



FACULDADES E ESCOLAS TÉCNICAS QI

SERGIO ALFREDO HENRIQUE DA SILVA

SERGIOALFREN@GMAIL.COM

QIMK – Quick Inventory Management Kit

Porto Alegre

2019

SÉRGIO ALFREDO HENRIQUE DA SILVA

QIMK QuickInventory Management Kit

QIMK Quick Inventory Management Kit é apresentado como requisito para a aprovação na disciplina de Projeto Aplicado a Projetista, nas Faculdades e Escolas Técnicas QI

Orientador: Prof. Esp. Silvio Viegas

Porto Alegre

2019

AGRADECIMENTOS

Agradeço a Deus em primeiro lugar pela oportunidade de realizar esse curso, também à minha esposa Miriam que deveras tem me apoiado em todo o decorrer dos estudos, e minhas filhas Isabel e Rebeca que foram extremamente compreensíveis por não poderem estar comigo nos dias de aula ou enquanto eu trabalhava nos projetos, aos amigos e colegas que também de uma forma tão essencial contribuíram propiciando assim que mais uma etapa se encerre e se inicia uma nova fase onde se abrem novos horizontes.

RESUMO

Esse trabalho tem como objetivo demonstrar os conhecimentos adquiridos ao longo do curso de análise e desenvolvimento de sistemas, trata-se de um projeto de análise de *software* chamado QIMK – *Inventory Management Kit*, que propõem uma solução tecnológica para integração de dados, entre um *software* existente a um novo, auxiliar, que complementa os dados de uma mesma empresa tornando o acesso aos dados mais rápido e fácil, sem comprometer as outras funções do *software* vigente na empresa. A ideia surgiu para facilitar o acesso aos dados da empresa por pessoas de menos conhecimento tecnológico, haja vista que alguns funcionários e o próprio diretor da empresa são pessoas de um nível de conhecimento tecnológico limitado. O *software* é desenvolvido utilizando a linguagem de programação JAVA, o que possibilita a utilização dos paradigmas da programação orientada a objeto na criação de *softwares* complexos e com uma manutenibilidade fácil e ampla, contando com a utilização de *frameworks* que auxiliam na sua elaboração, tais como, o JavaFX, onde utiliza a linguagem FXML, acoplada ao Java na criação das GUI's Interface Gráfica que com o Usuário, assim, produzindo interfaces de um alto nível de usabilidade e uma performance agradável no seu desenvolvimento, disponibilizando um servidor de aplicações “GlassFish” muito poderoso e alguns bancos de dados como JavaDB e MySQL. Também é utilizado o *framework* Hibernate que remete a um solução ORM, ou seja, cria-se uma interligação entre a linguagem de programação e o banco de dados relacional e por consequência, sua utilização prove o desacoplamento a um SGBD.

Palavras-chave: Controle de estoque, Hibernate, Java, JavaFx, GlassFish, Integrações.

LISTA DE ILUSTRAÇÕES

Figura 1 – Demonstrativo de operação.....	14
Figura 2 – Tabela levantamento de requisitos	16
Figura 3 – Estrutura de diagramas	21
Figura 4 – Exemplo de diagrama de caso de uso	23
Figura 5 – Exemplo de Caso de Uso Expandido	24
Figura 6 – Exemplo de diagrama ER	25
Figura 7 – Exemplo de diagrama de Classes	26
Figura 8 – Exemplo de diagrama de atividades	27
Figura 9 – Exemplo de diagrama de sequencia	28
Figura 10 -Diagrama de Caso de Uso	38
Figura 11 – DER (Diagrama Entidade Relacionamento) Modelo Conceitual	47
Figura 12 – DER (Diagrama Entidade Relacionamento) Modelo Lógico	49
Figura 13 – Diagrama de classe	51
Figura 14 – Diagrama de Atividade cadastrar produto	52
Figura 15 – Diagrama de Atividade Atualização rápida	54
Figura 16 – Diagrama de Atividade Cadastrar Usuário	55
Figura 17 – Diagrama de Sequência Cadastrar Fornecedor	56
Figura 18 – Diagrama de Sequência Cadastrar Produto	57
Figura 19 – Diagrama de Sequência Cadastrar NF	58
Figura 20 – Tela de login	59
Figura 21 – Tela Inicial do Sistema	60
Figura 22 – Tela de cadastro de fornecedor	61
Figura 23 – Tela de cadastro de produto	62

LISTA DE TABELAS

Tabela 1 -Requisito Funcional Cadastro de produto.....	32
Tabela 2 -Requisito Funcional Cadastro de fornecedor.....	32
Tabela 3 -Requisito Funcional Atualizar produto.....	33
Tabela 4 -Requisito Funcional Atualização rápida de preço.....	33
Tabela 5 -Requisito Funcional Cadastro de quantidade.....	33
Tabela 6 -Requisito Funcional Atualizar fornecedor.....	34
Tabela 7 -Requisito Funcional Buscar produto.....	34
Tabela 8 -Requisito Funcional Busca de produto.....+	34
Tabela 9 -Requisito Funcional Remover produto.....	35
Tabela 10 -Requisito Funcional Remover fornecedor.....	35
Tabela 11 -Requisito Funcional Entrada de nota fiscal.....	35
Tabela 12 -Requisito Funcional Entrada de item.....	36
Tabela 13 -Requisito Funcional Saída de item.....	36
Tabela 14 -Requisito Funcional Cadastrar Usuário.....	37
Tabela 15 -Requisito Funcional Autenticação do Sistema.....	37
Tabela 16 -UC01 Cadastro de Produto.....	39
Tabela 17 -UC02 Cadastro de Fornecedor.....	39
Tabela 18 -UC03 Atualização de Produto.....	40
Tabela 19 -UC04 Atualização rápida de preço.....	40
Tabela 20 -UC05 Atualização rápida de quantidade.....	41
Tabela 21 -UC06 Atualizar Fornecedor.....	41
Tabela 22 -UC07 Buscar Produto.....	42
Tabela 23 -UC08 Buscar Fornecedor.....	42
Tabela 24 -UC09 Remover produto.....	43

Tabela 25 -UC10 Remover Fornecedor.....	43
Tabela 26 -UC011 Entrada de Nota fiscal.....	44
Tabela 27 -UC12 Entrada de item.....	44
Tabela 28 -UC13 Saída de Itemduto.....	45
Tabela 29 -UC14 Cadastrar Usuário.....	45
Tabela 30 -UC15 Autenticação do Sistema.....	46

LISTA DE SIGLAS E ABREVIATURAS

CSS Linguagem de folha de estilo (do inglês, *CascadingStyleSheets*).

DDL é uma linguagem para definição de dados (do inglês, *Data Definition Language*)

DER Diagrama Entidade Relacionamento.

DIP Princípio Inversão de Dependência (do inglês, *Dependency InversionPrinciple*).

FTP Protocolo de Transferência de Arquivos (do inglês, *File Transfer Protocol*).

FXML JavaFx linguagem de marcação extensiva (do inglês, *FX eXtensible MarkupLanguage*).

GUI *Interface* Gráfica com o Usuário (do inglês, *GraphicalUser Interface*).

HTTP Protocolo de Transferência de Hipertexto (do inglês, *Hypertext TransferProtocol*).

IDE Ambiente Integrado de desenvolvimento, (do inglês, *integrated developmentenvironment*).

IP Protocolo de interconexão (do Inglês, *Internet Protocol*).

ISP Princípio Segregação de *Interfaces* (do inglês, *Interface Segregation Principle*).

JPA Objeto Java para persistência de dados (do inglês, *Java Persistence API*).

JVM Máquina virtual do Java (do inglês, *Java Virtual Machine*).

LAN Rede Local (do inglês, *Local Area Network*).

LSP Princípio Substituição de Liskov (do inglês, *LiskovSubstitution*

Principle).

MER Modelo entidade relacionamento.

OCP Princípio Aberto Fechado (do inglês, *Open Closed Principle*).

OMG Organização internacional para padrões programação orientada a objeto, (do inglês, *Object Management Group*).

ORM Mapeamento objeto relacional (do inglês, *Object-Relational Mapping*).

POO Programação Orientada a Objeto.

RIA Aplicações ricas para internet (do inglês, *Rich Internet Application*).

SGBD Sistema Gerenciador de Banco de Dados.

SOLID Padrão de projeto no desenvolvimento de software, acrônimo dos princípios (SRP, OCP, LSP, ISP, DIP).

SPED Sistema Público de Escrituração Digital.

SQL Linguagem de consulta estruturada (do inglês, *Structured Query Language*).

TCP Protocolo de Controle de Transmissão (do inglês, *Transmission Control Protocol*).

UML Linguagem de Modelagem Unificada (do inglês, *Unified Modeling Language*).

WS Serviços na internet para integração de sistemas (do inglês, *Web Services*).

XML Linguagem de marcação extensiva (do inglês, *eXtensible Markup Language*).

SUMÁRIO

1. INTRODUÇÃO	11
1.2 Problema	12
1.3 Objetivo.....	13
1.3.1 Objetivo Geral	13
1.3.2 Objetivos Específicos	13
1.4 Justificativa	14
2. FUNDAMENTAÇÃO TEÓRICA	15
2.1 Levantamento de requisitos	15
2.2 Linguagem de programação	16
2.2.1 Java.....	16
2.2.2 JavaFX	18
2.2.3 Hibernate	19
2.3 Modelagem do sistema	20
2.3.1 UML.....	20
2.3.2 Modelo e Diagrama Entidade Relacionamento.....	21
3 METODOLOGIA	28
3.1 Pesquisa Aplicada	29
4 ANÁLISE DE RESULTADOS.....	31
4.1 Modelagem	31
4.2 Levantamento de requisitos	32
4.3 Diagramas	37
4.3.1 Diagrama de Caso de Uso	38
4.3.2 Caso de Uso Expandido	39
4.3.3 Diagrama ER	46
4.4.4 Diagrama de Classe	50

4.4.5 Diagrama de Atividades	52
4.4.1 Diagrama de Sequência	56
4.5 Prototipação	59
5 CONSIDERAÇÕES FINAIS.....	63
REFERÊNCIAS	64

1. INTRODUÇÃO

Essa proposta de construção de um novo sistema auxiliar de controle de estoque se dá pela complexidade de uso do sistema original tendo em vista não ser de fácil entendimento para usuários, levando em conta isso se criou uma necessidade de desenvolvimento de uma nova aplicação de fácil utilização para todos que a manusearem e que traga resultados de maneira mais rápida. A seguir serão apresentadas algumas funções básicas que o estoque deverá ter como Entrada de Nota Fiscal Automática, Gerenciamento de Compras, e Controle de Produtos. Vamos analisar cada uma dessas divisões caracterizando cada uma delas:

- Entrada de Nota Fiscal Automática: É um campo onde é feita a importação da nota fiscal eletrônica de acordo com cada fornecedor (DANFE) utilizando o arquivo XML registrando a entrada dos produtos respectivos a cada nota fiscal de maneira fácil e simples, vinculando a nota fiscal a cada fornecedor.
- Gerenciamento de compras: Trata-se de um processo de compras dividido em quatro etapas: requisição de compra, mapa de cotação, pedido de compra e recebimento da compra. Cada etapa pode ser realizada e aprovada por uma pessoa diferente. No mapa de cotação é possível comparar os preços e prazos de até três fornecedores. É possível ainda visualizar na própria tela qual foi o último preço pago por aquele produto. Com o pedido de compras pronto é possível gerar um PDF e enviar para o fornecedor.
- Controle de produtos:

Podemos ter aí também algumas divisões como:

✓ Cadastro

O cadastro de produtos será prático e intuitivo. É possível classificar o produto por setor, marca e fornecedor. O sistema pode ser configurado para trabalhar com ou sem grade de tamanho/cor. Os códigos

podem ser gerados de maneira automática ou utilizando o próprio código de barras do produto.

✓ Impressão de Etiquetas

É possível imprimir etiquetas personalizadas para os produtos, com a sua marca, código e preço. O sistema também comunica diretamente com as principais impressoras cadastradas.

● Controle de Fornecedores

Gerenciamento do cadastro dos fornecedores, com contato, telefone e endereço. Pode-se acompanhar o valor de compras feitas de cada fornecedor e o valor de venda de produtos por fornecedor. Classifica os fornecedores que têm melhor forma de pagamento ou com produtos que vendem mais.

A seguir, trataremos do problema em questão e como podemos agir de forma efetiva que satisfaça a exigência demandada estabelecendo uma forma de trabalho prática trazendo uma resolução para a necessidade instaurada.

1.2 Problema

O problema consiste em que o sistema que roda na empresa hoje é complexo para o entendimento de quem o controla, tendo assim uma má gestão do estoque da empresa utilizando dados incorretos, tudo isso porque o sistema não foi criado de uma maneira objetiva e segundo a padronização personalizada a que a demanda exigia, sendo que o sistema ficava dando muitas voltas para fazer uma tarefa básica tomando tempo desnecessário. Com a construção de um sistema auxiliar menos complexo e

mais objetivo, ou seja, definitivamente mais eficiente, teremos resultados mais rápidos e qualquer pessoa poderá extrair informações do mesmo sem muito esforço.

Como poderíamos ter uma eficiência em vendas se não estamos obtendo uma eficiência absoluta no controle e manutenção dos produtos que são vendidos na empresa?

O problema já foi detectado e agora será feita análise de uma solução para o mesmo e esse é o objetivo específico para esse projeto que depois de terminado poderá tranquilamente passar para fase de construção do sistema e implementação do mesmo.

1.3 Objetivo

1.3.1 Objetivo Geral

O objetivo é corrigir uma área da empresa que não está sendo usada plenamente que é o controle de estoque, devido à complexidade de operação do software existente, a empresa optou em desenvolver um software para esse fim, de uma maior simplicidade usual e a ser operado de forma mais intuitiva.

1.3.1 Objetivos Específicos

Para efetivar esse objetivo será primeiramente feita uma análise de um software que atenda às necessidades e que seja de fácil manuseio e entendimento para quem o operar, após análise do que deve ser feito e como ser feito for concluída pode-se começar o planejamento de codificação e implementação do sistema junto a empresa.

