnologia em Analise e Desenvolvimento de Sistemas
Faculdade Guairacá Guarapuava 2012

will.agner@hotmail.com, rsdavilla@gmail.com.br

Abstract

With the increasing of information technology use in operational and management processes of enterprises, opened up many niche markets. Many companies are working with software development and software products that aid in carrying out its enterprise customers, but there are branches unexplored. It was found that there are few softwares that perform management of the butchers, based on this we began the development of "SisGera" (Butcher Management System). The work began with the analysis of business trade of meat, identified existing processes along with the users through interviews, which were later identified features to be implemented in software to anchieve the process.

Resumo

Com o crescimento da utilização da tecnologia da informação nos processos operacionais e gerenciais dos empreendimentos, abriram-se muitos nichos de mercado. Muitas empresas trabalham com desenvolvimento e produtos de *software* que realizem este auxílio no empreendimento de seus clientes, porém existem ramos ainda não explorados. Foi verificado que existem poucos *softwares* que realizem o gerenciamento de açougues, e com base nisto iniciou-se o desenvolvimento do "SisGera" (Sistema de Gerenciamento de Açougue). O trabalho se iniciou com a análise de negócio do comercio de carnes, identificados os processos existentes juntamente com os usuários através de entrevistas, posteriormente foram identificados quais funcionalidades a serem implementadas no *software* para atender o processo.

1.0 Introdução

Foi definido o ramo de comercio de carnes, que hoje possuí poucos *softwares* de gerenciamento no mercado. Para o levantamento dos requisitos do *software* a ser desenvolvido foram realizadas entrevistas com conhecedores de gerenciamento da venda de carnes. Este é o primeiro passo para a definição de como funciona o processo do cliente, e do que é necessário conter no *software*. A partir destas entrevistas foram gerados os casos de uso com base no conhecimento dos mesmos e das necessidades apresentadas por eles, e também foram-se definidos os requisitos do que a ferramenta necessitaria possuir (Marcio Freitas analyst/Developer .NET, 2012).

O desenvolvimento foi realizado na linguagem Java com a ferramenta NetBeans 7.2 integrado com banco de dados MySQL, com o auxilio do SGBD (Sistema de Gerenciamento de Banco de Dados) *"MySql WorkBench"*. Os diagramas do *software* foram criados no programa Astah, e para o auxilio na persistência de dados se utilizou o *framework* Hibernate.

Foi identificado na análise do negócio, que o processo de verificação de "inventário de carnes" neste tipo de negócio é trabalhoso, pois se faz necessária à verificação manual de cada entrada de produto que foi realizada, para se realizar a subtração do que foi vendido mais o que consta em estoque, para se efetuar o "balanço". O controle dos produtos que entraram é realizado com base nas notas fiscais dos fornecedores que são arquivadas, o que transforma o processo trabalhoso.

Com a utilização do *software* o usuário terá as informações referentes às entradas, que são necessárias para realizar este trabalho que operacionalmente é um dos pontos críticos. Com a utilização dos relatórios ele poderá realizar buscas de quantidades de entradas de produto, possuindo também a informação de quem foi o fornecedor.

2.0 Comércio de Carnes

O País se tornou o terceiro produtor mundial e líder em exportação de carnes. Atualmente, a carne nacional chega a 142 países. Outras aves, como peru e avestruz, também têm se destacado nos últimos anos, contribuindo para diversificar a pauta de exportação do agronegócio brasileiro. A bovinocultura é um dos principais destaques do

agronegócio brasileiro no cenário mundial. O Brasil é dono do segundo maior rebanho efetivo do mundo, com cerca de 200 milhões de cabeças. Além disso, desde 2004, assumiu a liderança nas exportações, com um quinto da carne comercializada internacionalmente e vendas em mais de 180 países. (Ministério da Agricultura, 2012).

O Brasil é um grande produtor mundial de proteína animal e tem no mercado interno o principal destino de sua produção. Considerando a produção brasileira de carnes (bovina, suína e de aves) em 2010, estimada em 24,5 milhões de toneladas, temos que 75% dessa produção é consumida internamente no país. (Ministério da Agricultura, 2011).

Neste ano, o consumo per capita de carnes aumentou em relação ao ano anterior chegando a 37,4 kg para carne bovina; 43,9 kg de carne de aves e 14,1 kg de carne suína, refletindo o bom desempenho da economia brasileira. Também as carnes ovina e caprina, assim como a produção de leite e seus derivados, são consumidas majoritariamente no mercado interno brasileiro. (Ministério da Agricultura, 2012).

