A Importância da Tecnologia no Gerenciamento das Vendas e o Desenvolvimento do Protótipo SOFTVAV

Voncarlos Marcelo de Araújo[1], Regiane Orlovski[2]

[1] [2]Tecnologia em Análise e Desenvolvimento de Sistemas – Faculdade Guairacá

Rua XV de Novembro, 7050 – Centro -85010-000 – Guarapuava – PR – Brasil.

[1] voncarlos13a@hotmail.com, [2]regianeorlovski@hotmail.com

Abstract: *This article aims to present the importance of information technology in sales management and development of a software able to help companies that need to manage their activities in a flexible and helpful way, generating future profits contributing significantly to those the work in commerce. This importance is evident by presenting a software prototype that makes use of this technology. It was developed using the prototyping life cycle for the results obtained attend the customer expectations.*

Resumo: *Este artigo tem por objetivo apresentar a importância da Tecnologia da Informação no gerenciamento de vendas e o desenvolvimento de um software capaz de administrar e auxiliar as empresas que necessitam realizar suas atividades de forma ágil e prestativa, gerando lucros futuros contribuindo significativamente para quem atua no comércio. Essa importância fica evidente apresentando um protótipo de Software que faz uso dessa Tecnologia. Foi desenvolvido utilizando o ciclo de vida chamado prototipação para que os resultados obtidos suprissem as expectativas do cliente.*

Introdução

Conforme a expansão no mercado de vendas de produtos surgiu à necessidade de um controle maior das informações e a automatização desse processo, desta forma foi desenvolvido um software de gerenciamento de vendas capaz de suprir as necessidades do mercado, já existem muitos sistemas de gerenciamento e controle, mas o diferencial que esse software tem a apresentar é conduzir o usuário a fazer uma venda de forma simples, tranquila e sem preocupações a falhas, melhorando o controle das vendas e, consequentemente aumentando os lucros, aonde não seja necessário treinamento para utiliza-lo, pois integra com um sistema de autoajuda para o usuário que esta interagindo não se perca.

A implementação é realizada empregando a linguagem Java em um sistema para *desktop* utilizando o framework JPA - *Java Persistente API (Application Programming Interface)*, para o gerenciamento do banco de dados foi utilizado o SGBD MYSQL.

Hoje o software em uma empresa é entendido como uma forma de agregar valores e gerar lucros para seus negócios, e pensando nisso para desenvolver o software SOFTVAV (Software Voncarlos Araujo Vendas) com qualidade foi adotado um

processo de desenvolvimento chamado XP (*Extreme Programming*), divididos em varias etapas as quais se as atividades trabalharem de forma correta e de acordo com os padrões requeridos, o produto final desejado é realizado.

Desenvolvimento

Fundamentação Teórica

Com o avanço da Tecnologia de Informação (TI), as empresas tornaram-se cada vez mais dependentes de sistemas informativos, que auxiliam nas operações diárias em seus negócios. O comércio tem sido um dos orgulhos do Brasil nos últimos anos, com a inclusão de novos consumidores na classe média e o aumento da renda familiar é motivo para muitos economistas apontarem o País como uma potência promissora, Flávio Silva (2011). Mas nada disso ocorreria se as empresas de atacado e varejo sustentassem seus negócios no modelo papel e caneta, resultando em lenta transferência de informações, sendo a mesma pouco confiável e propensa a erros. Pensando nisso, estes setores sabem da importância e a necessidade de um controle de informações e automatização para inovar os seus negócios.

O custo da TI para as empresas é associado a sua maior facilidade de uso, permite aos usuários que trabalham no comércio usar essa tecnologia, com o objetivo de transferir e gerenciar informações manualmente, porém, com maior eficiência, qualidade e rapidez que proporcionam oportunidades de redução de despesas por meio de melhor coordenação do que está acontecendo, além do aperfeiçoamento dos serviços que podem ser vistos com cada uma das informações dos clientes, contendo um relatório do que ele mais compra, se ele é ativo na loja, etc. São informações básicas que fazem a diferença no mercado na hora de interagir com o seu publico alvo.

Torna-se de extrema necessidade para organizações a missão de administrar as informações, pois existe uma crescente demanda e sofisticação na TI de software e hardware, este recurso será de vital importância para sobrevivência das empresas Dalfovo e Amorim (2000). Desta forma, os sistemas de informação tem um papel fundamental e cada vez maior em todas as organizações de negócios, com o intuito de agilizar, aperfeiçoar e obter um controle maior sobre as operações.

Sistema de informação é um conjunto de elementos ou componentes inter-relacionados que coletam, manipulam e disseminam dados e informações, e fornecem um mecanismo de feedback para atingir um objetivo, Stair (1998). Conforme a necessidade do usuário é importante que toda a interação do sistema com o usuário, sejam apresentadas em mensagens para atingir suas metas e superar suas possíveis expectativas perante o sistema, além de ajudar o usuário a aprender melhor como usar o sistema.

Juntamente com a TI a ideia central é a implementação de um sistema de gerenciamento de vendas simples e que resolva o problema diário de uma empresa deficiente de realizar uma venda controlada, mas isso tudo com ferramentas disponíveis no mercado e principalmente gratuitas, para que a realização do projeto não tenha custo adicional.

Para Bruin (2000), um exemplo é o e-mail, que surgiu como simples ferramenta destinada à troca de mensagens e vem se revelando um eficiente canal de vendas, ao assumir funções de mala direta, tal utilização apresenta vantagens atraentes, pois além

de eliminar custos com papel, impressão e postagem, possibilita mensagens que facilitam ligações do receptor às páginas eletrônicas da empresa, através de links.

Desta forma, no ambiente das organizações a TI passa a desempenhar papel estratégico, onde os Sistemas de Informação possam alavancar dados, transformando-os em ativos estratégicos de negócios.

Na maioria das empresas, a adoção da TI surge em função de uma necessidade derivada dos objetivos organizacionais preestabelecidos. Assim, um dos itens avaliados refere-se à identificação dos motivos que levaram as empresas a implantar a TI. Os motivos considerados mais importantes são apresentados abaixo na Tabela 1

Tabela 1: Motivos para Implantação de TI

Motivos para implantação de TI	%
Necessidade de integração	2
Melhoria de controles organizacionais	24
Competitividade	8
Manter sua participação	5
Aumentar sua participação	22
Melhoria da qualidade de atendimento	8
Aumentar a produtividade	20
Gerar um ambiente criativo	1
Reduzir custos	10

Fonte: Prates e Ospina (2004).

Com a evolução da TI e a busca por adquirir melhoria de processos e qualidade, as empresas necessitam desenvolver cada vez mais técnicas na produção e também no gerenciamento financeiro de sua propriedade. De acordo com Yong (1992), nos países do primeiro mundo a TI tem sido considerada como um dos fatores responsáveis pelo sucesso das organizações, tanto no âmbito de sobrevivência, quanto no aumento da competitividade. Corroborando neste pensamento, Zuboff (1994), afirma que a TI, baseada nos computadores, está proporcionando nova infraestrutura para as várias atividades produtivas e comunicativas, algo vital para a vida organizacional.

Em função disso foi visto que a necessidade de uma empresa se render as características da TI se torna uma dependência, pois ela traz benefícios que fazem a empresa se destacar das demais, por motivos de eficiência, qualidade de trabalho, exatidão das informações e segurança, por sua vez empresas que já atuam com a TI ficam totalmente dependente das suas informações, mas deve ser encarada como uma ferramenta auxiliar para os negócios. Em determinadas circunstâncias, ela passa a ser caracterizada como essencial, para que as atividades-base das empresas consigam ser realizadas em menos tempo possível, com menor custo e com o menor número de erros.