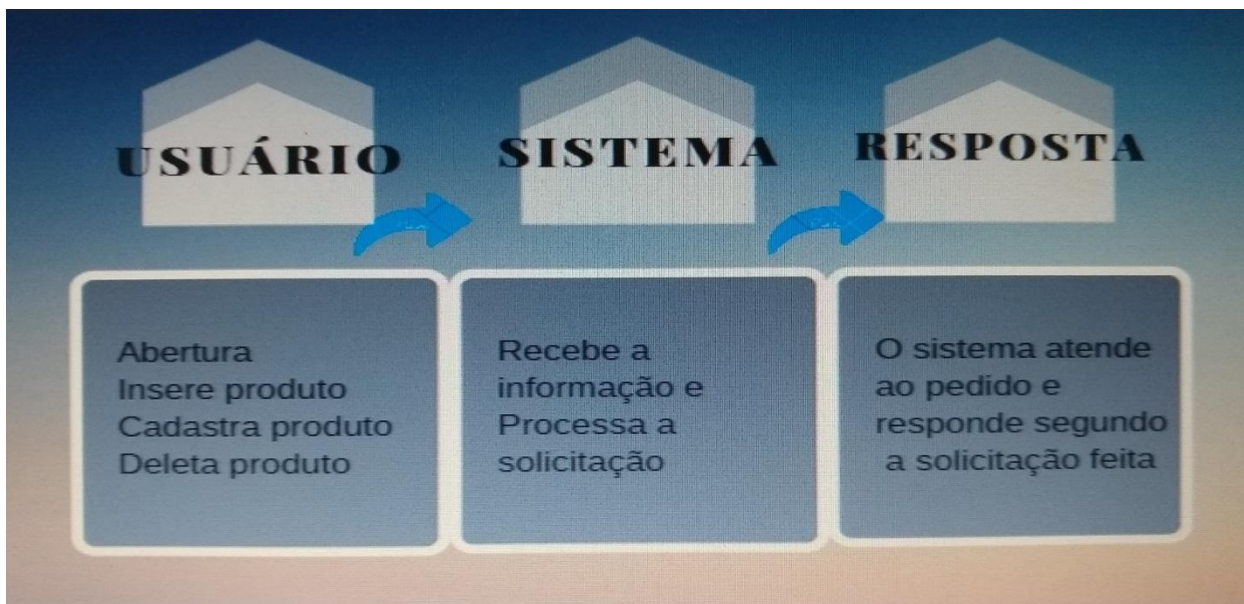
1.4 Justificativa

As possibilidades das soluções tecnológicas estão em ascendência, e as soluções alternativas são necessárias para a uma melhor gestão empresarial e acaba influenciando diretamente no bom andamento e desenvolvimento da própria empresa.

Desse modo, a elaboração de um *software* para auxílio que contribui para um melhor andamento da empresa é extremamente necessário. É óbvio que envolve um custo para desenvolver um programa além do existente, mas, sendo algo que vai somar e estabelecer um rendimento maior em todos os âmbitos da empresa, creio que é um investimento bem menor comparado com a substituição de todo o sistema vigente por outro que contemple a personalização necessária para atender a necessidade cotidiana demandada.

Segue abaixo quadro demonstrativo de operação do sistema:

Figura 1 – Demonstrativo de operação



Fonte: Elaborado pelo autor (2019)

2. FUNDAMENTAÇÃO TEÓRICA

A função dos estoques é maximizar as vendas, aperfeiçoar o planejamento e controle de produção, quanto maior o investimento, maior será o comprometimento e responsabilidade de cada departamento. Minimizar perdas e custos, otimizar Investimentos, reduzindo as necessidades de capital investido.

Conclui-se que estoque é como nossa despensa de casa, sabemos o quanto precisamos manter para suprir a nossa necessidade e de outros, sabemos onde escolher nossos fornecedores (mercados), e sabemos quando investir e o momento certo de comprar mais "itens", conforme as dificuldades do momento ou do negócio. Muitas vezes, alguns desses conceitos praticamos em nossa vida naturalmente, basta às vezes uma pequena sensibilidade de percepção para entender que somos bons administradores.

Mas, se possuímos falhas no controle, não obteremos o *feedback* necessário para que o estoque execute sua função corretamente segundo a empresa requer.





Nessa seção serão abordados os conceitos relativos à linguagem de programação e a modelagem do sistema.

2.1 Levantamentos Requisitos

Essa etapa consiste em levantar os requisitos que o software deverá atender e esse levantamento deve ser feito juntamente com o usuário que irá operá-lo para entender melhor a sua necessidade, contudo esse levantamento não é detalhado sobre como deverá ser, é mais formal como por exemplo devemos ter um requisito cadastrar

produto sem mais informações, contudo, não quer dizer que isso não será feito mais à frente.

Figura 2 – Tabela Levantamento de Requisitos

Grau de relacionamento	Símbolo	Valor
Relacionamento Nulo		0
Relacionamento Fraco		1
Relacionamento Médio		3
Relacionamento Forte		9

Fonte: Enegep, 2013.

2.2 Linguagem de programação

É a forma de comunicação “humano computador”, chamada de linguagem de programação, que é constituída por conjuntos de instruções. De acordo com Leitão (1995) as linguagens de programação devem obter simplicidade para o melhor entendimento e juntas serem capazes de criar abstrações complexas. Com isso se faz necessário a utilização de uma linguagem para a construção do sistema, nas subseções a seguir será apresentado a linguagem e seus conjuntos de ferramentas utilizada neste trabalho.

2.2.1 Java

A linguagem Java foi iniciada no ano de 1991, onde a Sun Microsystems, uma empresa originalmente fabricante de componentes para computação, resolveu financiar um projeto interno de pesquisa, liderado pelo James Gosling. Contudo a pesquisa tinha um foco na portabilidade de dispositivos e não na criação de uma linguagem, porém somente no ano de 1993 com a popularização da internet, a Sun Microsystems incluiu no Java um conteúdo dinâmico para páginas da *World Wide Web* (WEB) tornando o projeto viável para sua publicação que iria acontecer em 1995 (DEITEL; DEITEL, 2010, p. 6–7).

A partir da publicação de sua primeira versão o Java está presente em *softwares* e em diversos dispositivos móveis (MERCER, 2017). No ano de 2010 a Oracle incorporou a Sun Microsystems com suas soluções e por consequência, o Java (ORACLE).

O Java obteve sua reputação devido as suas características assim apresentadas:

- **Orientado a objetos:** Refere-se a forma de elaboração do código fonte, onde é possível criar uma abstração utilizando objetos e classes na elaboração do *software* (CORRÊA, 2016, p. 50–51).
- **Multi-plataforma:** O Java apresenta desde sua criação a *Java Virtual Machine*(JVM) onde é possível que as aplicações Java sejam executadas em diversos Sistemas Operacionais(HOFFMAN, 2016).
- **Tipagem:** Outra característica importante relacionada ao Java é sua tipagem, ou seja, como uma variável irá se comportar no código fonte, nesse caso o Java é uma linguagem fortemente tipada, isto é: Uma variável quando criada ela precisa ter um tipo (Inteira, *String*[palavra], *Float*[número decimal]) e o mesmo é imutável (HORSTMANN; CORNELL, 2015, p. 21).
- **Compatibilidade com redes:** O Java disponibiliza diversas bibliotecas

(conjuntos de código fonte) para o manuseio nas aplicações na *WEB* e nas redes locais (LAN) por meio dos seguinte protocolos: Protocolo de Controle

21

de Transmissão (TCP), Protocolo de interconexão (IP), Protocolo de Transferência de Hipertexto (HTTP), Protocolo de Transferência de Arquivos (FTP) (HORSTMANN; CORNELL, 2015, p. 2).

O Java apresenta outras características interessantes, porém necessário é o conhecimento das *Interfaces GraphicalUser Interface* (GUI) que permite ao usuário manipular o software. Que será tratado na próxima subseção.

2.2.2 JavaFX

O JavaFX é uma biblioteca multimídia do Java criada pela Sun Microsystems no ano de 2007 para desenvolvimento das GUI's com o foco na *Rich Internet Application*(RIA), ou seja, aplicações ricas para internet que podem ser executadas em 3 âmbitos: *Desktop* (software que são executados localmente), Web (sites) e Aplicações *Mobile* (SRIVASTAVA, 2016). Porém isso só foi possível devido aos seus atributos assim apresentados:

- **FXML:** *JavaFxMarkupLanguage*(FXML) é a linguagem elaborada para o desenvolvimento das *Interfaces*, sua estrutura é criada separadamente da estrutura de código fonte, assim obtendo uma melhor performance em tempo de execução, agilidade e segurança no desenvolvimento (DEITEL; DEITAL, 2016, p. 864).
- **SceneBuilder:** O SceneBuilder é um software desenvolvido para a criação de FXML, totalmente a parte do Java. Ele proporciona a criação das *Interfaces* GUI's com um conceito de *draganddrop*, em português o arrasta e solta, com isso possibilitando uma subdivisão entre desenvolvimento e

criação de *Interfaces* (MARX, 2011). Contudo outros aspectos relacionados ao SceneBuilder são importantes, conforme alega Vaid (2017) a possibilidade de estender os componentes visuais com a utilização bibliotecas e a distribuição do código-fonte pela Oracle à comunidade Java, efetivamente, trazem uma vantagem a essa aplicação e ao futuro das aplicações RIA utilizando o Java.

- **CSS:** *CascadingStyleSheets*(CSS) ou folhas de estilo é uma linguagem de marcação utilizada para definição da aparência nas páginas Web, criada pelos Srs. Håkon Wium Lie e Bert Bos no ano de 1994 e lançado oficialmente no ano de 1996 pela comunidade W3C (EIS, 2006). Logo a Sun
22

incorporou o CSS ao FXML permitindo a utilização dos todos componentes de uma página Web no JavaFX (PAWLAN, 2013).

Outra questão relacionada ao JavaFX é a possibilidade de geração de Gráficos e os recursos de terceira dimensão (3D) e de acordo com Deitel e Deitel (2016, p. 864) esses recursos do JavaFX podem fazer aplicações mais intuitivas e fáceis de usar.

2.2.3 Hibernate

Hibernate é um *framework* (caixa de ferramentas) criado pela empresa JBoss no ano de 2003, foi criado para a implementação da *Java Persistence API* (JPA), que é a responsável pela persistência de dados (BAUER; KING, 2006, p. 27).

Especialmente o Hibernate foi arquitetado para o tratamento do *Object-Relational Mapping*(ORM), ou seja, esse *framework* gerencia o conceito da programação orientada a objeto para a lógica relacional dos bancos de dados, assim criando as sentenças já utilizada nos bancos de dados relacionais a linguagem *Structured Query Language*(SQL)(FRAGOSO, 2008).

Contudo sua utilização é opcional, no entanto a sua implementação remete benefícios ao desenvolvimento de *softwares*, conforme alega Faria e Junior (2015, p. 15), a manutenibilidade do *software* fica acessível, transparente e possibilita a abstração do sistema gerenciado de banco de dados (SGBD) utilizado no software.

2.3 Modelagem de sistemas

A modelagem de sistemas é uma forma que possibilita a abstração que será utilizada na assimilação de um sistema e seus requisitos, como por exemplo os diagramas.

A modelagem pode ser empregada em sistemas já existentes e/ou em novos sistemas

Os diagramas são uma representação gráfica de atividades que serão desenvolvidas no sistema, logo em seguida iremos apresentar vários tipos de diagramas e cada um tem sua finalidade para melhor entendimento de como irá funcionar o sistema, sendo alguns simples e de fácil entendimento até para um usuário. Em contrapartida temos alguns mais complexos e outros mais detalhistas, mas todos são importantes para construção dessa análise.

2.3.1 UML

Basicamente, UML (Unified Modeling Language) é uma linguagem de notação (um jeito de escrever, ilustrar, comunicar) para uso em projetos de sistemas.

Esta linguagem é expressa através de diagramas. Cada diagrama é composto por elementos (formas gráficas usadas para os desenhos) que possuem relação entre si. Os diagramas da UML se dividem em dois grandes grupos: diagramas estruturais e

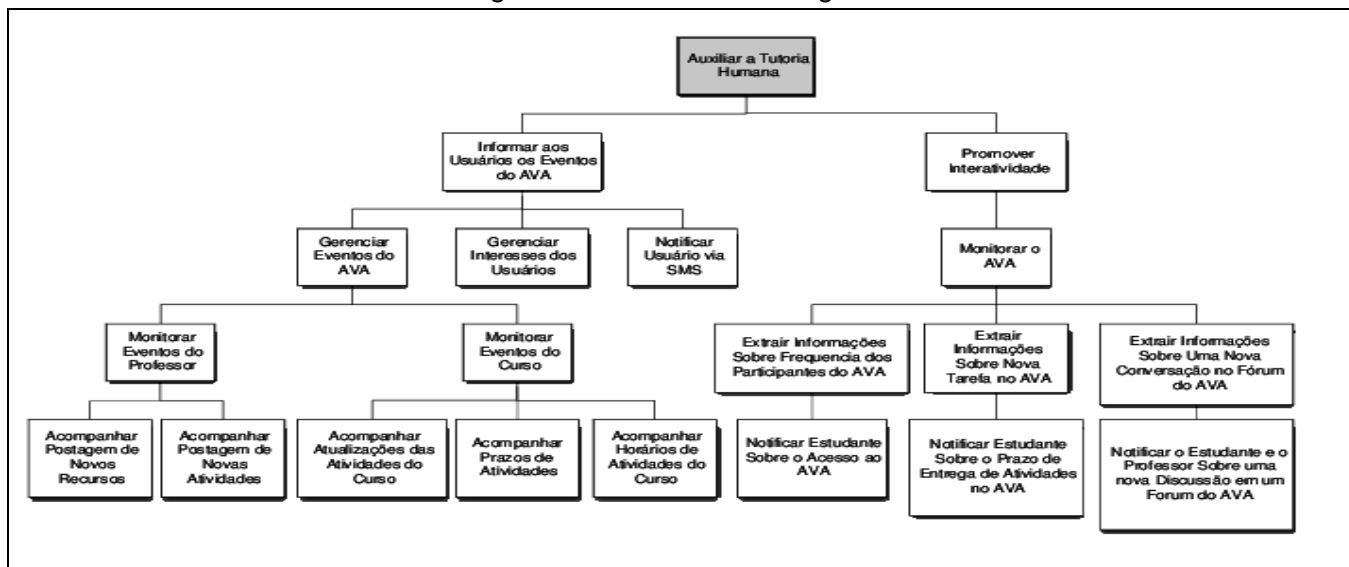
diagramas comportamentais.

Diagramas estruturais devem ser utilizados para especificar detalhes da estrutura do sistema (parte estática), por exemplo: classes, métodos, interfaces, namespaces, serviços, como componentes devem ser instalados, como deve ser a arquitetura do sistema etc.

2.3.2 Modelo e Diagrama Entidade Relacionamento

Diagramas comportamentais devem ser utilizados para especificar detalhes do comportamento do sistema (parte dinâmica), por exemplo: como as funcionalidades devem funcionar, como um processo de negócio deve ser tratado pelo sistema, como componentes estruturais trocam mensagens e como respondem às chamadas etc.

Figura 3 – Estrutura de Diagramas



Fonte: OMG (2005) com alterações do autor, 2018.