Estes fatores apresentados justificam o desenvolvimento de *software* para gerenciamento de açougues, uma vez que o mercado interno possui carência de *softwares* que administrem estas informações.

3.0 Ferramentas Utilizadas no Desenvolvimento

3.1 Java

O Java foi criado a partir de um projeto da *Sun Microsystem* em 1991, sendo lançado oficialmente em 1995, e teve um grande crescimento desde então, é um das mais utilizadas linguagens para desenvolvimento de sistemas, foi uma das linguagens em que a orientação a objeto ganhou espaço e tornou-se conhecida, porém também tem suporte a desenvolvimento estruturado (Deitel, 2009, p6).

O Java utiliza o paradigma de programação orientado a objeto (O.O), que emprega a reutilização de código, e uma melhora na manutenção do sistema, o crescimento das empresas e mudanças da legislação são exemplos de motivos que geram a necessidade de alterações nos *softwares*. A utilização do modelo MVC com orientação a objeto vem de encontro com estes fatos, pois facilita as alterações nas regras de negócios do *software*, pois neste formato de trabalho tudo fica devidamente

separado. O trabalho com a orientação a objeto iniciou-se na década de 60, e desde lá vem amadurecendo cada vez mais. A participação com afinco da orientação a objeto na computação deu-se na década de 90, foi onde linguagens já existentes como C++ e PHP passaram a ser orientadas a objeto (BARREIRO; SOBRAL, 2000, p8).

Além de ser uma linguagem de desenvolvimento aberto, o Java possuí um forte recurso, que é a JVM(*Java Virtual Machine*). A *Java Virtual Machine* é uma máquina virtual utilizada para interpretar os códigos na linguagem Java, assim o Java pode ser utilizado em inúmeros SO's. Desta maneira o Java criou capacidade de portabilidade, pois indiferente do sistema operacional que você está utilizando, não será ele que irá interpretar o código fonte, e sim a máquina virtual do Java, isso facilita a implementação de um software desenvolvido em Java nos hardwares que o seu cliente possuir, pois tudo estará vinculada a JVM, não havendo problemas de compatibilidade (BARREIRO; SOBRAL, 2000, p15).

3.2 Hibernate

Uma das principais frentes de trabalho do desenvolvimento de software é a persistência dos dados, ou seja, todo o trabalho que o usuário executar no software, deve ser registrado com a possibilidade de ser resgatado quando ele iniciar a aplicação novamente, mantendo o histórico dos dados movimentados. É a análise desses dados, agrupados da forma correta que se tornam o diferencial de competitividade dos empreendimentos, informações não vistas sem o agrupamento de informações são encontradas ao se fazer a análise de dados.

Quando se iniciou a utilização dos bancos de dados relacionais, foi criada uma linguagem para a comunicação com esta base de dados, a SQL (*Structured English Query Language)* que foi baseada nos conceitos de Álgebra Relacional. Esta linguagem aperfeiçoou e padronizou o processo da utilização dos bancos de dados relacionais (MySQL Team, 2010 p3).

Porém sua utilização acarreta um gasto de tempo de desenvolvimento superior se comparado com a utilização de *Frameworks*, por exemplo, para o desenvolvimento de cinco cadastros em um *software*, você necessitará escrever as cinco vezes as mesmas linhas de código SQL e de conexão com o banco, alterando-se apenas os campos do banco de dados que estão sendo utilizados e as informações que são passadas.

Utilizando a padronização do SQL como beneficio, identificou-se a possibilidade de criar um padrão de desenvolvimento para aplicar a reutilização do código. Nesta nova forma de se realizar a programação, o desenvolvedor deve preocupar-se com a regra de negócio, e com a interface de apresentação para o usuário final, pois a persistência de dados no banco é realizada pelos *frameworks* que realizaram esta padronização da utilização do SQL (Gavin King, at all , p.10).

No desenvolvimento do sistema foi aplicado a utilização do Hibernate, que trata-se de um *framework* de código livre desenvolvido pela empresa Jboss Inc. Ele trabalha com a estrutura MOR (Mapeamento Objeto Relacional), que realiza o mapeamento das classes Java, transformando-as em objetos compatíveis com o banco, ou seja, uma tabela no banco, terá uma classe Java correspondente, porém não necessariamente uma classe Java terá uma tabela no banco de dados (Hibernate Team, 2010).