A TI é muito importante para que as diversas empresas envolvidas em uma cadeia de suprimentos consigam ter acesso às informações certas, no momento mais apropriado. Aplicar a tecnologia adequada a diferentes informações às quais o

comerciante tem acesso é fundamental para aumentar suas possibilidades na geração de lucro.

Conforme Nielsen (2000), para aplicar esse conceito da TI nas empresas é necessário o usuário adaptar-se e interagir com o software, com base nisso foram usados os princípios ergonômicos em interfaces, fácil aprendizagem, flexibilidade e atitude. Essas características fazem com que o sistema alcance níveis de desempenho aceitável dentro de um tempo especificado, que deve ser atingido considerando custos humanos aceitáveis, em termos de fadiga, stress, frustração, satisfação e desconforto na utilização do sistema, buscando sempre o conforto e de reação favorável do operador no que se refere ao uso do sistema, sendo este, talvez o aspecto da usabilidade mais difícil de medir e quantificar, devido aos seus fatores. Ainda para Nielsen (2000), se tratando de princípios ergonômicos a usabilidade é a parte dos objetivos e da metodologia ergonômica de adequações das interfaces tecnológicas as características e capacidades humanas físicas, cognitivas e emocionais.

Continua Nielsen (2000), podemos dizer de uma forma simples que a usabilidade é a preocupação que os desenvolvedores devem ter com o bem estar do usuário ao desempenharem quaisquer tarefas no produto desenvolvido. Por fim, destaca-se a importância de considerar, em um processo de avaliação ergonômica, a satisfação do usuário com relação ao software, definida como uma avaliação critica do equipamento de acordo com as expectativas do usuário, e ilustra o nível de sucesso ou insucesso do software em suprir as necessidades de determinada atividade do operador.

Para alcançar os critérios ergonômicos de acordo com o que foi proposto é necessário para o desenvolvimento do software um ciclo de vida que representa as diversas etapas pelas quais passa um projeto de desenvolvimento. Em sua forma tradicional o ciclo de vida inclui as etapas de levantamento de requisitos, definição de escopo do projeto, analise de alternativas, projeto do sistema, codificação, testes, conversão de dados e manutenção.
Para entender esse objetivo, a metodologia utilizada fundamentou-se no ciclo de vida de desenvolvimento de software, baseando-se no conceito de prototipação, aonde sucessivas repetições de todas as etapas vão refinando incrementalmente o produto final até que esteja pronto para ser efetivamente implementado, uma vez que propicia ao desenvolvedor criar um modelo de software que, posteriormente, será avaliado pelo cliente e, então, implementado. Essa abordagem tem seu inicio na coleta e refinamento dos requisitos, avaliação pelo cliente e finalmente, a construção do produto.

Segundo Pressman (1995), é dever observar três conceitos fundamentais na prototipação.

- Protótipo - É um conjunto de rotinas e programas (modelo) de software a ser desenvolvido e construído para avaliação do desenvolvedor e do cliente;
- Modelo - É o protótipo em fase de testes e refinamento;
- Pacote - É o modelo já depurado em fase de produção, funcionando na empresa do cliente.

A noção de ciclo de vida também incorpora a ideia de que sistemas passam por fases sucessivas de crescimento, evolução e declínio, e que ao final deste ciclo devem

ser substituídos por outros sistemas que possam melhor entender as necessidades das empresas, Lucas (1985).

Após o intuito de focar os princípios ergonômicos e ciclo de vida utilizado, necessita efetuar a técnica de desenvolvimento usada, pois quando se aplica um padrão de desenvolvimento tem uma visão do todo, economizando linhas de código, além de deixar o cliente satisfeito em aspectos como custos e horas que vão favorecer ambos os lados.

Para definir de forma clara o que, quando, como e até mesmo aonde aplicar o software foi definida uma metodologia de desenvolvimento *Programming* (XP) é uma metodologia ágil para equipes pequenas que desenvolvem software baseado em requisitos, aonde seu principal objetivo é dar agilidade ao desenvolvimento do projeto e buscar garantir a satisfação do cliente, o desenvolvimento é incremental (ou iterativo), onde o sistema começa a ser implementado logo no início do projeto e vai ganhando novas funcionalidades ao longo do tempo Malhães e Beck(2005).

As praticas, regras e os valores da XP garantem um agradável ambiente de desenvolvimento que são conduzidos por três princípios como a comunicação, simplicidade e feedback. A comunicação busca manter o melhor relacionamento entre clientes e desenvolvedores, preferindo conversas pessoais. A simplicidade tem como objetivo implementar apenas requisitos atuais, evitando assim adicionar funcionalidades que podem ser importantes apenas no futuro. E o principio do feedback significa que as informações do código é dada pelos testes constantes , que indicam os erros tanto individuais quanto o software integrado. Na concepção de Malhães e Beck (2005), essa metodologia se aplica no software, pois define o que é ou não necessário ser feito no projeto conforme os requisitos aparecem e devem satisfazer os requisitos atuais, aonde os requisitos futuros devem ser adicionados assim que eles realmente existirem.

Para o desenvolvimento do software proposto neste trabalho foi utilizada Orientação a Objetos (OO), pela agilidade no desenvolvimento e características apresentadas abaixo.

Programação orientada a objetos é uma tecnologia chave para a implementação de sistema complexos, provendo benefícios e reusabilidade, consistência e manutenibilidade e as bases de dados relacionais são repositórios para dados persistentes Russell (2008).

O avanço das tecnologias na área de análise, projeto, desenvolvimento e qualidade de software permite que aplicações mais flexíveis e poderosas sejam construídas, a necessidade de interagir com estas aplicações criou uma nova metodologia de analise e desenvolvimento: a OO que traz muitas vantagens que motivam a quem não usa essa técnica a se readaptar, uma delas é a redução no custo de manutenção do software, que permite que, quando for necessária alguma alteração, modifique-se apenas o objeto que necessita desta alteração, e ela propaga-se automaticamente as demais partes do software que utilizam este objeto, e o aumento na reutilização do código, é uma maneira de programar que ajuda na organização e resolve muitos problemas, além de concentrar as responsabilidades nos pontos certos, flexibilizando sua aplicação.

Segundo Dall`oglio (2007), não é possível desenvolver nenhum programa sem seguir tal paradigma. Um sistema OO é composto de classes e objetos bem definidos

que interagem entre si, de modo a gerar o resultado esperado. Ainda para Dall`oglio (2007), no que diz respeito ao desempenho, ao compromisso, ao esforço, á dedicação, não existe meio termo. Ou você faz uma coisa bem feita ou não faz.

Para o desenvolvimento OO foi optado pela linguagem Java, ela oferece diversas funcionalidades com relação à segurança, também se torna altamente produtiva devido à orientação a objetos. O código dessa linguagem é organizado em classes, que podem estabelecer relacionamentos de herança simples entre si, adequada para o desenvolvimento de aplicações baseadas na rede Internet, redes fechadas ou ainda programas *stand-alone* Campione (1996). Ainda para Campione (1996), É uma linguagem simples, de fácil aprendizado, pois possui possibilidades de erros de programação em Java reduzidas, além disso, a linguagem traz outros recursos para tornar seu código, tais como ponteiros e gerenciamento de memória via código de programação também faz com que a programação em Java seja mais eficiente, contém um conjunto de bibliotecas que fornecem grande parte da funcionalidade básica da linguagem, incluindo rotinas de acesso à rede e criação de interface gráfica.

A linguagem Java tem varias ferramentas dentre elas se destaca as IDE's (*Integrated Development Environment*), para Deitel (2010), as organizações a fim de desenvolverem sistemas de informações, podem buscar recursos em fornecedores de softwares importantes que disponibilizam ambientes de desenvolvimento integrado. Uma IDE fornece ferramentas que suportam o método de desenvolvimento de software, compreendendo editores para criação e alteração de programas e ainda os depuradores para localizar os erros de sintaxe.