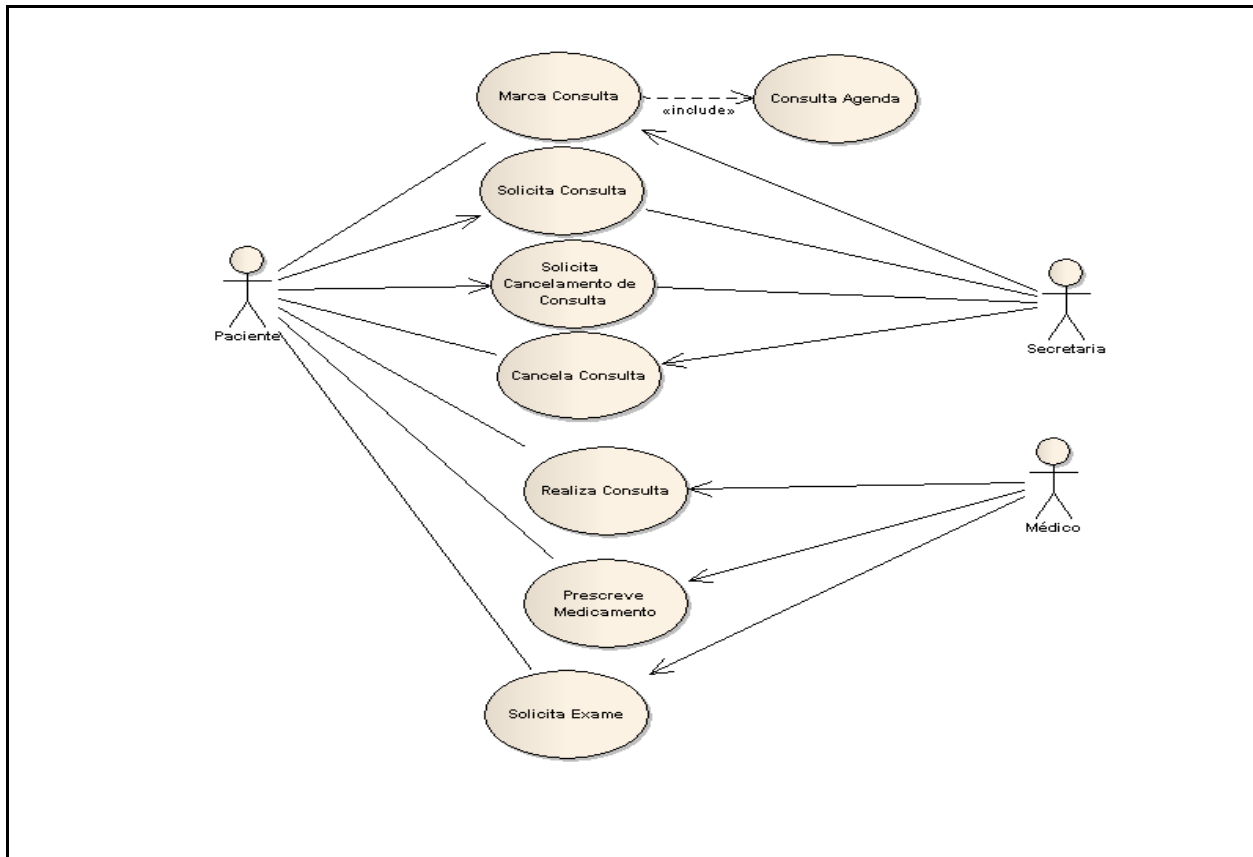
Segue abaixo os diagramas que serão apresentados nesse projeto:

- Diagrama de Caso de Uso
- Caso de Uso Expandido
- Diagrama ER
- Diagrama de Classes
- Diagrama de Atividades
- Diagrama de Sequência

O **Diagrama de Caso de Uso** descreve a funcionalidade proposta para um novo sistema que será projetado, é uma excelente ferramenta para o levantamento dos requisitos funcionais do sistema. Segundo Ivar Jacobson, podemos dizer que um caso de uso é um "documento narrativo que descreve a sequência de eventos de um ator que usa um sistema para completar um processo". Um caso de uso representa uma unidade discreta da interação entre um ator (humano, dispositivo ou outro software[1]) e o sistema. Um caso de uso é uma unidade de um trabalho significativo. Por exemplo: o "login para o sistema", "registrar no sistema" e "criar pedidos" são todos casos de uso. Cada caso de uso tem uma descrição da funcionalidade que será construída no sistema proposto. Um caso de uso pode "incluir" outra funcionalidade de caso de uso ou "estender" outro caso de uso com seu próprio comportamento.

Casos de uso são tipicamente relacionados a "atores". Um ator é um humano ou entidade máquina que interage com o sistema para executar um significativo trabalho.

Figura 4 – Exemplo de Diagrama de Caso de Uso



Fonte: Diagramas de Caso de Uso: Introdução Prática à UML, 2019.

A **Descrição Expandida de Casos de Uso** é utilizada para descrever detalhadamente o fluxo do caso de uso, quais os eventos que acontecem para garantir o sucesso do caso de uso.

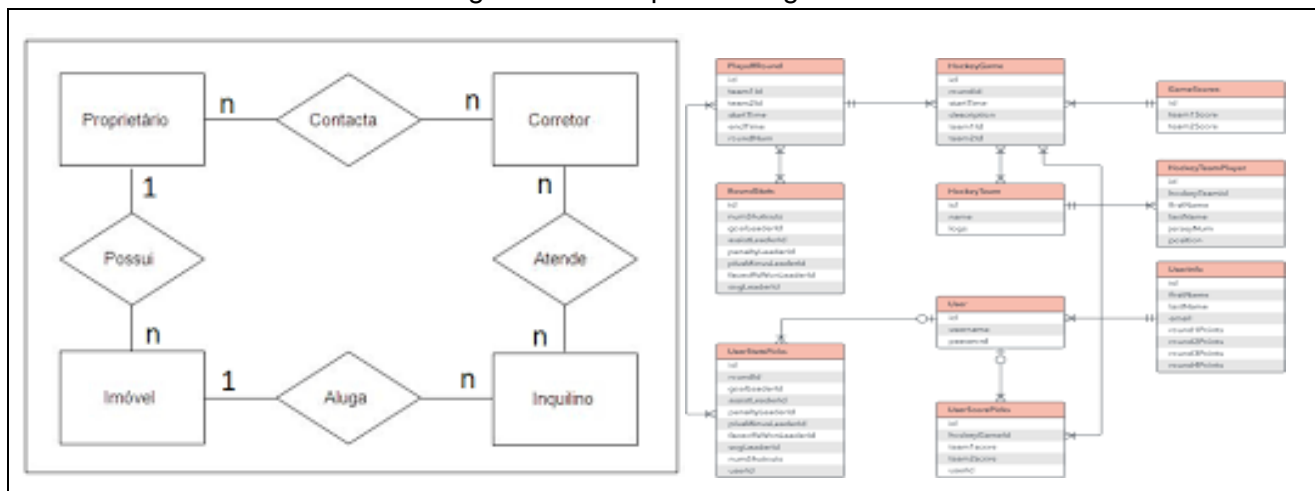
Figura 5 – Exemplo de Caso de Uso Expandido

Caso de uso: Finalizar Compra
Referências: RF_F2
Descrição Geral: O caso de uso inicia-se quando cliente deseja efetuar compra dos produtos que estão inseridos no carrinho de compras.
Atores: Cliente.
Pré-condições: Cliente logado no sistema, produtos já inseridos no carrinho de compras.
Garantia de sucesso (Pós-condições): Pedido fechado, compra efetuada, sistema aguardando confirmação de pagamento.
Requisitos Especiais:
Fluxo Básico: <ol style="list-style-type: none">1. Cliente deseja finalizar compra, sistema solicita que informe a forma de pagamento e de entrega.2. Cliente deseja efetuar pagamento em forma de cartão crédito/débito.3. Sistema solicita informações do cartão do cliente.4. Sistema faz validação das informações.5. Sistema gera o número do pedido.6. Compra finalizada com sucesso.
Fluxo Alternativo: <ol style="list-style-type: none">1. Cliente deseja efetuar pagamento através de boleto bancário.<ol style="list-style-type: none">1. Sistema gera o boleto para o cliente. Retorna ao passo 5.

Fonte: Treinamento WAEI/MSEML, 2019.

Um **Diagrama Entidade Relacionamento (ER)** é um tipo de fluxograma que ilustra como “entidades”, p. ex., pessoas, objetos ou conceitos, se relacionam entre si dentro de um sistema. Diagramas ER são mais utilizados para projetar ou depurar bancos de dados relacionais nas áreas de engenharia de software, sistemas de informações empresariais, educação e pesquisa. Também conhecidos como DERs, ou modelos ER, usam um conjunto definido de símbolos, tais como retângulos, diamantes, ovais e linhas de conexão para representar a interconectividade de entidades, relacionamentos e seus atributos. Eles espelham estruturas gramaticais, onde entidades são substantivos e relacionamentos são verbos.

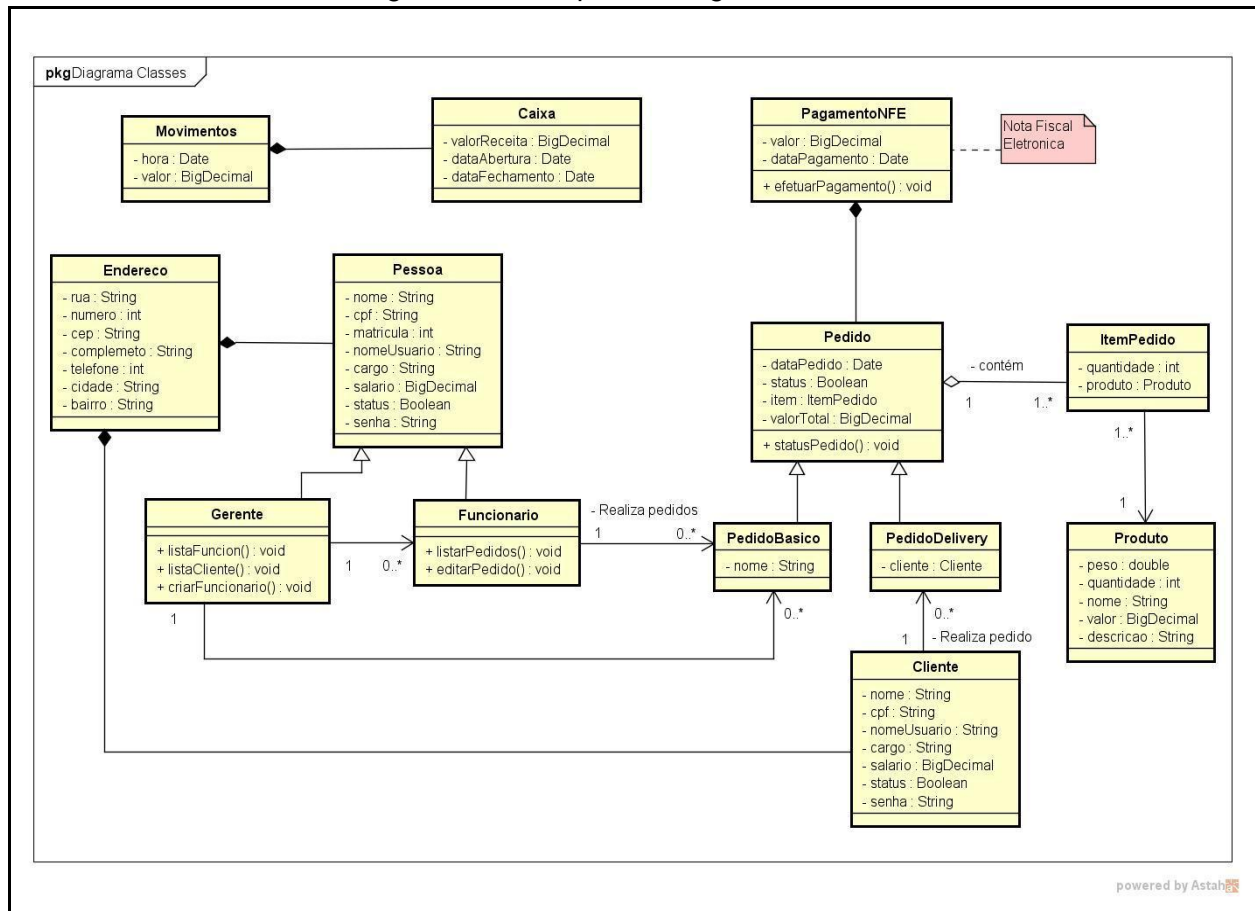
Figura 6 –Exemplo de Diagrama ER



Fonte: Lucidchart, 2019.

O **Diagrama de Classes** é uma representação da estrutura e relações das classes que servem de modelo para objetos. Podemos afirmar de maneira mais simples que seria um conjunto de objetos com as mesmas características, assim saberemos identificar objetos e agrupá-los, de forma a encontrar suas respectivas classes. Na UnifiedModelingLanguage (UML) em diagrama de classe, uma classe é representada por um retângulo com três divisões, são elas: O nome da classe, seus atributos e por fim os métodos. Veja abaixo na Figura 7 sua representação:

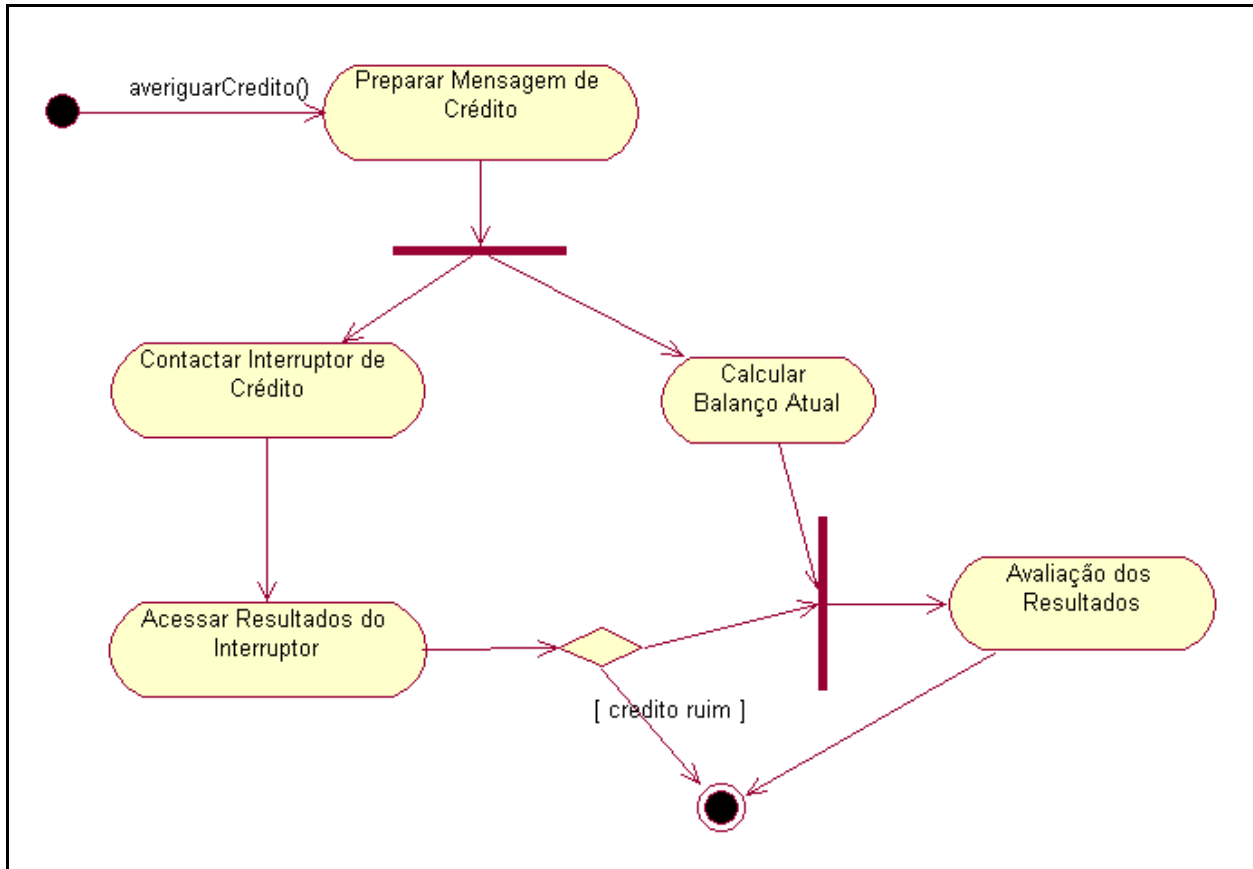
Figura 7 – Exemplo de Diagrama de Classes



Fonte: Diagramas de Classes: Introdução Prática à UML, 2019.