O Hibernate traz a proposta de realizar a persistência em banco de dados, para isto o desenvolvedor necessita realizar o mapeamento das suas classes, que se trata de criar as referencias do banco de dados nas classes java, e garantir que seu trabalho está sendo realizado com praticas de programação Orientadas a Objeto. A partir deste mapeamento o Hibernate saberá qual tabela do banco representa a classe java que está sendo trabalhada na aplicação, e onde os dados gerados devem ser persistidos. Assim sendo o desenvolvedor passa a maior parte do seu tempo trabalhando na regra de negócio e na interface de apresentação ao usuário, que são outras principais frentes de trabalho no desenvolvimento de *software*, junto com a persistência dos dados.

> Segundo King
>
> "A meta de Hibernate é aliviar o desenvolvedor em 95% de dados comuns de persistência relacionados as tarefas de programação. O Hibernate talvez não seja a melhor solução para as aplicações centradas em dados, das quais apenas usam procedimentos armazenados para a implementação das lógicas comerciais no banco de dados. Isto é mais utilizado orientando o objeto aos modelos de domínio e lógicas comerciais na camada intermediária baseada em Java." (Gavin King, , p.11).

3.3 MySQL

O SGBD MySql é um sistema de banco de dados, e foi utilizado para persistir os dados do *software*. Foi criado na Suécia por dois suecos e um finlandês: David Axmark, Allan Larsson e Michael "Monty" Widenius, que têm trabalhado juntos desde a década de 1980. (MySQL Team, 2010)

O MySQL, que utiliza a linguagem SQL como interface, é atualmente um dos bancos de dados mais populares, com mais de 10 milhões de instalações pelo mundo. (MySQL Team, 2010)

Entre os usuários do banco de dados MySQL estão: NASA *(National Aeronautics and Space Administration*; Administração Nacional da Aeronáutica e do Espaço), Friendster, Banco Bradesco, Dataprev, HP (Hewlett-Packard), Nokia, Sony, Lufthansa, U.S Army, US. Federal Reserve Bank, Associated Press, Alcatel, Slashdot, Cisco Systems, Google CanaVialis S.A. e outros. (MySQL Team)

Hoje seu desenvolvimento e manutenção empregam aproximadamente 400 profissionais no mundo inteiro, e mais de mil contribuem testando o software, integrando-o a outros produtos, e escrevendo a respeito dele. (MySQL Team)

Ele também dispõe de um servidor de banco de dados que é muito rápido e confiável, de utilização simples, e este foi utilizado juntamente com o Hibernate para as persistências de dados no banco de dados.

3.3.1 MySQL WorkBench

O MySQL WorkBench é uma ferramenta que possui uma interface gráfica para a visualização dos servidor e banco de dados MySQL, ele facilita a modelagem do banco, vinculações de tabelas com chaves estrangeiras bem como as configurações de conexão do servidor. Nele também é possível executar comandos SQL para verificar como a persistência de dados da aplicação desenvolvida, está se refletindo no banco de dados. (MySQL WorkBench Team)

3.4 JasperReport

Quando deu-se iniciou a utilização de relatórios de *softwares*, não existia a opção de visualização dos dados no meio digital, os relatórios eram gerados apenas em formato físico, impresso em papel. Porém com o crescimento da comunicação digital, passou-se a ser impresso apenas as informações necessárias, pois é seguido o sentido de que a informação deve ficar salva em meio digital (Atie, Silva, p2, 2010).

O JasperReport é uma ferramenta que independe da ferramenta que foi desenvolvida para armazenamento de informações do banco de dados, ela pode ser utilizada tanto integrada com o produto quanto separadamente para a geração dos relatórios (Ireports Team 2010).

Esta ferramenta foi integrada ao *software* SisGera, foi desenvolvida uma tela para cada chamada de relatório do sistema, os relatórios desenvolvidos no ireports, são abertos diretamente do *software*, sem que haja necessidade de utilização de mais de um *software*.

4.0 Desenvolvimento

O processo de desenvolvimento do SisGera iniciou-se com o levantamento de requisito e definições dos processos, que foram realizados com dois usuários com experiência no ramo operacional e gerencial do negócio. Após este passo foi gerado o caso de uso, o diagrama de classe e por fim o diagrama de sequência.