As IDE's que agilizam o processo de desenvolvimento, além de apresentar uma tradicional interface com o desenvolvedor, com uso de menus, barras de ferramentas, componentes de interface e editores para aplicações como Netbeans e Eclipse, essas IDE's para desenvolvimento Java, na sua maioria utilizam o componente gráfico Swing Toolkit, fato este que fez com que o framework JPA (Java Persistente API) fosse escolhido devido ao ganho de produtividade e rapidez em relação a outros Frameworks.

O uso de Frameworks é comum entre desenvolvedores, para trazer benefícios a quem desenvolve, como rapidez, visualização, produtividade e segurança. Os benefícios do JPA é realizar o mapeamento objeto-relacional. Para que objetos Java possam ser persistidos em um banco de dados relacional é necessário que eles sejam mapeados para tabelas e seus campos, denominados objeto-relacional, esse mapeamento tem o objetivo de reduzir a impedância da programação OO, com está técnica, o programador não precisa se preocupar com os comandos SQL, ou seja, ele irá usar uma interface de programação simples que faz todo o trabalho de persistência. A forma com que esse mapeamento é configurado neste projeto é a partir do *persistente. xml.*

O *persistence.xml* é um arquivo de configuração que contém as conexões com o banco de dados e as classes que serão persistidas em banco. Nesse arquivo são encontradas informações como a biblioteca de persistência que será utilizada no projeto, a URL (*Uniform Resource Locator)* de conexão com o banco de dados e a estratégia de geração de tabelas. Para fazer a comunicação por linha de código com o banco foi utilizado um comando de persistência chamado Entity Manager, ele é usado para criar, alterar, excluir ou consultar os registros do banco de dados, para que as linguagens de programação ofereçam formas de solução e implementações dessas operações. O armazenamento dos dados é referido como persistência, no sentido que os dados podem

ser mantidos, ou seu estado preservado, após o sistema que os manipula não estar mais em execução, Booch (1998).

Para implementar persistência em Java, a camada responsável por esta atividade mapeia objetos para o banco de dados, de modo que eles possam ser consultados, carregados, atualizados ou removidos sem a necessidade de sintaxe SQL nativa do sistema gerenciador de banco de dados utilizado.

A API (*Application Programming Interface*) de persistência permite que programadores realizem todas as operações de CRUD padrão do banco de dados Russell (2008), desta forma, ao utilizar o JPA, todas as consultas e operações de persistência tornam-se independentes do banco de dados que está sendo utilizado. Com isso, se o banco de dados for mudado o impacto na aplicação é menor. Outras vantagens atribuídas ao uso de JPA segundo O'Conner (2011):

- Não é necessário criar objetos de acesso a dados complexos;
- Possibilidade de escrever código padrão que interage com qualquer base de dados relacional;
- Livre de código e é possível descartar SQL, dando preferência a uma linguagem de consulta que usa os nomes das classes e suas propriedades.

Para possibilitar o desenvolvimento do sistema em Java e seu Framework foi optado pela ferramenta NetBeans IDE 7.1.1, por ser um ambiente de desenvolvimento gratuito e de código aberto para desenvolvedores de software, e diante do conhecimento prévio de utilização, além do IDE ser executado em múltiplas plataformas como Windows, Linux, Solaris e MacOs e trabalhar com SGBD (Sistema Gerenciador de Banco de Dados). Esses, portanto foram os fatores que levaram a escolha dessa ferramenta.

Um sistema gerenciador de banco de dados SGBD (*Data Base Management System)* é uma coleção de programas que permite aos usuários criar e manter um banco de dados, além de ser um sistema de software de uso geral que facilita o processo de definição, construção, manipulação e compartilhamento de banco de dados entre diversos usuário e aplicações Elmasri e Navathe (2011).

É essencial no desenvolvimento do sistema a escolha do SGBD, este deve ser escolhido de acordo com as necessidades da analise de requisitos e também levando em consideração a conectividade com a ferramenta de desenvolvimento. Portanto para auxiliar na criação do software o MYSQL se torna favorável, pois possui uma licença de uso livre (GLP– *General Public License*) e características que influenciaram na escolha pelo MYSQL, como o desempenho e ser compatível tanto com a linguagem de programação citada para o desenvolvimento quanto com softwares de modelagem lógica além de ter um ambiente de fácil operação que se faz presente no phpMyAdmin, que faz a administração do banco de dados e pode gerenciar até mesmo um servidor MYSQL inteiro.

Para Elmasri e Navathe (2011), um banco de dados é projetado, construído e populado com dados para uma finalidade especifica, possui um grupo definido de usuários e algumas aplicações previamente concebidas nas quais esses usuários estão interessados e pode ser facilmente gerenciada via web e executar varias tarefas tais como criar, copiar, eliminar, renomear, alterar, controlar alterações em banco de dados,

tabelas e visualizações e sincronizar dois bancos de dados que estão alocados no mesmo servidor ou em servidores remotos.

Os tópicos abordados como definição da arquitetura do protótipo, bem como a categoria de programação utilizada, a linguagem de programação desenvolvida, a IDE e o SGBD para controlar as funcionalidades do banco de dados. Juntamente com isso foi necessário realizar a modelagem lógica do banco de dados, pois é a base para a organização do armazenamento dos dados gerados pelo software.

Durante ou após o projeto do esquema conceitual, as operações básicas do modelo de dados podem ser as operações básicas do modelo de dados podem ser usadas para especificar as consultas e operações do usuário de alto nível, identificadas durante a análise funcional, Elmasri e Navathe (2011).

As empresas possuem seu patrimônio distribuído entre bens ativos, capital, ações, imóveis, carros, clientes, entre outros, porém o mais valioso recurso de uma empresa são suas informações, buscamos com esse estudo evitar uma serie de fatores que precisarão ser corrigidos posteriormente, uma grande vantagem de prever e prevenir esses erros é a garantia de não haver perdas ou inexatidão nos dados de amanhã, principal motivo para programar a Modelagem de Dados foi, eliminar redundâncias, abranger todos os dados necessários, facilitar a compreensão, e aumentar o poder de realização de transações com alto desempenho.

Além do conhecimento da OO e modelagem de dados, o conhecimento da UML (*Unified Modelling Language*) é essencial para o desenho e planejamento de sistemas. A UML é uma representação gráfica complexa, rica em recursos, amplamente utilizada para modelar sistema OO. Para Deitel (2010), ela de fato unificou os vários esquemas de notações populares fazendo com que aqueles que projetam sistemas a (na forma de diagramas) para modelar seus sistemas. Ainda Para Deitel (2010), o desenvolvimento de um sistema em UML divide-se em cinco fases: analise de requisitos, analise, design (projeto), implementação (programação) e testes, além de ter uma padronização das metodologias de desenvolvimento de sistemas baseados na OO, aonde incorpora noções do desenvolvimento totalmente visual e baseia-se em diagramas que modelados e classificados em visões de abstração.

A análise de requisitos captura as necessidades básicas funcionais e não funcionais do sistema que deve ser desenvolvido, modela o problema principal, o design expande e adapta os modelos da analise para um ambiente técnico. A implementação consiste em codificar em linguagem de programação e banco de dados os modelos criados e as atividades devem testar o sistema em diferentes níveis, verificando se o mesmo corresponde às expectativas do usuário.