Um **diagrama de atividade** é essencialmente um gráfico de fluxo, mostrando o fluxo de controle de uma atividade para outra e serão empregados para fazer a modelagem de aspectos dinâmicos do sistema. Na maior parte, isso envolve a modelagem das etapas sequenciais em um processo computacional; Enquanto os diagramas de interação dão ênfase ao fluxo de controle de um objeto para outro, os diagramas de atividades dão ênfase ao fluxo de controle de uma atividade para outra; Uma atividade é uma execução não atômica em andamento em uma máquina de estados e acabam resultando em alguma ação, formada pelas computações atômicas executáveis que resultam em uma mudança de estado do sistema ou o retorno de um valor.

Figura 8 – Exemplo de Diagrama de Atividades

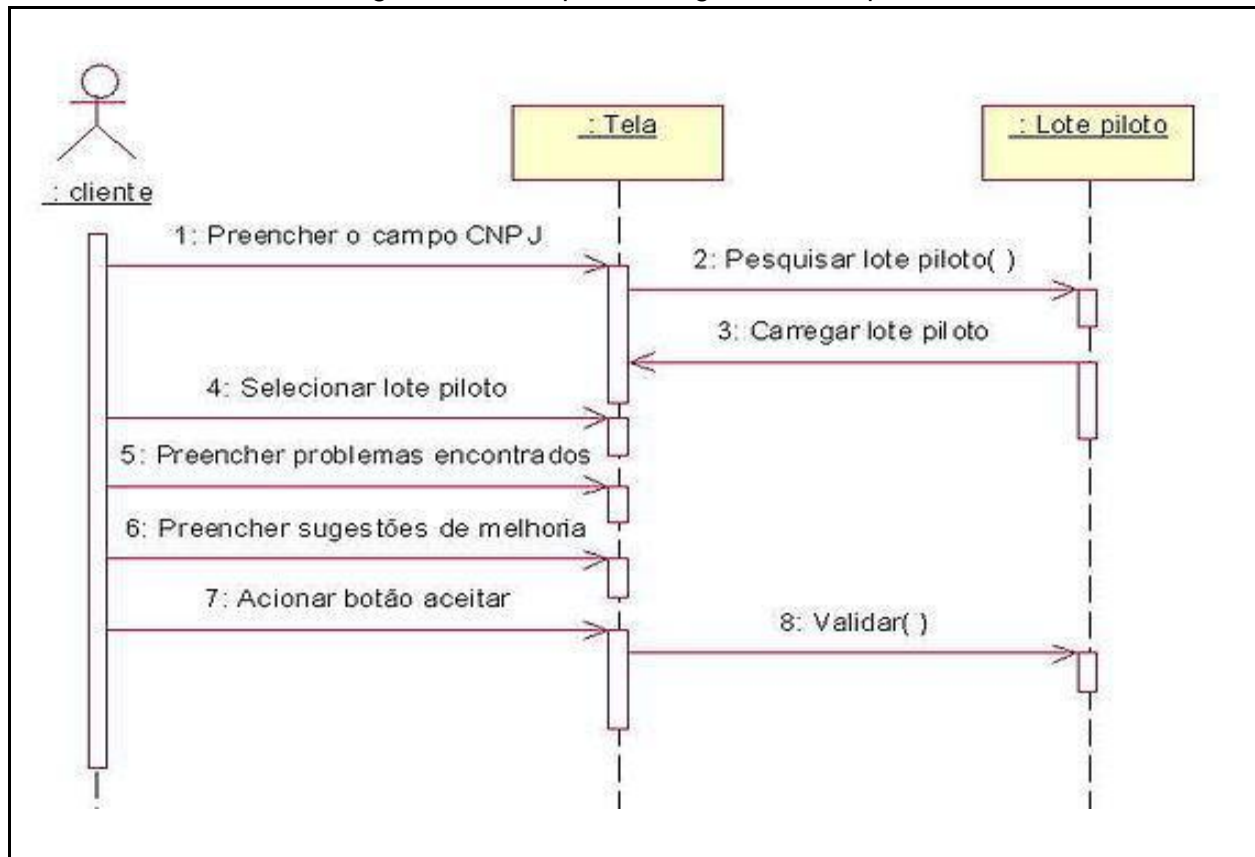


Fonte: Diagramas de Atividades: Introdução Prática à UML, 2019.

Diagrama de sequência (ou Diagrama de Sequência de Mensagens) é um diagrama usado em UML (Unified Modeling Language), representando a sequência de processos (mais especificamente, de mensagens passadas entre objetos) num programa de computador. Como um projeto pode ter uma grande quantidade de métodos em classes diferentes, pode ser difícil determinar a sequência global do comportamento. O diagrama de sequência representa essa informação de uma forma simples e lógica.

Um diagrama de sequência descreve a maneira como os grupos de objetos colaboram em algum comportamento ao longo do tempo. Ele registra o comportamento de um único caso de uso e exibe os objetos e as mensagens passadas entre esses objetos no caso de uso.

Figura 9 – Exemplo de Diagrama de sequencia



Fonte: Diagramas sequencia: Introdução Prática à UML, 2019.

3 Metodologia

A Metodologia corresponde a um conjunto de procedimentos a ser utilizado na obtenção do conhecimento”. Esses conjuntos supracitados são conhecidos como métodos e conforme Barros e Lehfel'd (2007, p. 3) “O método é o caminho ordenado e sistemático para se chegar a um fim”. Nesse conceito, este trabalho apresenta o método de pesquisa aplicada, que será abordado na próxima subseção.

3.1 Pesquisa Aplicada

A Pesquisa científica é uma atividade que cria uma relação de proximidade entre a realidade e a investigação. Pesquisa científica é o produto de uma investigação detalhada e meticulosa, que busca responder o problema proposto, alicerçando-se em mecanismos científicos. Ou seja, a pesquisa científica utiliza-se de uma associação de métodos e técnicas empregados pelos pesquisadores para alcançar e produzir novos conhecimentos ou complementar conhecimentos já existentes.

Assim a pesquisa científica segue todos os processos e normas metodológicas para que a investigação seja validada e representativa. Desta maneira, todas as etapas são organizadas sequencialmente, como a escolha do tema e do problema de pesquisa, qual metodologia será adotada, averiguação dos resultados e a comunicação dos resultados. Já a estrutura da pesquisa foi elaborada visando compreender a regra de negócio, e está predisposta conforme Quadro 1.

Quadro 1 – Metodologia do projeto QIMK

1.1 Área de atuação; 1.2 Linguagem de Programação; 1.3 Banco de Dados;		1 Definição
2 Conhecimento	2.1 Estoque; 2.2 Plataformas de Gerenciamento; 2.3 Regras de Negócio;	
3.1 Moldes Estruturais ER; 3.1 Moldes Conceituais POO;		3 Modelagem

Fonte: O autor, 2019.

O Quadro 1 apresenta a segmentação que será utilizada ao longo do projeto QIMK, contudo está subdividida para melhor entendimento:

1. Definição: Nesse primeiro segmento são definidas as especificações macros do projeto, que são decisivas no seu desenvolvimento.

1.1 - Área de atuação: nesse ponto é identificado o foco do desenvolvimento, onde o quesito de um software multifornecedores e multicientes é importante, ou seja, com esse quesito é possível utilizar o software para vários fornecedores e clientes (CNPJ's) em uma única instalação.

1.2 Linguagem de Programação: foi utilizado a linguagem de programação Java na construção do software, pelo fato de ser a linguagem de programação que consta como base de ensino na instituição FAQI.

1.3 Banco de dados: foi utilizado o SGBD da Microsoft o SQL Server, essa escolha foi novamente embasada no conhecimento adquirido ao longo da grade curricular na instituição FAQI.

2. Conhecimento: Esse segmento é utilizado na capacitação das regras e leis que influenciam no desenvolvimento do projeto.

2.1 Estoque: Esse tópico tem como base a pesquisa e compreensão do que é o Estoque e quais são os pontos que o levaram a ter uma grande consideração na justificativa desse trabalho.

2.2 Plataformas de Gerenciamento: É necessário que se tenha conhecimento do software existente e como ele opera, para que se saiba se é possível a adaptação para esse novo conceito de software, e também verificar outras plataformas existentes para que se tenha ideia do que já existe no mercado, e o que ainda podemos implementar segundo a regra de negócio.

2.3 Regras de negócio: Esse tópico foi utilizado para a compreensão macro das possíveis regras de negócio que podem estar envolvidas nesse benefício da empresa.

3 Modelagem: Nesse segmento são definidas as especificações relativas a modelagem conceitual desse trabalho.

3.1 Moldes Estruturais ER: Esse diagrama é utilizado para retratar a visão da estrutura de

armazenamento dos dados.

3.2 Moldes Conceituais POO: Os diagramas escolhidos para retratar o projeto foram o diagrama de atividade, visando o fluxo de movimentação dos dados do software, diagrama de caso de uso que visa demonstrar as funções que serão realizadas por cada um no software e o diagrama de classe para abstrair a ideia dos objetos constantes no QIMK.

Como este trabalho se trata de uma análise, ou seja, não haverá um desenvolvimento na íntegra, pelo menos de momento. Assim será desnecessário detalhar o desenvolvimento do software.

A análise será criada pela experiência do usuário ao utilizar o *software* em seu dia a dia, assim criando uma retroalimentação das regras de negócio e da usabilidade. No próximo tópico será demonstrado a seção de Análise dos resultados.

4. ANÁLISE DOS RESULTADOS

O escopo desse tópico é a apresentação dos resultados obtidos com o projeto do QIMK *Quick Inventory Management Kit*, onde foram percorridas as bases teóricas e o conhecimento adquirido na instituição, resultando na análise de desenvolvimento do *software*, análise do problema e uma possível solução, conforme serão demonstradas nas seguintes subseções.

4.1 Modelagem

A modelagem do QIMK foi desenvolvida segundo as regras estabelecidas diretamente com a empresa que demandou o serviço, e assim foram definidas as regras de negócio, bem como a concepção da modelagem do software na sua totalidade, tendo sido estabelecida a apuração dos resultados, conforme as

próximas subseções.

4.2 Levantamento Requisitos

O levantamento de requisitos do sistema que estamos analisando foi feito juntamente com quem irá operar o sistema, sendo assim obtidas as informações necessárias para que pudesse ser atingida a expectativa, de forma que o software pudesse ser eficiente segundo a demanda gerada, segue abaixo o levantamento de requisitos do sistema.

Tabela 1 – Requisito Funcional Cadastro de produto

Identificador	RF01		
Nome	Cadastro de produto		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a cadastrar um produto no sistema.		

Tabela 2 – Requisito Funcional Cadastro de fornecedor

Identificador	RF02		
Nome	Cadastro de fornecedor		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta

Descrição	Esse requisito destina-se a cadastrar um fornecedor no sistema.
------------------	---

Tabela 3 – Requisito Funcional Atualizar produto

Identificador	RF03		
Nome	Atualizar produto		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a atualizar um produto no sistema.		

Tabela 4 – Requisito Funcional Atualização rápida de preço

Identificador	RF04		
Nome	Atualização rápida de preço		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a atualizar apenas o preço do produto no sistema.		

Tabela 5 – Requisito Funcional Cadastro de quantidade

Identificador	RF05		
Nome	Atualização rápida de quantidade		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima		Autor	

Modificação			
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a atualizar apenas a quantidade de um produto no sistema.		

Tabela 6 – Requisito Funcional Atualizar fornecedor

Identificador	RF06		
Nome	Atualizar fornecedor		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a atualizar um fornecedor no sistema.		

Tabela 7 – Requisito Funcional Buscar produto

Identificador	RF07		
Nome	Buscar produto		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se apenas buscar um produto no sistema.		

Tabela 8 – Requisito Funcional Buscar fornecedor

Identificador	RF08		
Nome	Buscar fornecedor		
Data da criação	13/08/2017	Autor	Sérgio Henrique

Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se apenas a buscar um fornecedor no sistema.		

Tabela 9 – Requisito Funcional Remover produto

Identificador	RF09		
Nome	Remover produto		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a remover um produto do sistema.		

Tabela 10 – Requisito Funcional Remover fornecedor

Identificador	RF10		
Nome	Remover fornecedor		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a remover um fornecedor do sistema.		

Tabela 11 – Requisito Funcional Entrada de nota fiscal

Identificador	RF11		
Nome	Entrada de nota fiscal		

Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a capturar uma nota fiscal e registrar todos os seus dados no sistema.		

Tabela 12 – Requisito Funcional Entrada de item

Identificador	RF12		
Nome	Entrada de item		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a inserção automática de item no sistema mediante a inserção da nota fiscal no sistema.		

Tabela 13 – Requisito Funcional Saída de item

Identificador	RF13		
Nome	Saída de item		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a saída automática de item referenciada a cada venda no sistema.		

Tabela 14 – Requisito Funcional Cadastrar Usuário

Identificador	RF14		
Nome	Cadastrar usuário		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a cadastrar um usuário no sistema.		

Tabela 15 – Requisito Funcional Autenticação do Sistema

Identificador	RF15		
Nome	Autenticação do sistema		
Data da criação	13/08/2017	Autor	Sérgio Henrique
Data da ultima Modificação		Autor	
Versão	1.0	Prioridade	Alta
Descrição	Esse requisito destina-se a entrada no sistema através de login e senha.		