Foi analisado o diagrama de classes, e com base nesta análise foi gerada a modelagem do banco de dados com a utilização do *WorchBench*, onde foram criadas e vinculadas as tabelas necessárias para atender o diagrama de classes. Com a modelagem pronta, foi iniciado o trabalho de configuração do Hibernate, partindo das configurações básicas de parâmetros de conexão com o banco de dados para que o Hibernate estabeleça comunicação, e posteriormente com a criação das *Annotations*.

4.1 MVC

O projeto seguiu o conceito de MVC, trabalhando com três camadas de desenvolvimento, inicialmente foram trabalhados os pacotes “*Model*”, “*Dao*”, “*View*”, e posteriormente sendo incrementados conforme se surgia à necessidade.

Segundo Melo 2007, MVC é uma estratégia de separação de camadas de software que visa desacoplar a interface de seu tratamento e de seu estado. Algumas pessoas consideram o MVC não como um padrão, mas sim como uma estratégia de projeto. O padrão MVC implementa um *use-case* interativo divido em três partes:

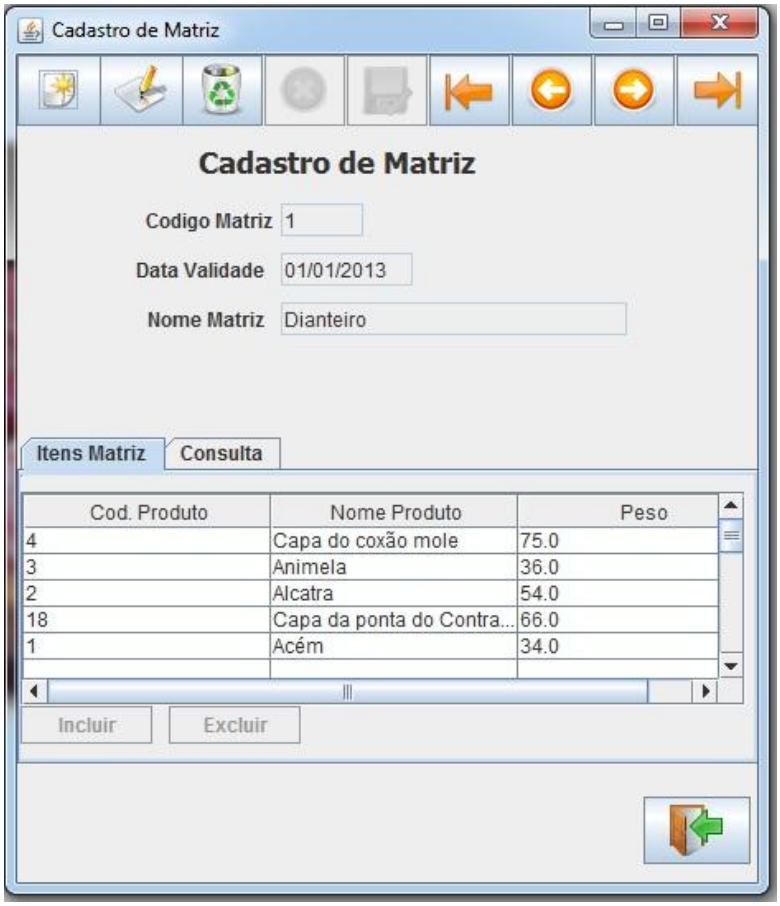
- ***Model* (modelo):** Mantém o estado atual da *view*. É responsável por alterar o estado e fornecer a *View* os valores atuais.
- ***View* (visualização):** Captura ações do usuário e comunica o *Controller* para que sejam executadas. Também atualiza as informações exibidas sempre que o controller avisa que devem ser atualizadas.
- ***Controller* (controlador):** Recebe solicitações da *View*, invoca as regras de negócio necessárias, comanda a alteração do estado do Model e atualização das informações exibidas na *view*. (Melo, p78, 2007).

As *Annotations* servem para que o Hibernate identifique a qual tabela no banco de dados a classe se refere, devem ser mapeados os atributos da classe, para que se identifique qual coluna da tabela o mesmo se refere. Com esta informação o Hibernate conseguirá gravar as informações corretamente. Foram criadas as annotations para as tabelas: cidades, estado, fornecedor, itemmatriz, itensnf, matriz, notafiscal, produto, usuário que se encontram no pacote model.