A utilização da UML foi essencial para definir e relacionar as atividades de um processo. Contribui com ferramentas que são apresentadas graficamente, essas ferramentas são divididas em diagramas estruturais, diagramas comportamentais e diagramas de interação. Foram especificados aqui os diagramas utilizados no software como o diagrama de casos de uso, diagrama de classe e diagrama de sequência. O diagrama de caso de uso especifica o comportamento de um sistema ou de parte de um sistema e é uma descrição de um conjunto de sequencias de ações, incluindo variantes realizadas pelo sistema para produzir um resultado observável do valor de um ator, Boock, James e Ivar (2005).

Esta visão proporciona suporte principalmente para o comportamento de um sistema, ou seja, os serviços externamente visíveis fornecem no contexto de seu ambiente, para Boock, James e Ivar (2005), os diagramas de casos de uso são importantes para visualizar, especificar e documentar o comportamento de um elemento, esses diagramas fazem com que sistemas, subsistemas e classes fiquem acessíveis e compreensíveis, por apresentarem uma visão externa sobre como esses elementos podem ser utilizados no contexto.

Para enfatizar o fluxo de controle de uma atividade para outra, foi utilizado o diagrama de classes que tem a finalidade de mostrar um conjunto de casos, interfaces e colaborações e seus relacionamentos. Produz a descrição mais próxima da estrutura do código de um programa, ou seja, mostra o conjunto de classes com seus atributos e métodos e os relacionamentos entre classes que relacionadas constituem os elementos sintáticos básicos do diagrama de classes, Silva (2007).

Por fim, foi usado o diagrama de sequência para mostrar uma interação formada por um conjunto de objetos e seus relacionamentos, incluindo as mensagens que poderão ser enviadas entre as classes, em uma situação especifica e delimitada no tempo, coloca ênfase especial na ordem e nos momentos nos quais mensagens para os objetos são enviadas, em diagramas de sequencia objetos são representados através de linhas verticais tracejadas (denominadas como linha de existência), com o nome do objeto no topo. O eixo do tempo é também vertical, aumentando para baixo, de modo que as mensagens são enviadas de um objeto para outro na forma de setas com a operação e os nomes dos parâmetros.

A UML está sendo a base para muitas ferramentas de desenvolvimento, incluindo modelagem visual, simulações e ambientes de desenvolvimento, integrou muitas ideias adversas, e esta integração acelera o uso do desenvolvimento de software orientado a objetos.

Etapas do Desenvolvimento do Projeto

Para alcançar o objetivo proposto deste trabalho foi divido em etapas, dentro das quais se utilizou a metodologia de desenvolvimento XP.

A primeira etapa de desenvolvimento do software proposto SOFTVAV foi focada no levantamento de requisitos e procedimentos sobre a área de vendas junto com alguns vendedores e representantes de software que já existem e pessoas que utilizam como ferramenta de trabalho no seu dia-dia, as dúvidas foram geradas a partir de entrevista direta sobre o processo de vendas, que devido à quantidade de informações coletadas formou-se o desenvolvimento do sistema, aonde esses dados foram filtrados visando um sistema mais produtivo e menos complexo, mas que supram as necessidades.

Na segunda etapa ficou clara a necessidade dos usuários de sistema independentes, desta forma o desenvolvimento de um software voltado para vendas de forma simples, rápida, gerenciar clientes, funcionários, fornecedores, produtos e vendas diárias alem de registros das vendas feitas entre determinada data. Por apresentar maior importância significativa, esses dados foram escolhidos.

A terceira etapa consistiu no desenvolvimento da modelagem do banco de dados após a filtragem dos dados devido à necessidade de manipular informações de forma

rápida e eficiente, aonde as informações são tão relevantes que ganharam uma linha de estudo dentro da Tecnologia e Analise de Sistemas, com desenvolvimento de uma larga gama de conceitos e técnicas que representam uma forma visual e fácil de entendimento entre tabelas e relacionamentos existentes no banco de dados, aonde essa modelagem foi elaborada através do software Power Designer, que por sua vez possui diversos recursos uteis para o gerenciamento de banco de dados MYSQL e apresenta um ambiente gráfico que possibilita a criação de forma pratica da estrutura do banco de dados. O principal motivo da escolha desta ferramenta foi realizar a conexão com o SGBD e fazer com que toda a estrutura do banco de dados seja transformada em código e interpretada pelo SQL (*Structured Query Language* – Linguagem de Consulta Estruturada) para criação da base de dados e as informações geradas. A modelagem do banco de dados pode ser visto no Apêndice A são nove tabelas, que armazenam os dados interagindo com o usuário.

Dentre as tabelas referentes à tabela venda_cab é a mais importante, pois é ela quem vai fazer o relacionamento com as demais tabelas e fazendo a funcionalidade do SOFTVAV, as tabelas venda_det fazem a parte do controle dos produtos e detalhe da venda feita, a tabela produto tem suas referências com a tabela fornecedor e unidade do produto. As demais tabelas fazem parte do controle da venda como a tabela cliente, cliente_endereco, funcionário, tipo_pgto através dos relatórios gerados ao cliente.

Com o termino da modelagem do banco de dados foi iniciado o desenvolvimento dos diagramas, para melhor entendimento do software utilizamos vários tipos de diagramas que combinados, forman todas as visões do sistema. O SOFTVAV possui casos de uso, eles estão representados neste documento de forma gráfica e textual, estes foram divididos em grupos para facilitar seu entendimento.

Foi desenvolvido o caso de uso Manter que demonstra seu funcionamento graficamente, ele possui outros quatro casos de uso relacionados a ele, Alterar, Consultar, Excluir, Incluir ou CRUD (*Create, Read, Update e Delete*) . Este caso de uso pode ser visto no Apêndice B e foi elaborado para demonstrar o funcionamento de um Cadastro, dessa forma como diversos cadastros utilizam esse mesmo formato de caso de uso, estes podem ser representados textualmente indicando-se quais tabelas do banco de dados utilizam-se deste, simplificando o diagrama sem deixar de ser explicativo.

Para efetuar as modelagens do sistema com o auxilio do *ASTAH Community* ferramenta esta que fornece uma interface visual para o desenvolvimento de diagramas UML, na qual são colocados os componentes gráficos para a representação dos modelos, esses componentes são apresentados de acordo com o tipo de modelo de diagrama escolhido. Os modelos de caso de uso deve focalizar o comportamento do sistema SOFTVAV que pode ser visto no Apêndice C.

Na sequencia a visão do projeto se destaca em diagramas de classes que focalizam a estrutura do sistema, mostra as classes, colaborações e as interfaces do sistema gerado pelo *ASTAH Community*, que se encontra no Apêndice D.

E por fim no Apêndice E a visão de implantação que se faz pelo diagrama de sequencia aonde focaliza os módulos que formam a topologia de hardware em que o sistema será executado que também teve o auxilio do *ASTAH Community*.

Depois de ter filtrado passo a passo os processos para o desenvolvimento, de qual seria o ciclo de vida do software e com a modelagem de banco de dados definida,

partiu-se para a última etapa do desenvolvimento aonde já definida a linguagem de programação Java, o principal requisito para esse software, foi à utilização do framework de mapeamento objeto relacional JPA utilizado na camada de persistência, é ele quem facilita e agiliza o processo de desenvolvimento tendo maior produtividade pelo fato de que o JPA abstrai boa parte do uso do JDBC (*Java Database Connectivity*) e de SQL, sem a intenção de dizer que os conhecimentos dessas tecnologias não são importantes para o programador, porém apenas utilizando recursos de OO. A conexão do banco de dados se faz pelo arquivo *persistente. xml* é nele que é feito a configuração de acesso ao banco de dados. O recurso utilizado para fazer a interface criar, remover, pesquisar e atualizar as entidades é chamado de *EntityManager* é responsável por interagir com o contexto de persistência, esse contexto de persistência é um conjunto de instancia de entidade que define os métodos que serão utilizados. A criação de consultas é feita por meio do EntityManager, que fornece métodos específicos para instanciar consultas estáticas e dinâmicas, além de permitir a execução das operações CRUD.