4.3 Diagramas

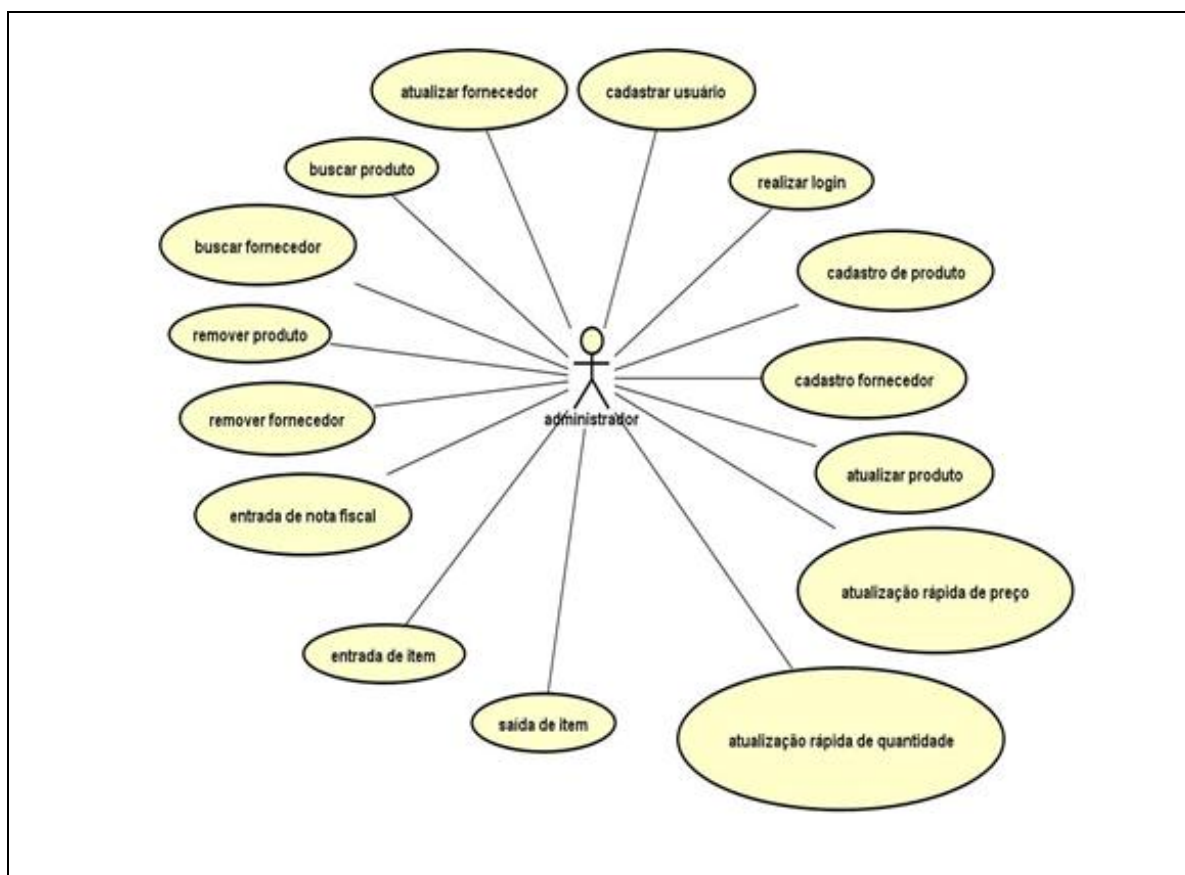
Segue abaixo os diagramas do sistema estabelecendo suas definições e

representando todas as suas funcionalidades:

4.3.1 Diagrama de caso de uso

Esse diagrama representa de forma visual os requisitos que foram levantados junto ao usuário. Ele usa um ator que pode ser a representação do usuário no sistema e mostra todas as atividades que o ator poderá realizar no sistema podendo umas tarefas serem vinculadas a outras e quando usuário disparar uma outra obrigatoriamente poderá ser disparada, segue abaixo o diagrama de caso de uso do nosso sistema.

Figura 10 - Diagrama de Caso de Uso Sistema Controle de estoque



Fonte: Elaborado pelo autor (2019)

Acima na Figura 10 vemos o diagrama de caso de uso onde são demonstradas as ações que serão realizadas pelo ator (administrador).

O Administrador terá privilégios para controlar todas as funcionalidades do sistema, nesse caso o sistema terá apenas uma pessoa que será designada como administrador e terá total liberdade junto ao sistema.

4.3.2 Caso de uso expandido

O caso de uso expandido é o detalhamento dos casos de uso especificados anteriormente, esse detalhamento envolve até mesmo o que terá nas telas o que deverá acontecer ou seja qual o fluxo da atividade com sua descrição como fluxo principal e também deve apresentar um fluxo alternativo que é para informar possíveis erros ou simplesmente retorno de sucesso para os usuários, segue abaixo o caso de uso expandido do sistema que está em análise neste projeto.

Tabela 16 – UC01 Cadastro de Produto

Descrição do UC01 – Cadastro de Produto	
Objetivo	Permite que o Administrador faça o cadastro de um produto no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	1.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 1.2. Ao logar-se o administrador tem acesso aos produtos e cadastros dos mesmos e pode então cadastrar todo e qualquer item.
Fluxo Alternativo	1.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 1.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 17 – UC02 Cadastro de Fornecedor

Descrição do UC02 – Cadastro de Fornecedor	
Objetivo	Permite que o Administrador faça o cadastro de um fornecedor no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	2.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 2.2. Ao logar-se o administrador tem acesso aos fornecedores e cadastros dos mesmos e pode então cadastrar todo e qualquer fornecedor.
Fluxo Alternativo	2.3. Se o login não estiver correto o acesso não poderá ser realizado. 2.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 2.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 18 – UC03 Atualização de Produto

Descrição do UC03 – Atualização de Produto	
Objetivo	Permite que o Administrador atualize o cadastro de um produto no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	1.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 1.2. Ao logar-se o administrador tem acesso aos produtos e cadastros do mesmos e pode então atualizar todo e qualquer item.
Fluxo Alternativo	1.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 1.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 19 – UC04 Atualização rápida de preço

Descrição do UC04 – Atualização rápida de preço	
Objetivo	Permite que o Administrador faça atualização rápida de preço em qualquer produto do sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	4.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 3.2. Ao logar-se o administrador tem acesso aos produtos e cadastros dos mesmos e pode então atualizar rapidamente o preço de todo e qualquer item.
Fluxo Alternativo	4.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 4.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 20 – UC05 Atualização rápida de quantidade

Descrição do UC05 – Atualização rápida de quantidade	
Objetivo	Permite que o Administrador faça atualização rápida de quantidade em qualquer produto do sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	5.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 5.2. Ao logar-se o administrador tem acesso aos produtos e a quantidade dos mesmos e pode então atualizar rapidamente a quantidade de todo e qualquer item.
Fluxo Alternativo	5.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 5.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 21 – UC06 Atualizar Fornecedor

Descrição do UC06 – Atualizar fornecedor	
Objetivo	Permite que o Administrador faça atualização no cadastro de fornecedores cadastrados no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	6.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 6.2. Ao logar-se o administrador tem acesso ao cadastro de fornecedores e pode então atualizar os dados dos fornecedores cadastrados no sistema.
Fluxo Alternativo	6.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 6.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 22 – UC07 Buscar Produto

Descrição do UC07 – Buscar produto	
Objetivo	Permite que o Administrador faça buscas de produtos cadastrados do sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	7.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 7.2. Ao logar-se o administrador tem acesso aos produtos e cadastros dos mesmos e pode então buscar produtos cadastrados no sistema.
Fluxo Alternativo	7.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 7.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 23 – UC08 Buscar Fornecedor

Descrição do UC08 – Buscar Fornecedor	
Objetivo	Permite que o Administrador faça buscas de fornecedores cadastrados no sistema
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	8.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 8.2. Ao logar-se o administrador tem acesso ao cadastro de fornecedores e pode fazer buscas de todo e qualquer fornecedor cadastrado no sistema.
Fluxo Alternativo	8.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 8.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 24 – UC09 Remover produto

Descrição do UC09 – Remover produto	
Objetivo	Permite que o Administrador remova produtos cadastrados no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	9.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 9.2. Ao logar-se o administrador tem acesso aos produtos e cadastros dos mesmos e pode então atualizar rapidamente o preço de todo e qualquer item.
Fluxo Alternativo	9.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 9.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 25 – UC10 Remover Fornecedor

Descrição do UC10 – Remover fornecedor	
Objetivo	Permite que o Administrador remova fornecedores cadastrados no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	10.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 10.2. Ao logar-se o administrador tem acesso ao cadastro de fornecedores e pode então remover fornecedores cadastrados no sistema.
Fluxo Alternativo	10.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 10.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 26 – UC011 Entrada de Nota fiscal

Descrição do UC11 – Entrada de Nota fiscal	
Objetivo	Permite que o Administrador dê entrada de nota fiscal eletrônica no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	11.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 11.2. Ao logar-se o administrador pode inserir nota fiscal de compra no sistema com todos os dados podendo atualizar automaticamente a quantidade dos produtos cadastrados referenciados aos itens da nota fiscal
Fluxo Alternativo	11.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 11.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 27 – UC12 Entrada de item

Descrição do UC12 – Entrada de item	
Objetivo	Permite que o Administrador dê entrada dos itens nota fiscal eletrônica quando ela é inserida no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	12.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 12.2. Ao logar-se o administrador pode inserir nota fiscal de compra no sistema com todos os dados podendo atualizar automaticamente a quantidade dos produtos cadastrados referenciados aos itens da nota fiscal
Fluxo Alternativo	12.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 12.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 28 – UC13 Saída de Item

Descrição do UC13 – Saída de Item	
Objetivo	Permite que o item tenha saída do sistema no ato da venda de produtos cadastrados no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Automático
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	13.1. Este Caso de Uso se inicia quando uma venda é realizada no sistema.
Fluxo Alternativo	

Tabela 29 – UC14 Cadastrar Usuário

Descrição do UC14 – Cadastrar usuário	
Objetivo	Permite que o Administrador cadastre um usuário no sistema.

Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Fazer login no sistema
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	14.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo. 14.2. Ao logar-se o administrador pode cadastrar usuários no sistema.
Fluxo Alternativo	14.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 14.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

Tabela 30 – UC15 Autenticação do Sistema

Descrição do UC15 – Autenticação do sistema	
Objetivo	Permite que o Administrador faça login no sistema.
Atores	Administrador
Prioridade	Alta Prioridade
Pré-Condições	Permite que o Administrador faça login no sistema.
Pós-Condições	
UC Dependentes	
Descrição	
Fluxo Principal	15.1. Este Caso de Uso se inicia quando o Administrador acessar o sistema a fim de se logar ao mesmo.
Fluxo Alternativo	15.3. Se o login não estiver correto o acesso não poderá ser realizado. 1.2. O Sistema deve apresentar uma mensagem informando que o login e/ou a senha não estão corretos. 15.4. Se o login e a senha estiverem corretos o acesso será realizado com sucesso.

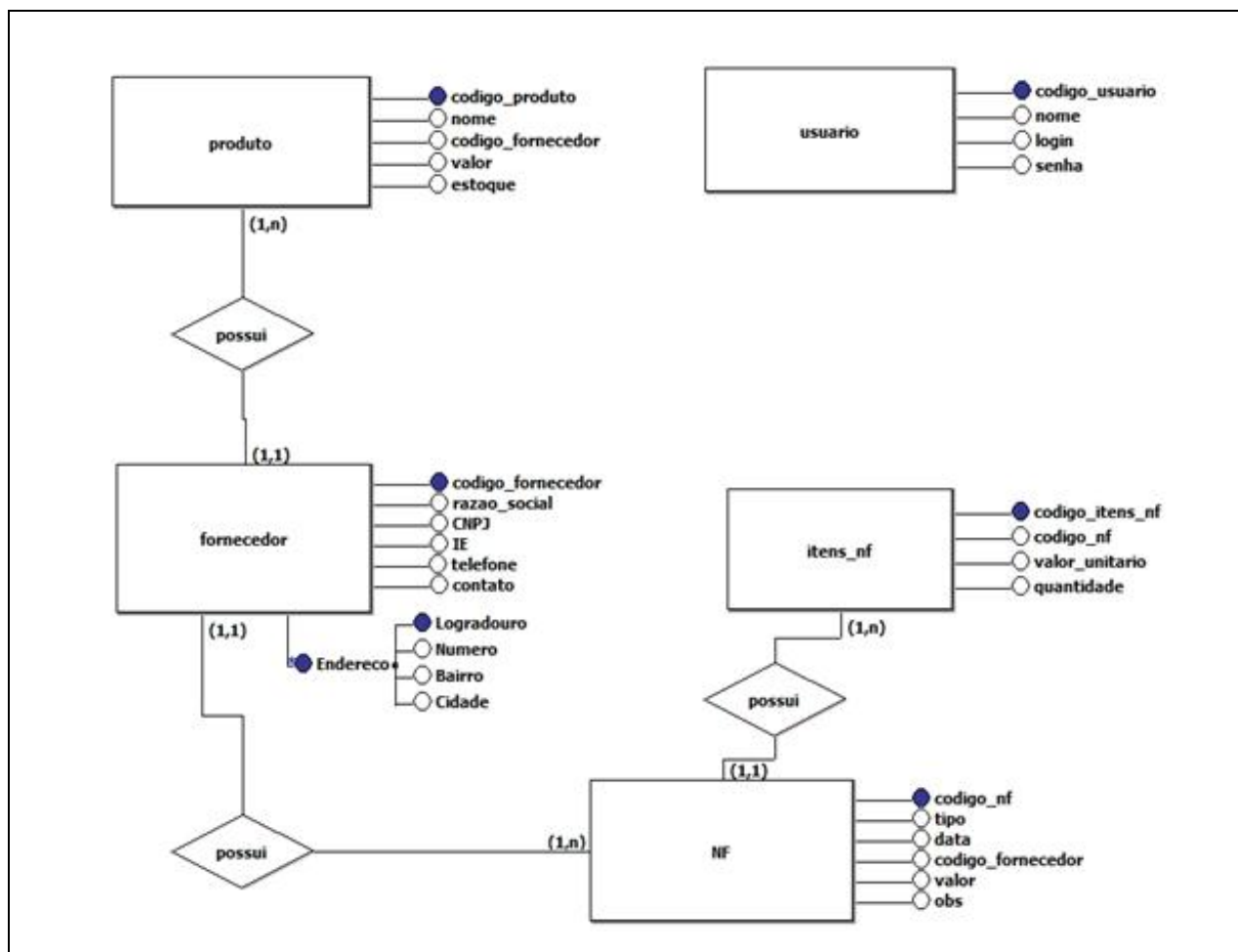
2.4.3 Diagrama ER

O diagrama ER serve para representar de forma gráfica como deverá ser o banco

de dados, nessa análise vamos apresentar dois modelos o modelo Conceitual e o modelo lógico ambos representam a mesma coisa, contudo um deles é mais semelhante às tabelas da base de dados mostrando quais os tipos que serão utilizados nos campos, mostrando com detalhes as chaves estrangeiras e primárias de cada tabela.

Segue abaixo os dois diagramas referentes a base de dados que será utilizada pelo sistema em análise neste projeto.