Depois de criados os mapeamentos com as annotations no pacote Model, iniciou-se o desenvolvimento da base do sistema, que realiza a chamada para telas. Foram trabalhados os menus e seus comandos iniciais. Depois de concluída a tela principal do sistema, foi elaborada a primeira tela de cadastro, que realizava o cadastro de cidades. Para este desenvolvimento, foi criado um padrão de funcionalidades para a tela, este padrão é seguido em todas as telas, ou seja, todas as telas possuem o mesmo *layout* de botões e funcionalidades, o que os diferencia são seus dados. A tela de cadastro de cidades ficou como especificada abaixo:

Padrão de Telas do Sistema Cadastro Simples

Outro modelo de tela que existe no sistema é a tela de cadastro complexo, que possui a presença de registros pai e filho, o modelo da tela em questão ficou como a abaixo:



Padrão de Telas do Sistema Cadastro Complexo

Por padrão as telas vêm com modo apenas de visualização para o usuário, com todos os campos desabilitados, e com base nas ações do usuário a tela altera seus status. As definições das funcionalidades padrões são as que seguem abaixo:

1. Botão Novo

Disponibiliza a tela em modo de inclusão de registro.

Habilita os campos para inclusão de informações.

Desabilita os botões de navegação.

Habilita o botão Cancelar (4) e o botão Salvar Registro (5).

2. Botão Editar

Disponibiliza a tela em modo de edição de registro.

Mantém as informações que estavam sendo visualizadas na tela.

Habilita os campos para inclusão de informações.

Desabilita os botões de navegação.

Habilita o botão Cancelar (4) e o botão Salvar Registro (5).

3. Botão Deletar Registro

Realiza a deleção do registro que está posicionado na tela.

4. Botão Cancelar

Cancela a ação que está sendo realizada no registro.

Cancela a inclusão de um novo registro (1).

Cancela a edição de um registro (2).

Desabilita os campos para inclusão de informações.

Habilita os botões de navegação.

5. Botão Salvar

Salva a inclusão de registros ou a edição dos registros realizados na tela.

6. Botão Primeiro Registro

Realiza a busca no banco de dados do primeiro registro e o disponibiliza na tela.

7. Botão Registro Anterior

Realiza a busca no banco de dados, buscando o registro que corresponda ao que consta em tela menos um.

8. Botão Próximo Registro

Realiza a busca no banco de dados, buscando o registro que corresponda ao que consta em tela mais um.

9. Botão Último Registro

Realiza a busca no banco de dados do último registro e o disponibiliza na tela.

10. Botão Buscar

Realiza a busca no banco de dados para o preenchimento das informações na tabela.

11. Botão Sair

Finaliza a utilização da tela.

12. Campo para Pesquisa

Campo utilizado para inclusão das informações a serem filtradas nas buscas com criteria.

13. Tabela de Registros

Tabela utilizada para apresentação dos registros existentes no banco de dados.
Ao clicar em um registro na tabela ele é posicionado na tela para ser utilizado.

Para realizar o controle de entradas de produtos no açougue que se foi proposto, foi desenvolvido os seguintes itens dentro do projeto:

Cadastro de Usuário

Conterá as informações do usuário, como nome, CPF, RG, telefone, data de nascimento e mais seu login e senha para acesso ao sistema.

Cadastro de Estados

Será utilizado para identificar qual estado o fornecedor se encontra, pois muitas vezes o produto vem de fora do estado em que o cliente se encontra.

Cadastro de Cidades

Especifica a cidade do fornecedor.

Cadastro de Produtos

Contêm os produtos que irão ser adquiridos dos fornecedores, estes produtos serão vinculados a matrizes, onde conterá a quantidade deste produto dentro da matriz.

Cadastro de Fornecedor

Possuirá as informações dos fornecedores de produtos, como nome, telefone, cnpj, cidade.

Matrizes

As matrizes são uniões de produtos, ou seja, uma matriz pode ter vários produtos, e quantidades variadas destes produtos. Para realizar a entrada de

carnes no açougue, não é realizada a pesagem do produto para ser estocado, o boi, por exemplo, é dividido em quatro partes, estas partes são pesadas, e posteriormente é realizado a média de quanto de cada produto havia em cada quarto do boi. Após este processo as entradas são feitas tendo como base a matriz e não quantidades exatas de cada produto.

Nota Fiscal

Neste controle é realizado o cadastro das notas fiscais externas, que são apresentadas pelos fornecedores. Nela é informada a matriz de produtos que entrou de qual fornecedor e a data de entrega da nota. Este processo realiza a vinculação de quase todas as tabelas do sistema.