Após definir o framework, a categoria de programação e a linguagem, restou apenas selecionar a IDE onde o código é inserido e as interfaces desenvolvidas. A IDE escolhida foi o NetBeans IDE 7.1.1 pois ela suporta as bibliotecas e auxilia o trabalho do API para a construção do software.

Resultados

Este trabalho teve como objetivo geral avaliar o desempenho produtivo das empresas que já utilizam a TI como objeto de negocio, visto que com ela uma empresa tem um diferencial para apresentar, se destacando das demais. O desenvolvimento desse sistema foi, também, uma oportunidade de aprendizado de tecnologias por meio do uso de Java e JPA no desenvolvimento de um aplicativo para ambiente *desktop*. Porém é um framework de uso não muito intuitivo, mas à medida que a curva de aprendizagem do seu uso aumenta, o desenvolvimento vai se tornando mais fácil e produtivo.

Como resultado desse aprendizado o SOFTVAV auxilia no gerenciamento de vendas de uma forma rápida e fácil, visto na fase de coleta de requisitos que os sistemas que atuam no mercado hoje, atendem as necessidades, mas são muito complexos e geralmente essas pessoas não desfrutam de todas as funcionalidades, assim voltando a fazer ações no papel e caneta desnecessárias por não saberem utilizar o sistema de forma adequada.

Considerações Finais

Por meio da realização deste trabalho, o sistema desenvolvido pode ser utilizado como uma ferramenta auxiliar no processo de gerenciamento de vendas de pequeno porte. O usuário pode fazer uma venda tranquilamente, além de gerenciar os custos de um produto, possibilitando repetições de várias vendas, consequentemente extrair um relatório do que foi vendido, e verificar todas as vendas feitas, contabilizando seu total.

Diante disso, a utilização do JPA representa um ganho considerável na programação de softwares na linguagem JAVA, porem a falta de material didático no idioma português sobre o mesmo torna o aprendizado demorado, o que dificultou o desenvolvimento do SOFTVAV.

Contudo, para que esse sistema se torne uma ferramenta mais efetiva de auxilio ao usuário, como atividades futuras pretende-se dar continuidade ao sistema, promovendo melhorias, tanto na parte de controle de estoque com a criação relatórios mais apurados, contendo um fechamento de caixa diário, quanto na parte funcional de condição de pagamento e numero de parcelas, oferecendo aos gerentes e administradores do sistema opções para maior detalhamento das vendas inseridas no sistema.

Referências:

Bruin, D (2000). “E-mail é novo instrumento de marketing direto”. Gazeta Mercantil. São Paulo, 17 de março, p.C-8.

Booch, G (1998). “Object-oriented analysis and design”. With applications, 2a .ed Addison Wesley.

Boock, Grady. James Rumbaugh. Ivar Jacobson (2005) “UML: guia do usuário”. Rio de Janeiro:elsevier.

Back, Kent, (1999). “Extreme Programming”. Explained: Embrace Change, Addison Wesley Professional.

Campione, M. Walrath, K. (1996) “The Java Tutorial: Object-Oriented Programming for the Internet”. [S.l.]: SunSoft Press.

Dalfovo, O e Amorim, S,N (2000). “Quem tem informação é mais competitivo”. Blumenau:Acadêmica.

Deitel, P. & Deitel, H. (2010). “Java: como programar”. Tradução Edson Furmankiewicz. 8. Ed – São Paulo: Pearson Prentice Hal.

Dall`oglio, Pablo. (2007). “PHP: programando com orientação a objetos”. São Paulo: Novatec Editora.

Flávio Silva (2011). ”Comercio atacadista e varejista trabalha tecnologia a toque de caixa”. Disponível em http://informationweek.itweb.com.br/4603/comercio-atacadista-e-varejista-trabalha-tecnologia-a-toque-de-caixa/.

Lucas, Henry C Jr. (1985). “The analysus,design and implementation of information systems”.3 ed.New York:McGraw Hill.

Malhães, Vinicius, T & Beck, Kent (2005). ”Extreme Programming Aprenda como encantar seus usuário desenvolvendo software com agilidade e alta qualidade” Novatec Editora

Moraes, Anamaria & Mont’alvão, Claudia. (2000) “Ergonomia, conceitos e aplicações”. Rio de Janeiro, 2AB. 2ª Ed.123p.

Nielsen, Jakob(2000). “Projetando Websites” . Rio de Janeiro: Campus

O’Conner, J (2011) “Using java persistence API in desktop applications”. <http://java.sun.com/developer/technicalArticles/J2SE/Desktop/persistenceapi/>. Acesso em 30 abr. 2012.

Pressman, Roger s. (1995) “Engenharia de software” . São Paulo, Akron Books.

Prates, Glaúcia,A & Ospina, Marco,T (2004)"Tecnologia da informação em pequenas empresas:fatores de êxito,restrições e beneficios" Acesso em 14 abr. 2012. <http://www.scielo.br/scielo.php?pid=S1415-65552004000200002&script=sci_arttext>.

Russell, C (2008) "Bridging the object-relational." ACM Queue.

Ramez Elmasri & Shamkant. B.Navathe (2011) "SISTEMAS DE BANCO DE DADOS" 6 edição são Paulo:Pearson addison Wesley.

Silva (2007), R.P e. "UML 2 em Modelagem Orientada a Objetos". Florian polis: Visual Books.

Stair, Ralph ,M (1998)."Principios de sistemas de informação".Rio de Janeiro: LTC

Vinicius Manhães & Kent Beck (2005) "Extreme Programming, Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade" Novatec Editora.

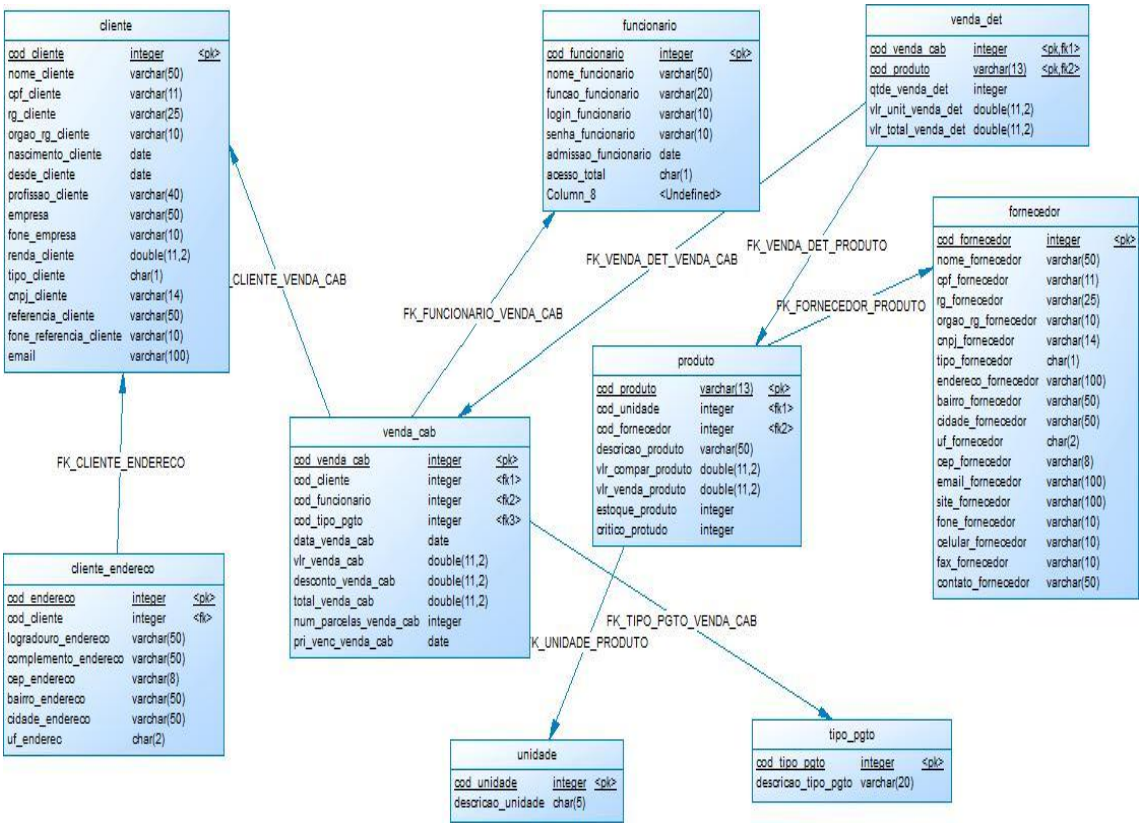
Yong, (1992) "C. S. Tecnologia de informação. Revista de Administração de Empresas", v.32, n.1, p.78-87.