Figura 11 – DER (Diagrama Entidade Relacionamento) Modelo Conceitual

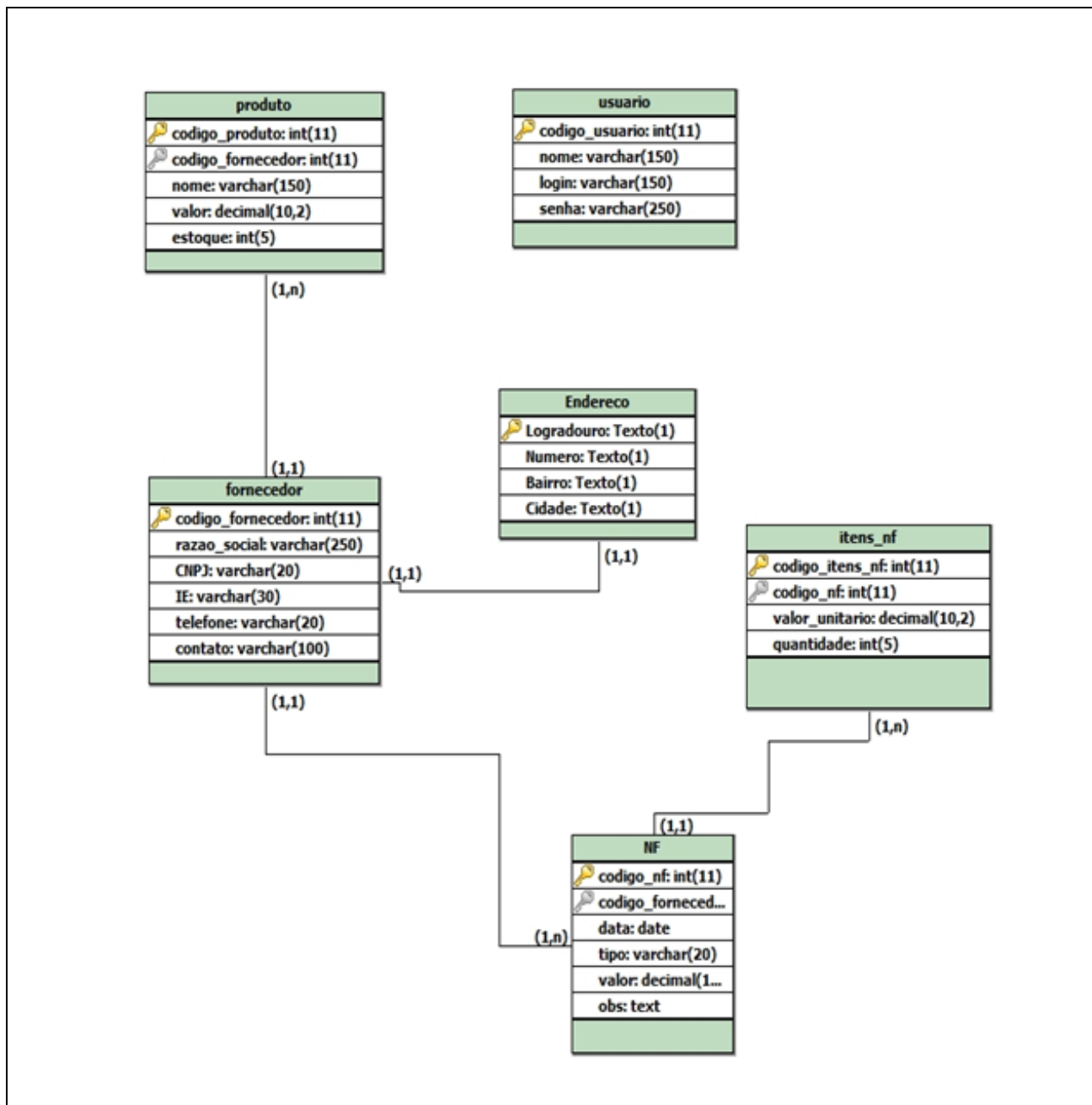


Fonte: Elaborado pelo autor (2019)

Na figura 11 é apresentado o DER modelo conceitual, esse modelo consiste em apresentar de maneira melhor como serão as tabelas e campos e seus relacionamentos, sendo assim o retângulo é usado para representar as tabelas, os círculos pequenos são os atributos e para vínculos entre as tabelas são utilizados losangos que apontam as cardinalidades entre as tabelas.

Nessa figura quase todas as tabelas se ligam como por exemplo: produto e fornecedor, pois não existe um produto sem um fornecedor, contudo a tabela usuário não se liga a ninguém por ser apenas utilizada para validar login, outra característica do modelo conceitual podemos ver no atributo multivalorado endereço da tabela fornecedor esse caso mais tarde irá virar uma tabela de endereço no banco de dados, temos também a chave primária que nesse diagrama é representada pelo círculo cheio enquanto os outros atributos são um círculo vazio.

Figura 12 – DER (Diagrama Entidade Relacionamento) Modelo Lógico



Fonte: Elaborado pelo autor (2019)

Na figura 12 é apresentado o DER lógico, onde as tabelas são apresentadas de

uma forma mais parecida com as tabelas no banco de dados apresentando o domínio de cada atributo (campo) da tabela é um modelo mais refinado.

O Modelo lógico do diagrama ER consiste em mostrar de maneira mais real como ficarão as tabelas no banco de dados real levando em consideração o tipo de dado das colunas, chave primária e chaves estrangeiras de cada tabela e também apresentando as ligações entre as tabelas com suas multiplicidades.

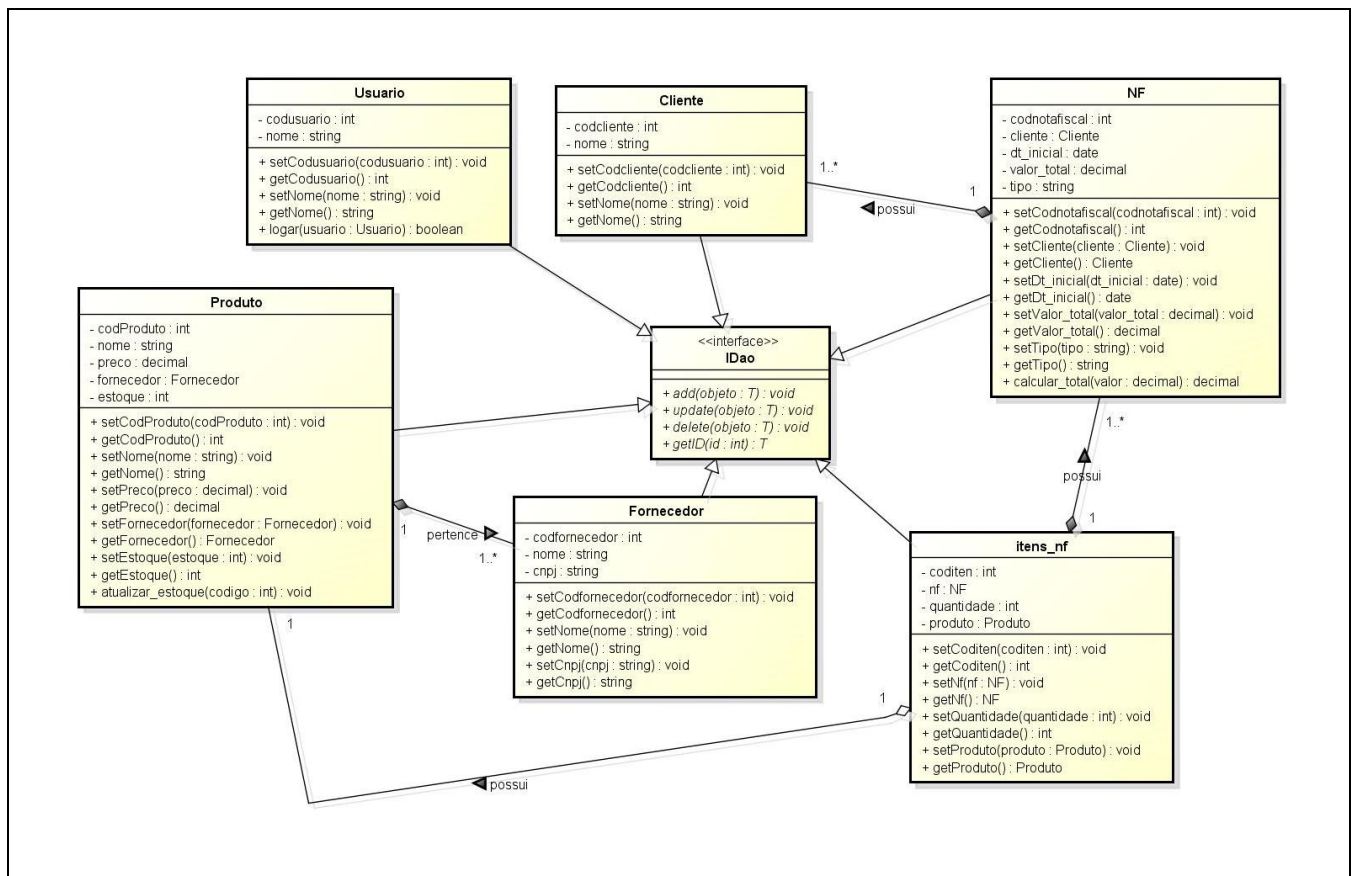
No exemplo da figura acima como falado anteriormente o campo endereço da tabela fornecedor virou uma tabela chamada endereço, após o diagrama ER lógico deve-se criar o script SQL que irá criar as tabelas na base de dados.

2.4.4 Diagrama de Classe

O diagrama de classe como seu próprio nome informa é a representação gráfica das classes que o sistema irá utilizar, para falarmos em classes devemos levar em consideração que o sistema será desenvolvido no padrão O.O (Orientado a Objetos) o que pode levar a utilização de inúmeros recursos entre as classe como por exemplo herança, interfaces, classes abstratas, polimorfismo entre outras funcionalidades que podemos utilizar usando O.O, esse diagrama consiste em mostrar as classes com seus atributos e métodos sendo eles privado, públicos ou protegidos e também as conexões entre as classes, podendo uma ter dependência da outra.

Segue abaixo o diagrama de classes do sistema:

Figura 13 – Diagrama de classe



Fonte: Elaborado pelo autor (2019)

Na figura 13 é apresentado o Diagrama de classe, onde as classes são representadas com seus atributos e métodos que são utilizados no sistema. No modelo apresentado todas as classes implementam uma interface com métodos genéricos comum a todos, e métodos únicos ficam dentro da própria classe. Para que fiquem mais seguros todos os atributos de classe são declarados privados e assim não podem ser acessados externamente, mas tão somente com a utilização dos métodos `get()` e `set()` que tem como função respectivamente pegar e setar o valor do atributo. Está também especificado os relacionamentos entre as classes que podem ser por composição que se designa a que um objeto pode existir sem o outro, e por agregação que significa que

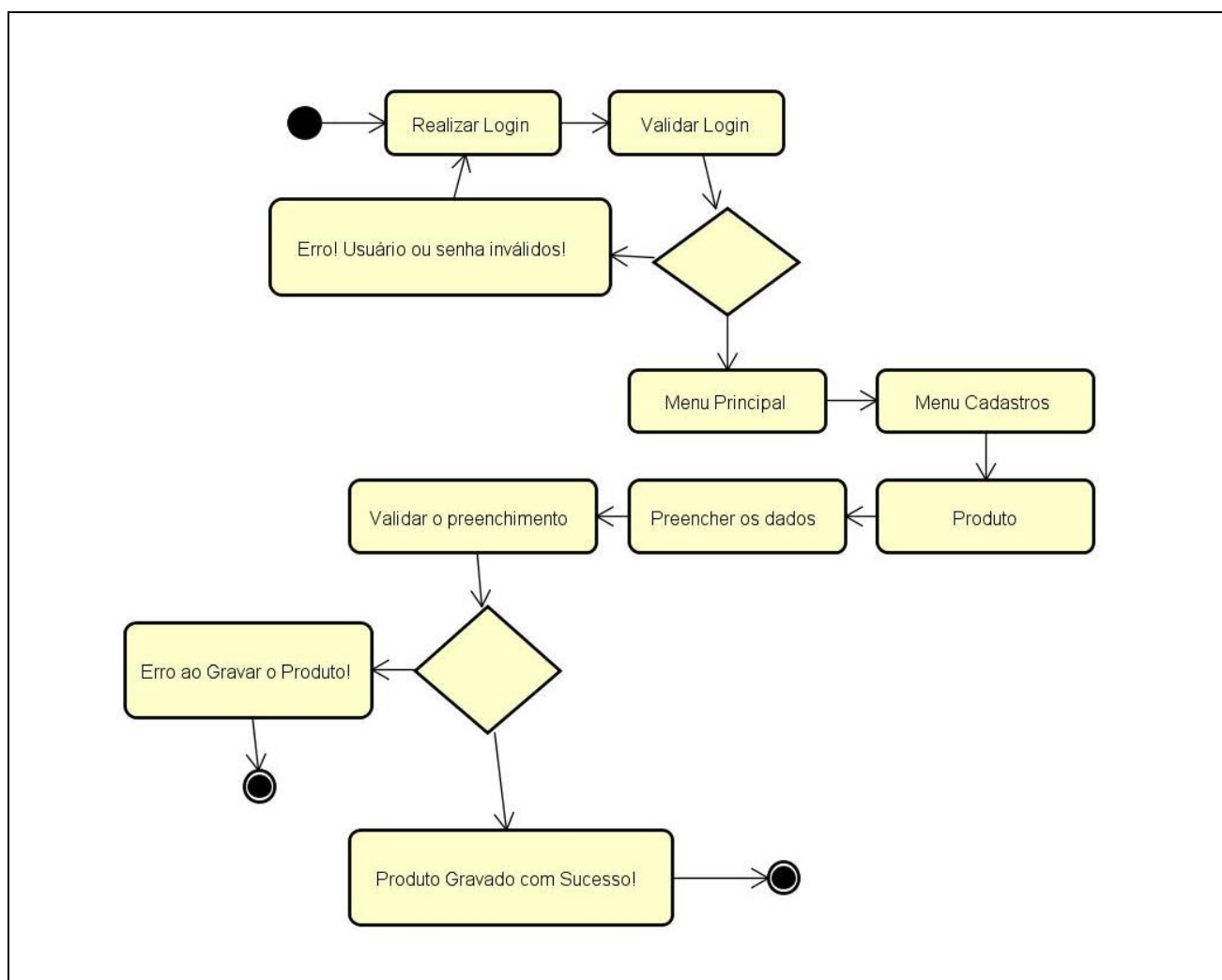
um objeto não pode existir sem o outro.

2.4.5 Diagrama de atividades

O diagrama é a representação gráfica do passo a passo para que uma determinada tarefa seja executada mostrando seu fluxo de controle de uma atividade para a outra, esses fluxos geralmente representam processamentos.