Para auxiliar na utilização do *software* também foi desenvolvida a tela de manutenção de senha de usuários, por ser um item bastante utilizado. Foram desenvolvidos também os relatórios para verificação das entradas e cadastros realizados no sistema.

Posterior a todos os desenvolvimentos foram criados os relatórios com a utilização do ireports, apresentando relatórios simples e compostos com sub-relatórios para as tabelas pai e filho.

Durante o processo de desenvolvimento, foram gastas 110 horas, nisto envolvendo toda a parte de modelagem do banco de dados, mapeamento das classes no Hibernate, criação da tela principal de controle do *software*, criação dos cadastros simples e dos cadastros complexos, validações, verificações, controle de edição, inclusão alteração e buscas e também o desenvolvimento dos relatórios. Com os diagramas UML foram gastas 19 horas, envolvendo levantamento, criação e revisão de caso de uso, diagrama de classe e diagrama de sequência.

Foi seguido à metodologia de desenvolvimento cascata, que trabalha de forma linear, partindo do levantamento de requisitos, indo para o planejamento de execução do projeto para a produção do sistema, posteriormente passando para a modelagem do sistema, onde são definidas as soluções técnicas que serão adotadas para o desenvolvimento, após definido a solução técnica é iniciada a construção do *software*, onde é realizada a codificação e aplicação de testes. Após estas etapas o próximo passo

é a implantação do sistema, porém no projeto SisGera não será aplicada esta fase por se tratar do desenvolvimento de um produto. (Engenharia de Software, Roger S. Pressman, p39).

Um dos problemas apresentados durante a utilização da metodologia cascata é que em alguns momentos específicos do projeto alguns membros podem ficar com tempo ocioso, por necessitar da conclusão das tarefas de outro membro da equipe. Por exemplo, para que o desenvolvedor possa realizar suas tarefas, ele necessita que o analista de requisitos conclua o levantamento e aprovação dos mesmos. Porém devido ao formato deste trabalho ser de execução por apenas um membro, este item não apresenta riscos ao projeto. (Engenharia de Software, Roger S. Pressman, p39)

5.0 Conclusão

O propósito deste trabalho foi desenvolver um *software* com linguagens de programação de código aberto, para diminuir o tempo de trabalho gasto com os processos do gerenciamento de casas de carne. Este seguimento possui carência de *softwares* no mercado, por este motivo foi o escolhido. O processo que o *software* se propõe a otimizar é um dos mais trabalhosos na visão gerencial, pois é um ponto crítico do empreendimento. Foi utilizada a linguagem Java que por se tratar de uma linguagem aberta, possui muito material a disposição do desenvolvedor, como livros, apostilas e fóruns com bom conteúdo. A utilização do banco de dados livre, o MySql também auxiliou na conquista do objetivo, pois é um banco de dados de fácil administração e com bastante conteúdo a disposição do usuário. A utilização do Hibernate proporcionou muito aprendizado na abstração de objetos, pois foi necessário aprender como se da o funcionamento do *framework*, buscar conhecimento em apostilas e livros, e desenvolver um grau elevado de abstração no que diz respeito a conceito Orientação a Objeto.

A realização deste projeto proporcionou um vasto ganho de conhecimento, a utilização da linguagem Java e o *framework* Hibernate criaram uma demanda de obtenção de conhecimento muito grande, foi gasto muito tempo apenas estudando os dois componentes sem escrever uma linha de código, para que depois, com este conhecimento, iniciar-se o desenvolvimento em si.

6.0 Trabalhos Futuros

Futuramente será desenvolvido o controle de saída dos produtos com realização de controle de estoque, pois para realizar este desenvolvimento deve se levar em consideração que seria necessário abranger mais produtos e outras regras de negócio, ou então uma possível integração com outro *software*. Na maioria dos casos são vendidos mais produtos juntamente com as carnes, e assim se perderia o foco no objetivo proposto.

Referencias Bibliográficas

Deitel, (2009) Java – Como Programar.

Daniela Barreiro Claro e João Bosco Mangueira Sobral (2008), Programação em Java.

Gavin King, Christian Bauer, Max Rydahl Andersen, Emmanuel Bernard, e Steve Ebersole - Documentação de Referência Hibernate.

Hibernate Team (Disponível em http://docs.jboss.org/hibernate/orm/3.5/reference/pt-BR/html/preface.html Acessado em 22/10/2012).

Ireports Team (2012), JasperReports Server User Guide.

Java Enterprise Edition 5 (2007) – Cleuton Sampaio de Melo Jr.