Zuboff, (1994) "S. Automatizar/informatizar as duas faces da inteligente. Revista de Administração de Empresas", v. 34, n. 6, p. 80-91,

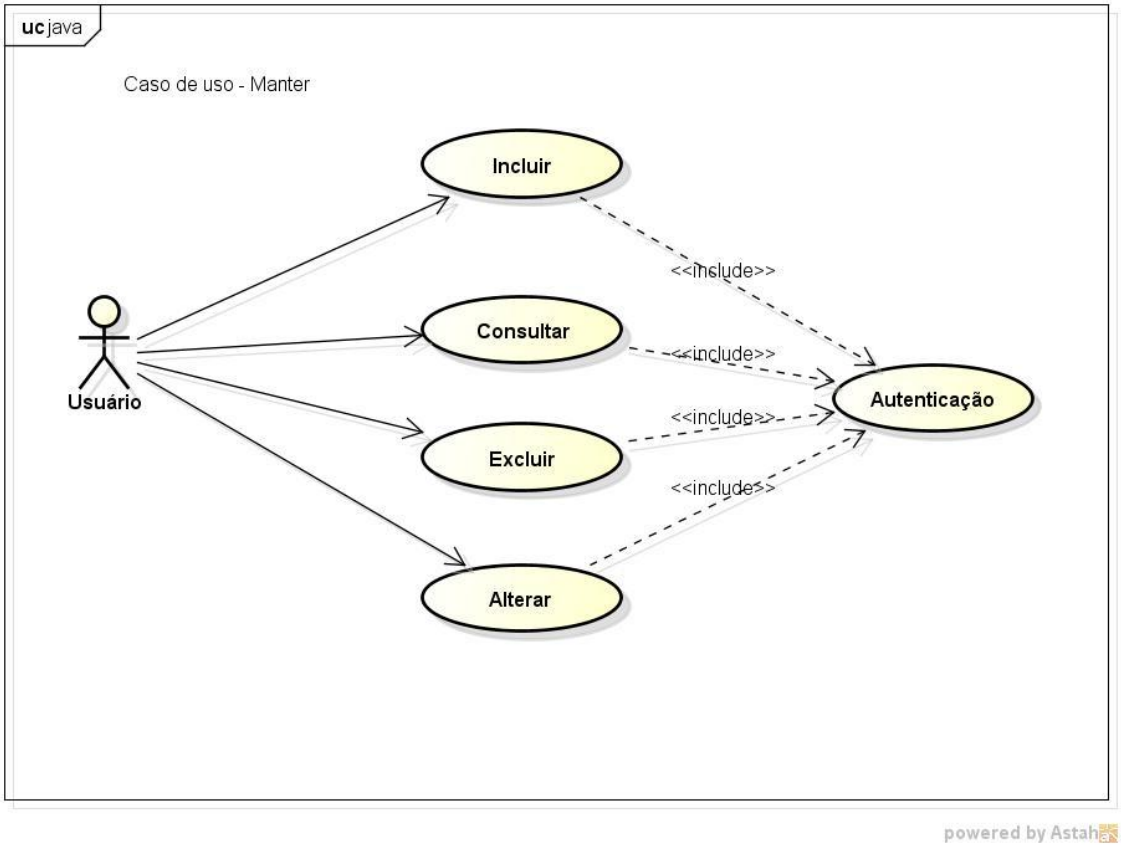
Apêndices

Apêndice A

Modelagem do Banco de Dados

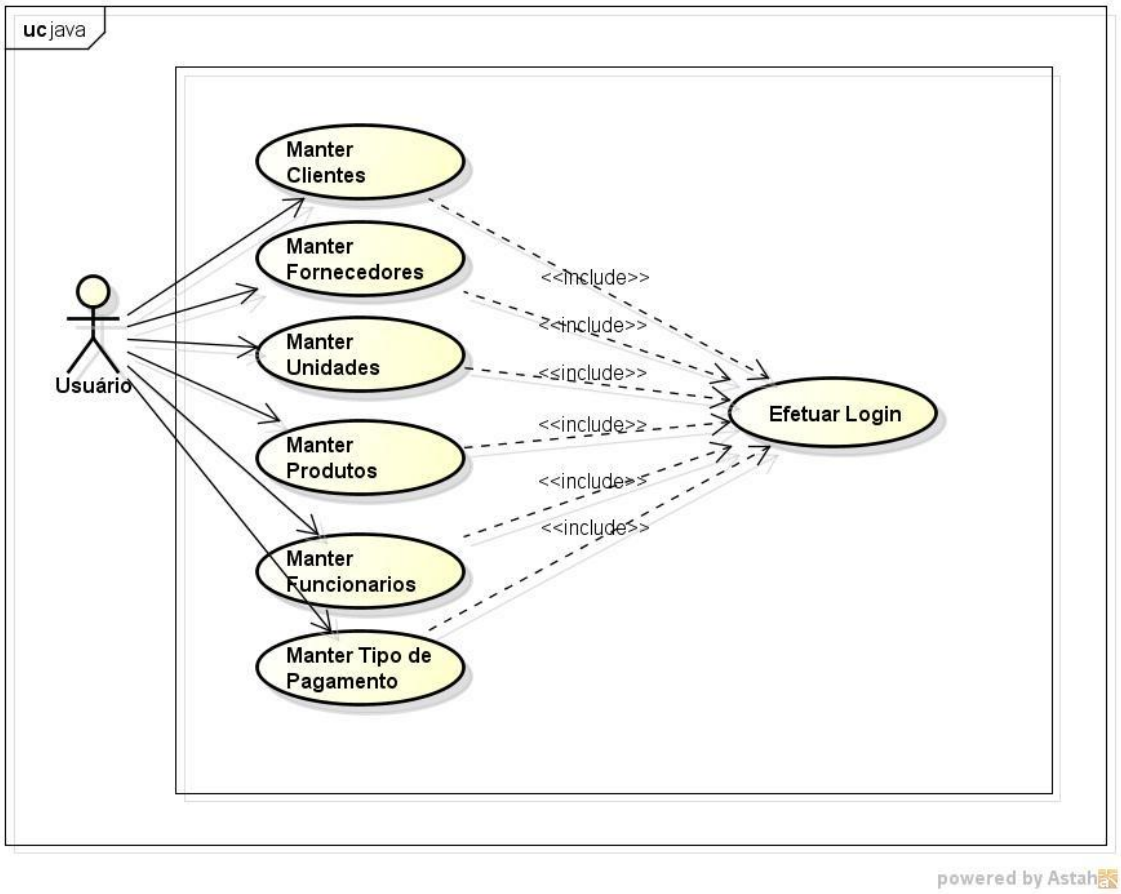


Apêndice B – Caso de Uso Manter.