Segue abaixo três diagramas de atividades de tarefas distintas no sistema:

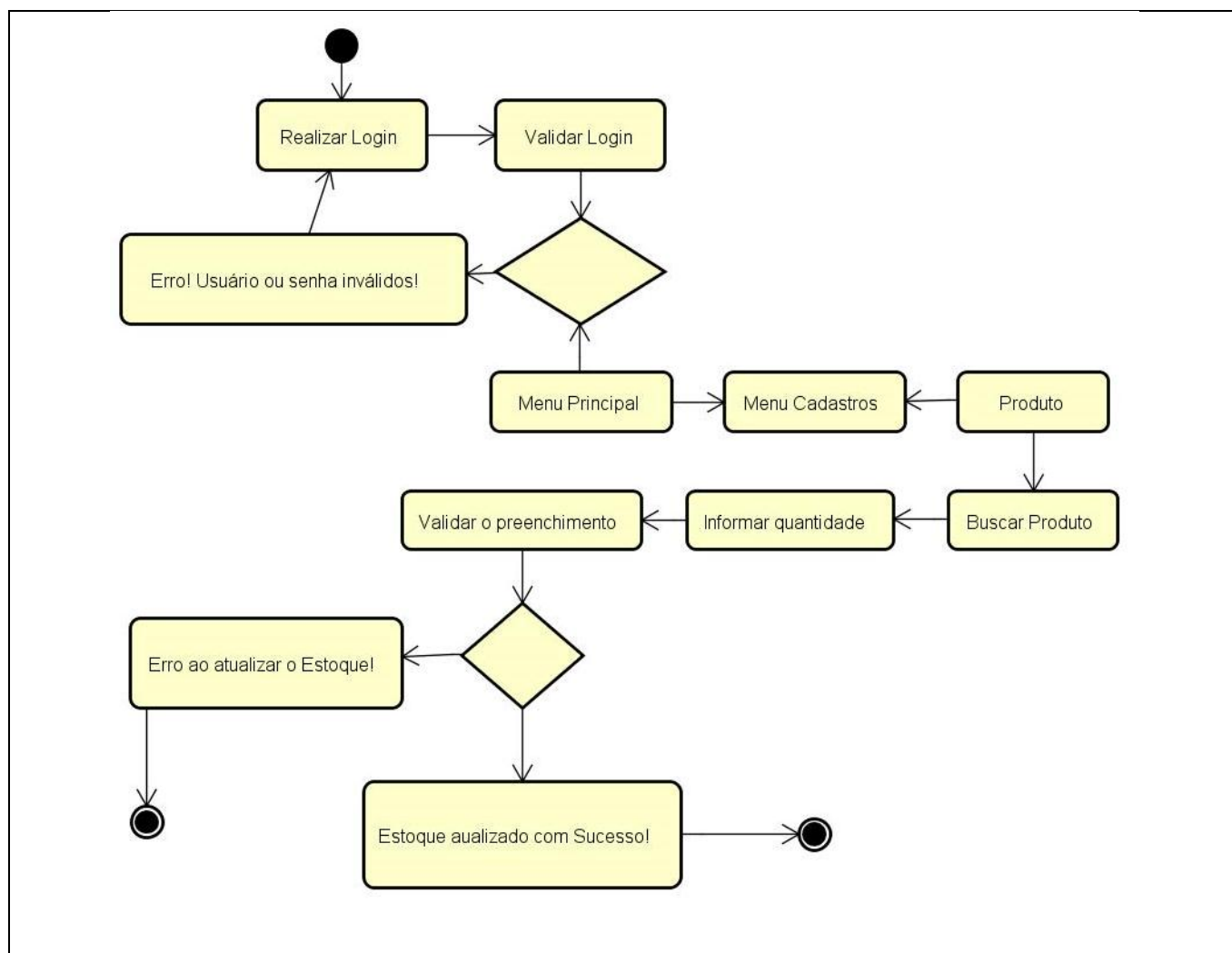
Figura 14 – Diagrama de atividade cadastrar produto



Fonte: Elaborado pelo autor (2019)

Na Figura 14 temos a representação gráfica de um cadastro de produto no sistema, onde temos o início da tarefa que é representado por um círculo preenchido que na vida real será por um usuário, após isso temos os retângulos com bordas arredondadas que representa a atividade que pode ser um processamento ou não. No nosso diagrama para cadastrar um produto deve-se efetuar o login acessar o menu principal opção cadastros, opção produto e preencher os dados do produto, logo após deve clicar em gravar e o sistema irá validar e devolverá uma mensagem, contudo nesse meio tempo temos dois losangos que representam uma tomada de decisão. Caso tudo ocorra de acordo ele continua o processo, caso contrário ele apresenta uma mensagem de erro e finaliza tarefa, ou volta para uma tela anterior como no caso do login.

Figura 15 – Diagrama de atividade Atualização rápida da quantidade de estoque do produto.

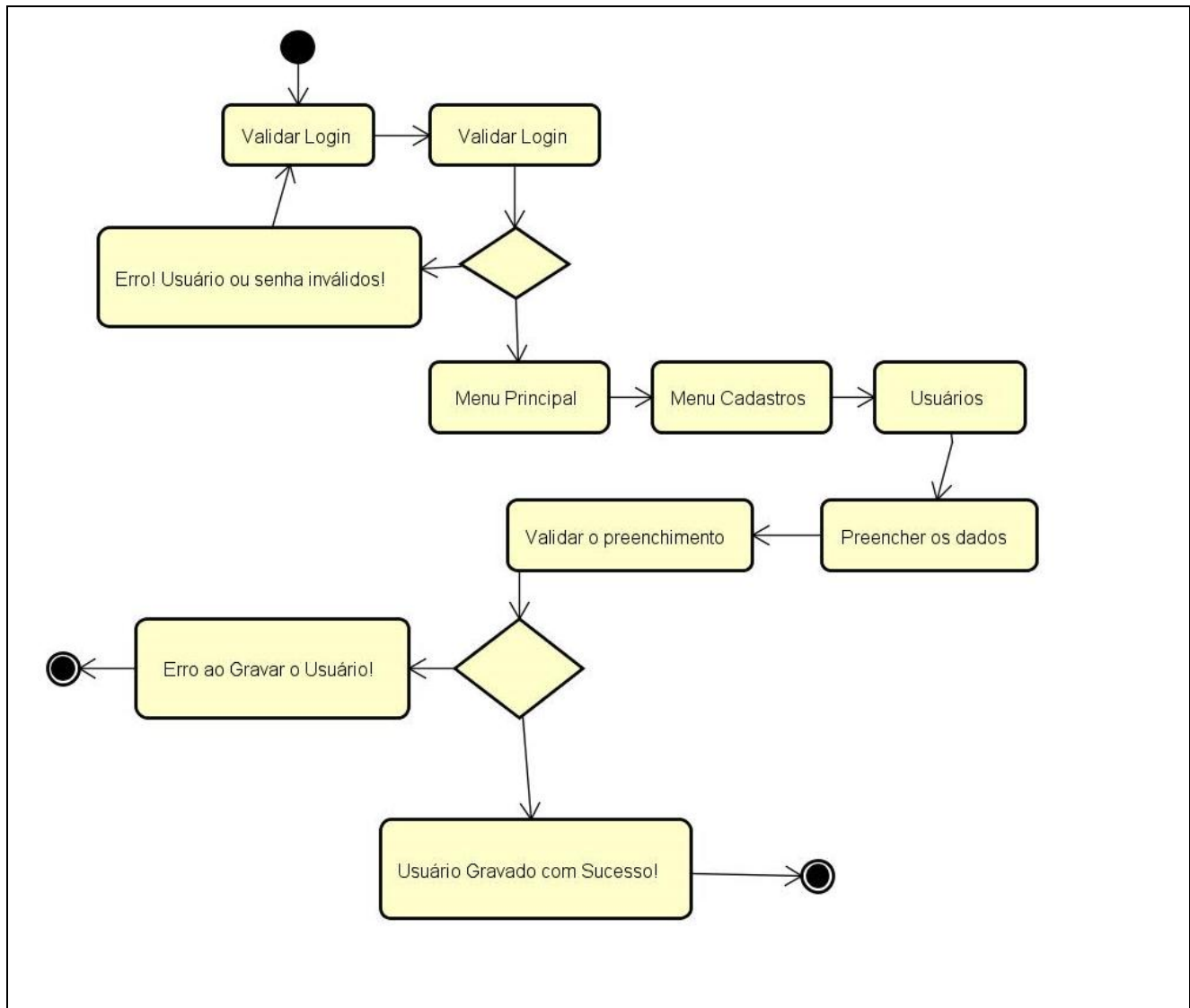


Fonte: Elaborado pelo autor (2019)

Na Figura 15 temos a representação gráfica de uma atualização rápida do estoque do produto, onde novamente devemos efetuar o login para acesso, pois autenticação é muito importante em qualquer sistema, depois temos até o menu cadastro opção produto o mesmo fluxo da atividade anterior a partir desse ponto mudam-se as atividades agora deve-se procurar ou buscar qual produto deseja

atualizar o estoque depois disso deve-se informar a quantidade para o estoque e clicar em gravar e novamente temos uma tomada de decisão que pode resultar em uma mensagem de erro ou de sucesso ambas após isso finalizam a tarefa.

Figura 16 – Diagrama de atividade Cadastrar Usuário.



Fonte: Elaborado pelo autor (2019)

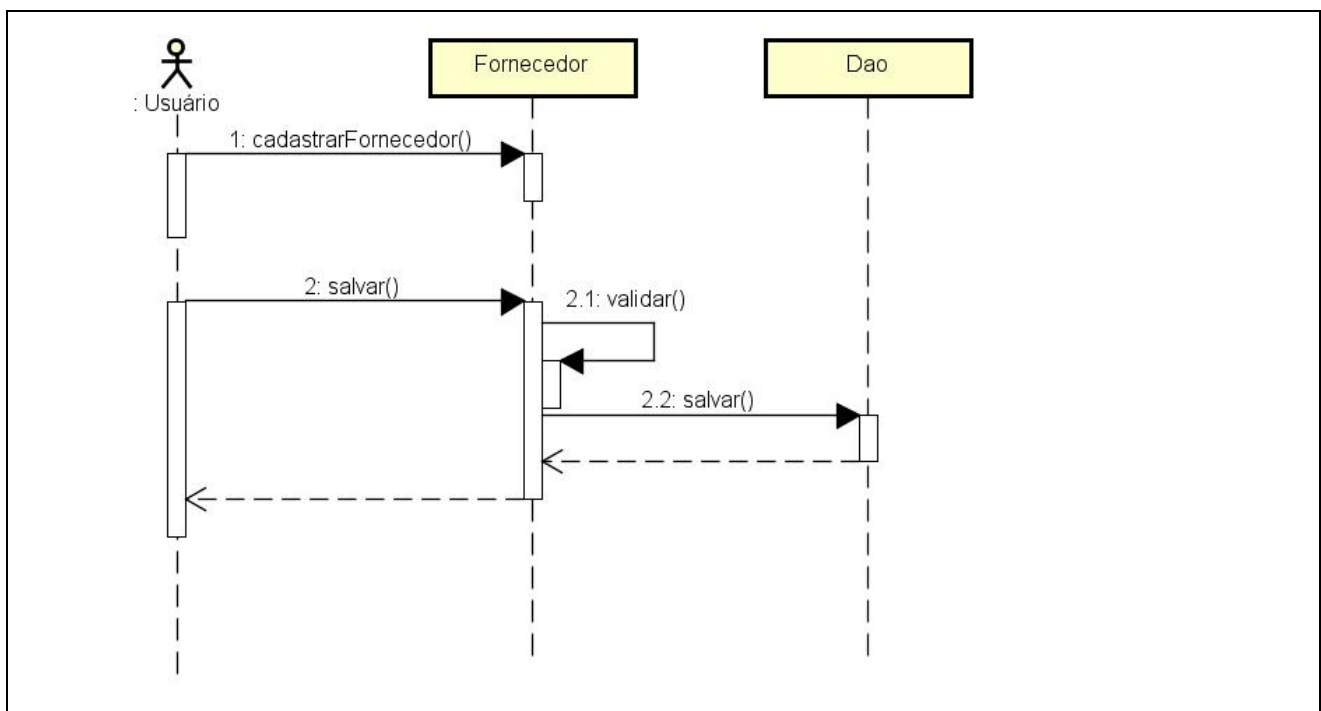
Na Figura 16 temos a representação gráfica de um cadastro de usuário cujo

fluxo é basicamente o mesmo do cadastrar produto, contudo ao invés de utilizar ou menu cadastro opção produto irá selecionar a opção usuário e após isso o restante do fluxo segue a mesma ordem da tarefa cadastrar produto.

2.4.6 Diagrama de sequência

Os diagramas de sequência representam a troca de mensagens entre as classes que serão utilizadas para execução de determinada tarefa, segue abaixo a representação de troca de mensagens de três tarefas que poderão ser realizadas no sistema proposto:

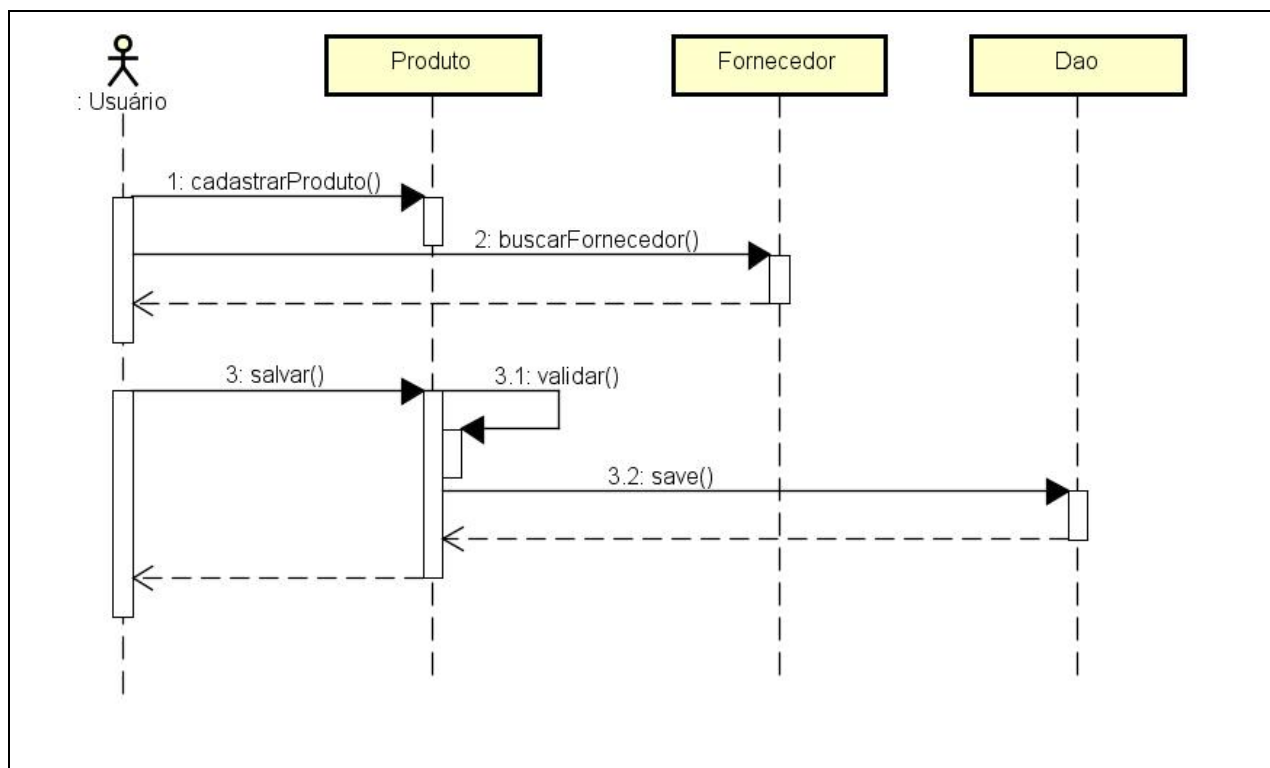
Figura 17 – Diagrama de Sequência Cadastrar Fornecedor.