MySQL Team (2006), Manual de Referência do MySQL 4.1.

Marcelo Atie1, Marcio Belo R. Silva, Relatórios em Aplicativos WEB Instituto Superior de Tecnologia em Ciências da Computação do Rio de Janeiro.

Marcio Freitas analyst/Developer.NET
(Disponível em http://www.devmedia.com.br/uml/8579 Acessado em 15/10/2012).

Mercado Interno, Ministério da Agricultura (Disponível em http://www.agricultura.gov.br/animal/mercado-interno Acessado em 05/09/2012).

Mercado de bovinos e bubalinos, Ministério da Agricultura disponível em http://www.agricultura.gov.br/animal/especies/bovinos-e-bubalinos Acessado em 05/09/2012).

Roger S. Pressman (2006), Engenharia de Software.

Raphaela G. Fernandes, Gleydson de A. F. Lima (2007), Hibernate com Anotações.

Anexos

Caso de Uso

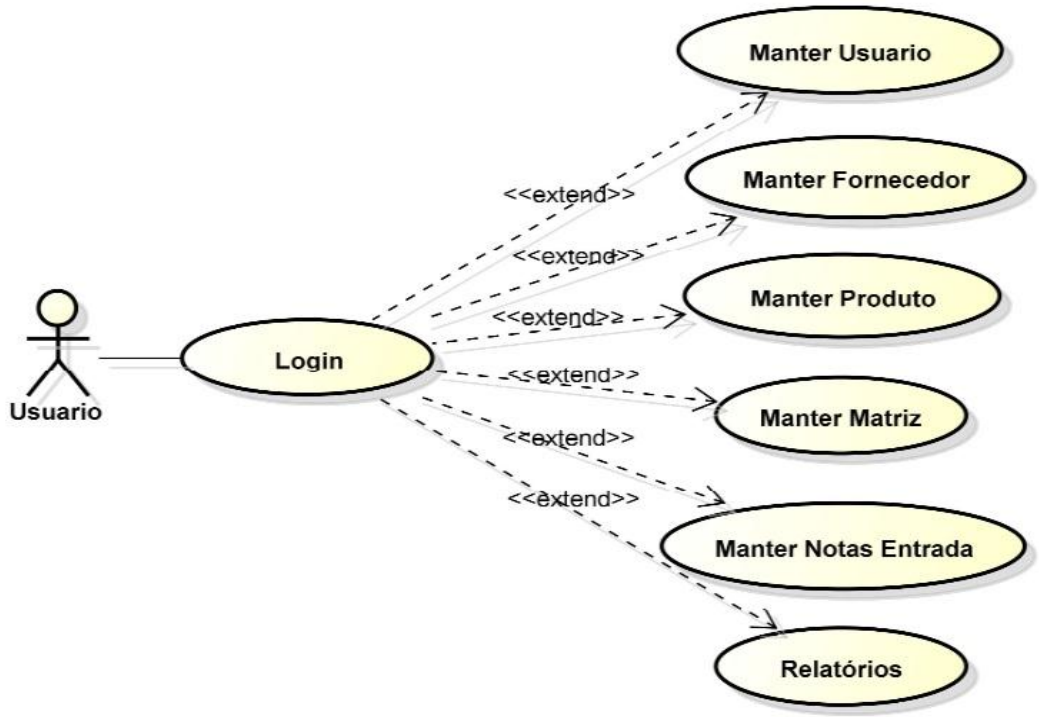


Diagrama de Classes

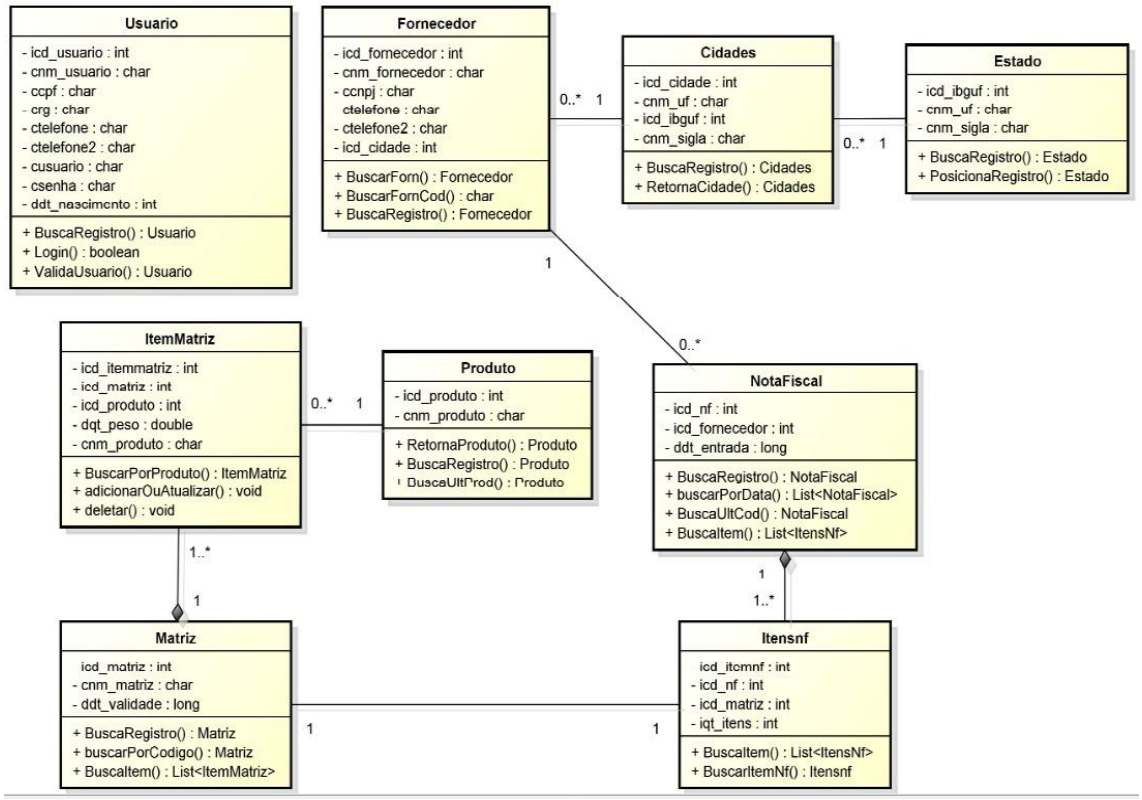
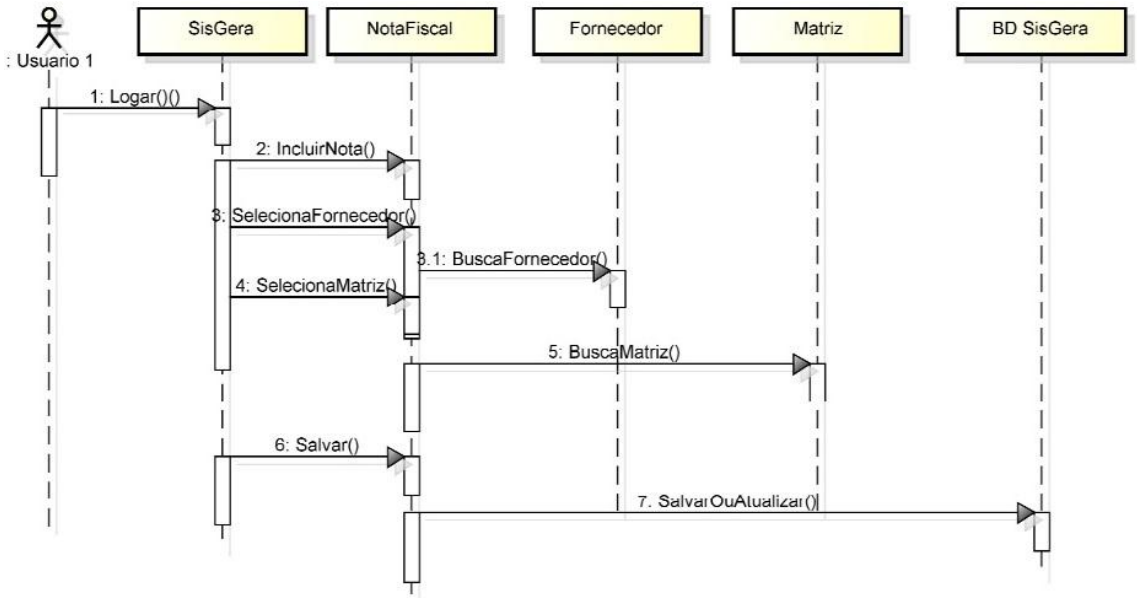


Diagrama de Sequência

Processo de Registro de Nota Fiscal



Modelagem Lógica