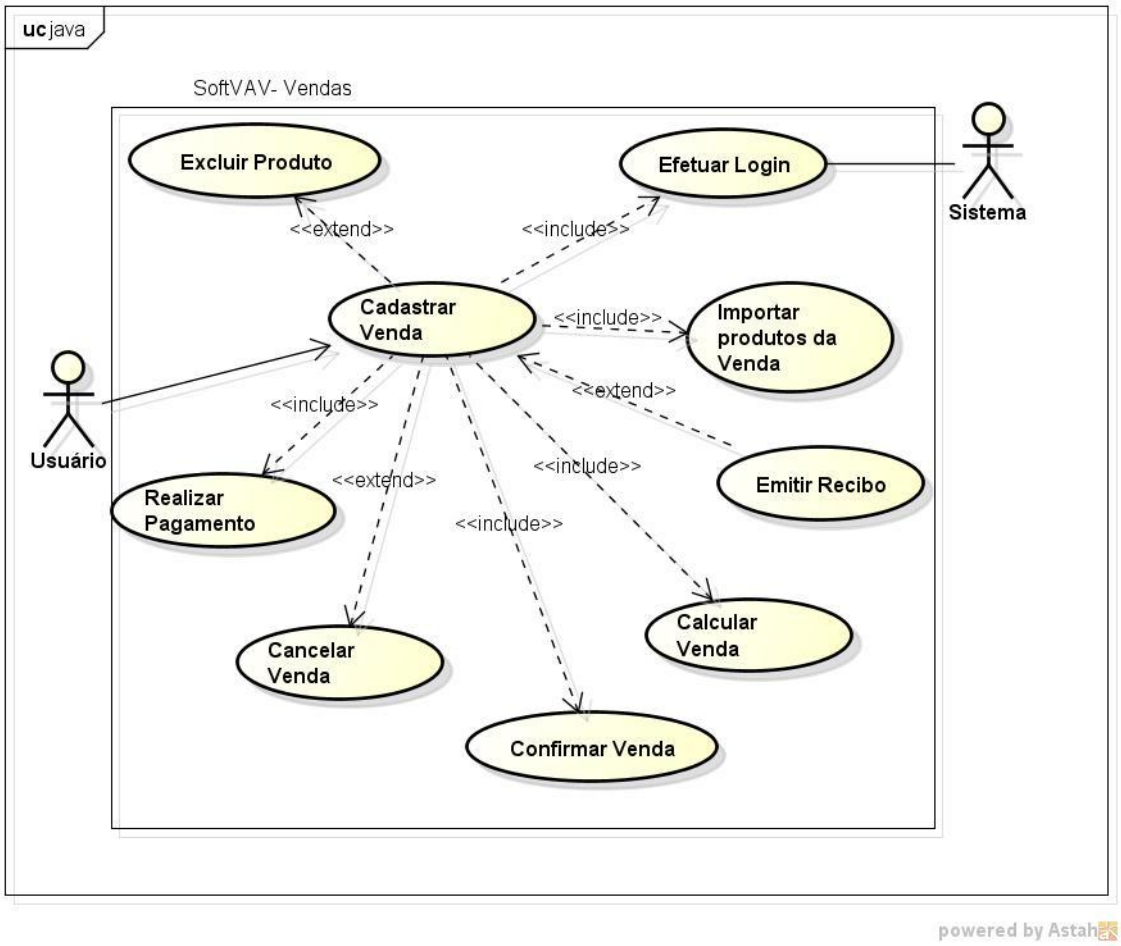


Apêndice C – Casos de Usos

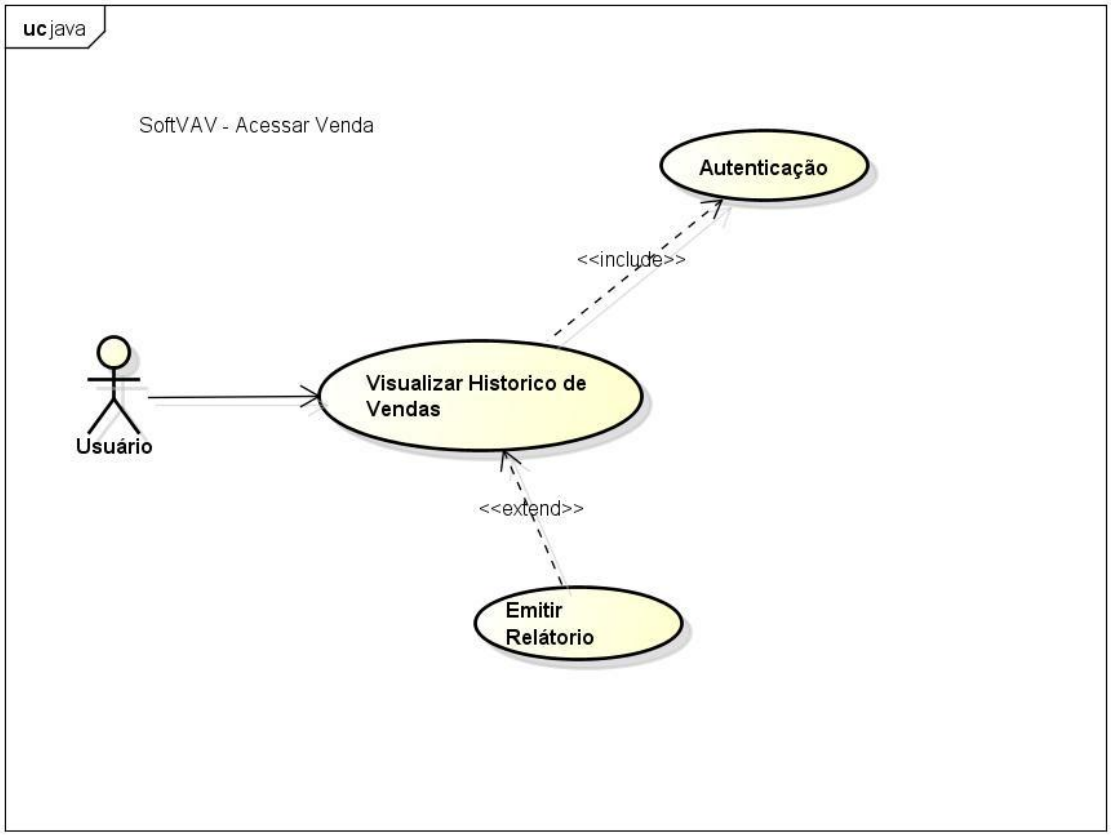
Caso de Uso CRUD



Caso de Uso Vendas

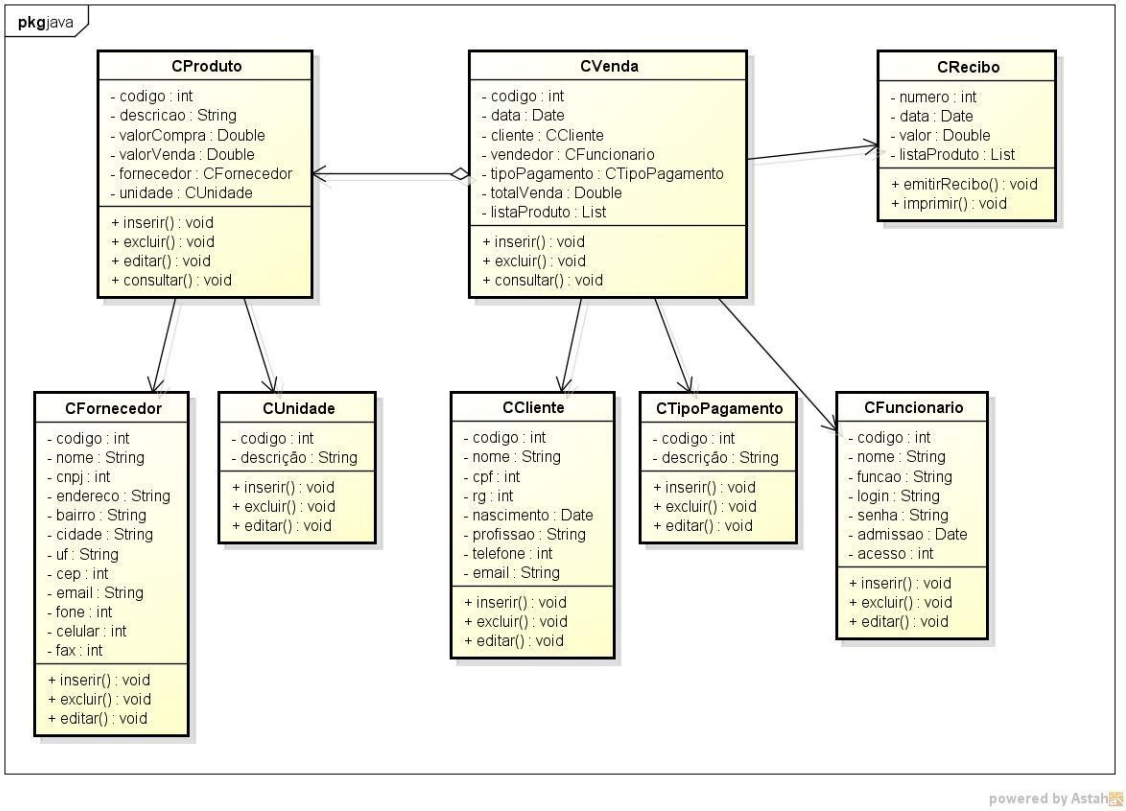


Caso de Uso Acessar Venda



Apêndice D – Diagrama de Classe.