Fonte: Elaborado pelo autor (2019)

Na Figura 17 temos o diagrama de sequência que representa o cadastro de um fornecedor, onde temos o ator que é representado pelo usuário e esse mesmo dispara uma ação para cadastrar o fornecedor, isso envolve a classe fornecedor e a interface Dao que por fim irá persistir na base de dados e irá devolver uma mensagem ao usuário.

Figura 18 – Diagrama de Sequência Cadastrar Produto.

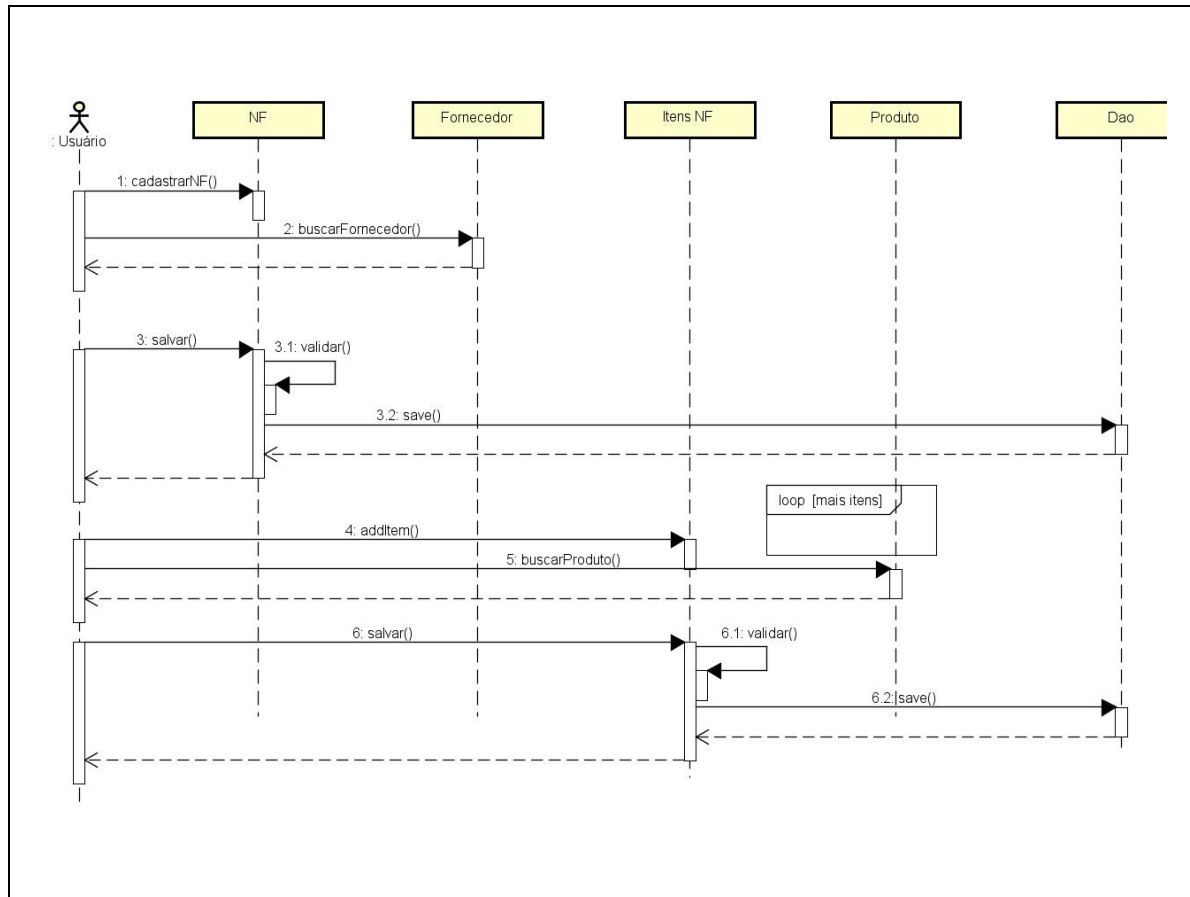


Fonte: Elaborado pelo autor (2019)

Na Figura 18 temos o diagrama de sequência de um cadastro de produto que não é muito diferente do diagrama anterior, contudo nesse temos mais uma classe envolvida além da classe produto que é a principal, a classe fornecedor que compõe parte da classe produto, sendo necessário criar uma função para buscar o fornecedor

para o produto. No mais é a o mesmo processo até persistir na base de dados e retornar uma mensagem para o cliente.

Figura 19 – Diagrama de Sequência Cadastrar NF.



Fonte: Elaborado pelo autor (2019)

Na Figura 19 temos o diagrama de sequência de um cadastro de Nota Fiscal cuja classe que a representa é a NF, esse processo é mais complexo pelo fato de uma nota fiscal possuir itens que pertencem a essa nota, sendo assim temos a representação do cadastro da nota fiscal e logo após de maneira repetitiva o cadastro dos itens da nota fiscal que no gráfico é representado pelo loop, e por fim todos persistem na base de dados e retornam uma mensagem para o usuário.

4.4 Prototipação

Na prototipação irei apresentar algumas telas que o sistema terá e também fazer a descrição de cada tela para entender a função de cada uma, segue abaixo algumas telas do sistema de controle de estoque que será construído e que por enquanto é somente uma análise.

Figura 20 – Tela de login.



O protótipo da tela de login apresenta o seguinte layout:

- Logo do **CONSTRUCENTER** com o slogan "Construindo os seus sonhos!"
- Ícone de uma fechadura e o texto **Login**.
- Formulário com fundo vermelho contendo:
 - Campos de entrada para **Usuário:** (com o placeholder "Digite seu usuário") e **Senha:** (com o placeholder "Digite sua senha").
 - Botões **Resetar** e **Logar**.
 - Link [Esqueci minha senha](#).

Fonte: Elaborado pelo autor (2019)

Na Figura 20 temos a tela login do sistema, essa tela serve para efetuar a validação de um usuário e senha não permitindo que alguém que não tenha permissão

acesse o sistema de controle de estoque.


Figura 21 – Tela Inicial do Sistema.



Fonte: Elaborado pelo autor (2019)

Na Figura 21 temos a tela inicial que vem logo após ter efetuado login no sistema, essa tela apresenta os menus de acesso aos cadastros, notas fiscais e de relatórios no sistema além dos menus de acesso rápido às funcionalidades do sistema.

Figura 22 – Tela de cadastro de fornecedor.



Cadastros | Notas Fiscais | Relatórios

sergio ▼

Cadastro de Fornecedor

Código:

Razão Social: CNPJ: IE:

Telefone: Contato:

Resetar

Adicionar Endereço

Salvar

Código	Razão Social	CNPJ	Telefone	Contato	Ações
1	Coral LTDA	99999999999	5133140000	Márcio Silveira	<div>Alterar</div> <div>Remover</div>

☐ Razão Social

☐ CNPJ

Pesquisar

Sistema desenvolvido por Sérgio Henrique

Fonte: Elaborado pelo autor (2019)

Na Figura 22 temos a tela de cadastro de fornecedor onde se pode efetuar cadastro de fornecedor e de vários endereços para o fornecedor, permite também alterar, excluir e pesquisar tanto pela razão social ou pelo CNPJ.


Figura 23 – Tela de cadastro de produto.

CONSTRUCCENTER
Construindo os seus sonhos

[Cadastros](#) | [Notas Fiscais](#) | [Relatórios](#) sergio ▼

Cadastro do Produto

Código:

Nome: Fornecedor:  Valor:

Estoque:

Código	Nome	Valor	Estoque	Ações
10	Parafuso	R\$ 1,50	1000	<input type="button" value="Alterar"/> <input type="button" value="Remover"/> <input checked="" type="button" value="Estoque"/>

☐ Nome ☐ Fornecedor

Sistema desenvolvido por Sérgio Henrique

Fonte: Elaborado pelo autor (2019)

Na Figura 23 temos a tela de cadastro de produtos onde estão as funcionalidades de inclusão, alterar, exclusão e pesquisa de produto, contudo aqui temos uma opção diferente que é a atualização rápida de estoque e também a opção de busca de fornecedor representada pela lupa do lado do campo. Na tabela terá um botão editar que ao ser clicado acionará no banco de dados e buscará os campos para editar com base no ID.

O usuário do sistema solicita o serviço desejado, o sistema busca a informação no SGBD e imediatamente a fornece. No caso de alteração do item ou cadastro o sistema configura o que o Hibernate deve fazer com o banco de dados ao ser iniciado.

Nesses casos o Hibernate irá checar por alterações entre o mapeamento e o banco de dados, como uma propriedade adicionada em uma classe. Quando é encontrada alguma alteração então o Hibernate executa o DDL para atualizar o banco de dados.

No caso de exclusão do item podemos ter uma classe com um atributo que indica se o registro está ativo ou não, e então modificamos a nossa lógica de exclusão para apenas atualizar o registro. Esta é uma solução simples e comum para implementar uma funcionalidade de exclusão lógica, mas alguns recursos do Hibernate permitem uma abordagem muito interessante. Podemos alterar como o Hibernate trabalhar com a exclusão de registros, através da anotação `@SQLDelete`

5 CONSIDERAÇÕES FINAIS

Os resultados obtidos no projeto QIMK – *Quick inventory Management Kit* demonstram uma das formas de conseguir uma solução de gerenciamento dos dados, correspondendo as integrações propostas, logo os resultados foram alcançados, vislumbrando o seu escopo.

Conseguiu-se desenvolver uma solução com níveis iniciais de alta coesão e um baixo acoplamento, sendo sustentado com a utilização de bibliotecas externas que auxiliam no desenvolvimento da aplicação.

Como trabalhos futuros, ficam como sugestão de desenvolvimento do projeto na íntegra e também a utilização das soluções de Web Site para que o projeto possa adequar-se às tendências nas soluções empresariais encontradas no mercado de softwares, outra questão, visando o futuro da aplicação é utilização de WS na importação e exportação dos dados.

REFERÊNCIAS

- BAUER, C.; KING, G. **Java PersistenceWithHibernate**. [s.l.] Manning, 2006.
- BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML Guia do Usuário**. 2. ed. [s.l.] Campus, 2006.
- CORRÊA, A. G. D. **Programação I**. São Paulo: Person, 2016.
- COSTA, W. S.; SILVA, S. C. M. Aquisição De Conhecimento: O Grande Desafio Na Concepção De Sistemas Especialistas. **Holos**, v. 2, p. 37, 2007.
- DEITEL, P.; DEITAL, H. **Java como programar**. 8. ed. São Paulo: Person, 2010.
- DEITEL, P.; DEITAL, H. **Java como programar**. 10. ed. São Paulo: Person, 2016.
- EIS, D. **Uma breve história do CSS - Artigos sobre HTML, JavaScript, CSS e desenvolvimento web**. Disponível em: <<https://tableless.com.br/uma-breve-historia-do-css/>>. Acesso em: 15 mai. 2019.
- ELMASRI, R.; NAVATHE, S. **Sistemas de Banco de Dados**. [s.l.] Person, 2005.
- FARIA, T.; JUNIOR, N. **JPA e Hibernate**. 1. ed. [s.l.: s.n.].
- FRAGOSO, R. R. **O que é Hibernate**. Disponível em: <http://www.dicas-l.com.br/arquivo/o_que_e_hibernate.php#.W5rveKZKjIV>. Acesso em: 15 mai. 2019.
- GERHARDT, T. E.; SILVEIRA, D. T. **Métodos de Pesquisa**. [s.l.] UFRGS, 2009.
- GUGIK, G. **A história dos computadores e da computação - TecMundo**. Disponível em: <<https://www.tecmundo.com.br/tecnologia-da-informacao/1697-a-historia-dos-computadores-e-da-computacao.htm>>. Acesso em: 15 mai. 2019.

HOFFMAN, C. **What Is Open Source Software, and Why Does It Matter?**

Disponível em: <<https://www.howtogeek.com/129967/htg-explains-what-is-open-source-software-and-why-you-should-care/>>. Acesso em: 15 mai. 2019.

HORSTMANN, C.; CORNELL, G. **Core Java - Volume 1.8.** ed. São Paulo: Person, 2015.

LEE, R.; TEPFENHART, W. **UML e C++ Guia Prático de Desenvolvimento Orientado a Objeto.** [s.l.] Makron, 2001.

LEITÃO, A. M. **Linguagem de Programação.** Disponível em: <<http://www.dca.fee.unicamp.br/courses/EA072/lisp9596/node2.html>>. Acesso em: 15 mai. 2019.

MARX, D. **JavaOne 2011: Introduction to the JavaFX Scene Builder | JavaWorld.** Disponível em: <<https://www.javaworld.com/article/2074393/core-java/javaone-2011--introduction-to-the-javafx-scene-builder.html>>. Acesso em: 31 mai. 2019.

MEDEIROS, L. F. **Banco de Dados princípios e prática.** [s.l.]Intersaberes, 2013.

MERCER, J. **A Short History of Java - DZone Java.**Disponível em:

NASKAR, V. **Whatis IDE?** Disponível em: <<https://www.quora.com/What-is->

OLIVEIRA, M. V. **Modelagem de Sistemas - Aula 1.** Disponível em: <https://www.youtube.com/watch?v=rjXFXJGfC_8>. Acesso em: 31 mai. 2019.

OMG.**ABOUT THE UNIFIED MODELING LANGUAGE SPECIFICATION VERSION 2.5.1.** Disponível em: <<https://www.omg.org/spec/UML>>. Acesso em: 31 mai. 2019.

ORACLE.**Oracle and Sun Microsystems | Strategic Acquisitions | Oracle Brasil.** Disponível em: <<https://www.oracle.com/br/sun/index.html>>. Acesso em: 31 mai.

2019.

PAWLAN, M. **What Is JavaFX? | JavaFX 2 Tutorials and Documentation.**

Disponível em: <<https://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm#>>.

Acesso em: 31 mai. 2019.

PEREIRA, A. P. **O que é XML? - TecMundo.** Disponível em:

<<https://www.tecmundo.com.br/programacao/1762-o-que-e-xml-.htm>>. Acesso em: 31

mai. 2019.

PIRES, E. **Orientação a Objeto - SOLID - Eduardo Pires.** Disponível em:

<<http://www.eduardopires.net.br/2013/04/orientacao-a-objeto-solid/>>. Acesso em: 31

mai. 2019.

SOMMERVILLE, I. **Engenharia de Software.** 9. ed. São Paulo: Person, 2011.

SRIVASTAVA, P. **JavaFX: A Rich Internet Application (RIA) Development Platform.** Disponível em: <<https://opensourceforu.com/2016/08/javafx-rich-internet-application-ria-development-platform/>>. Acesso em: 15 mai. 2019.

VAID, V. **Using Scene Builder to develop JavaFX apps - Open Source For You.** Disponível em: <<https://opensourceforu.com/2017/09/develop-javafx-apps/>>.

Acesso em: 15 mai. 2019.