Diagrama de Classe Vendas



Apêndice E – Diagramas de Sequência.

Diagrama de Sequência Login

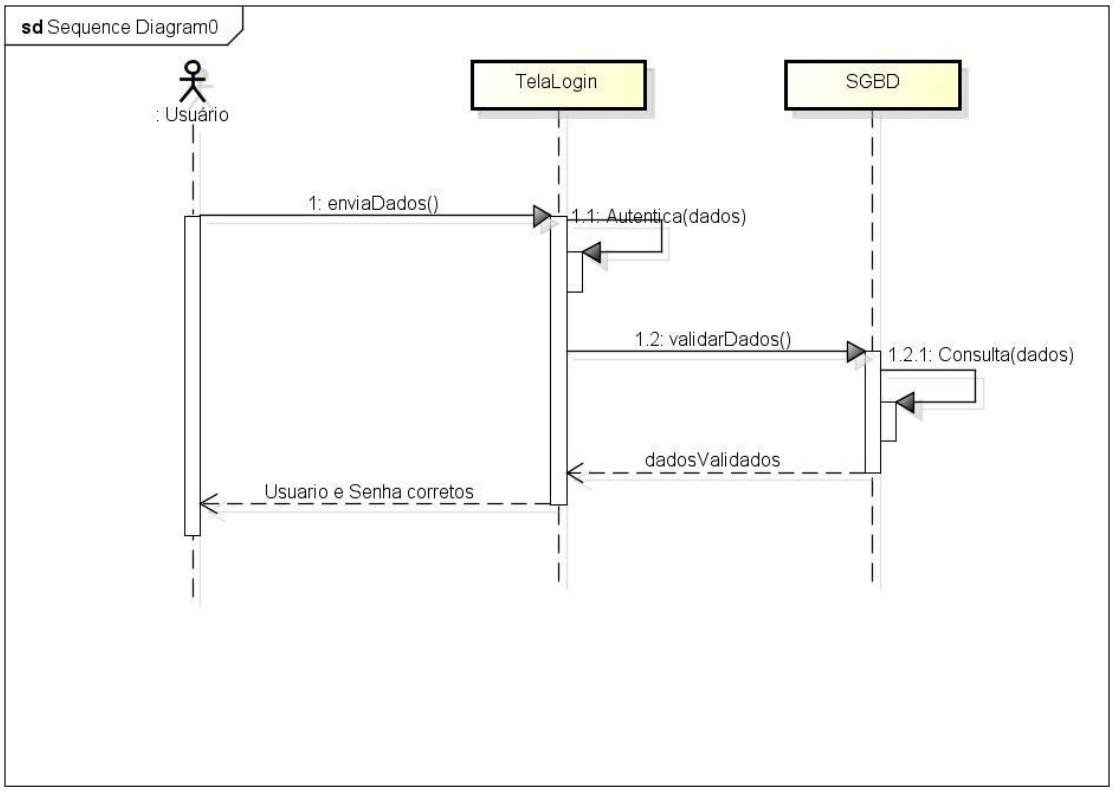


Diagrama de Sequência Vendas

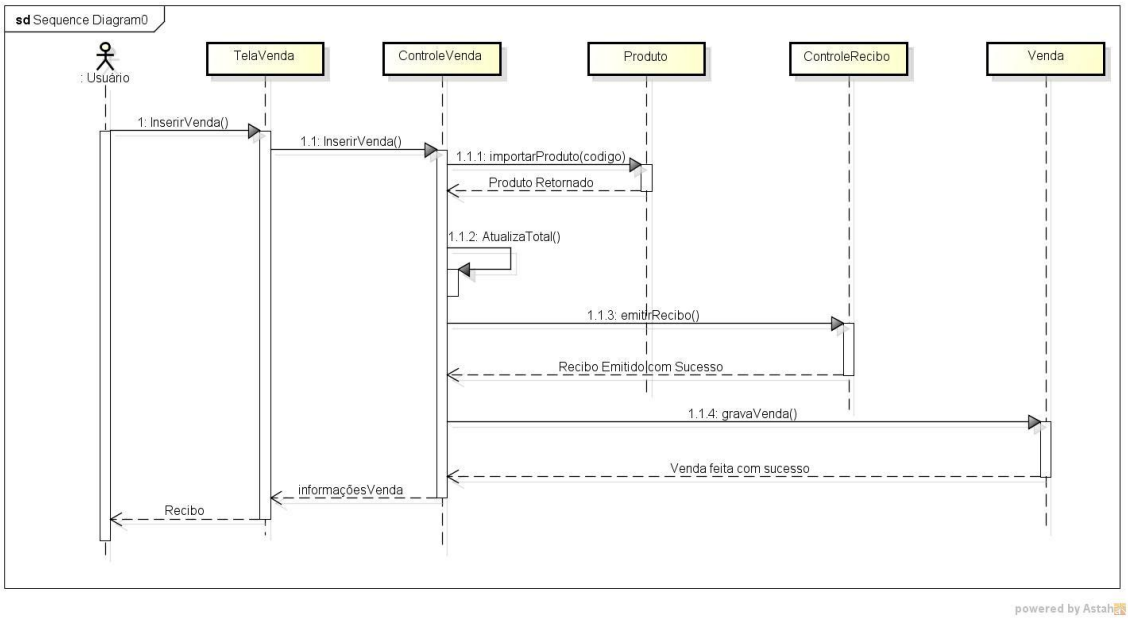


Diagrama de Sequência Acessar Venda

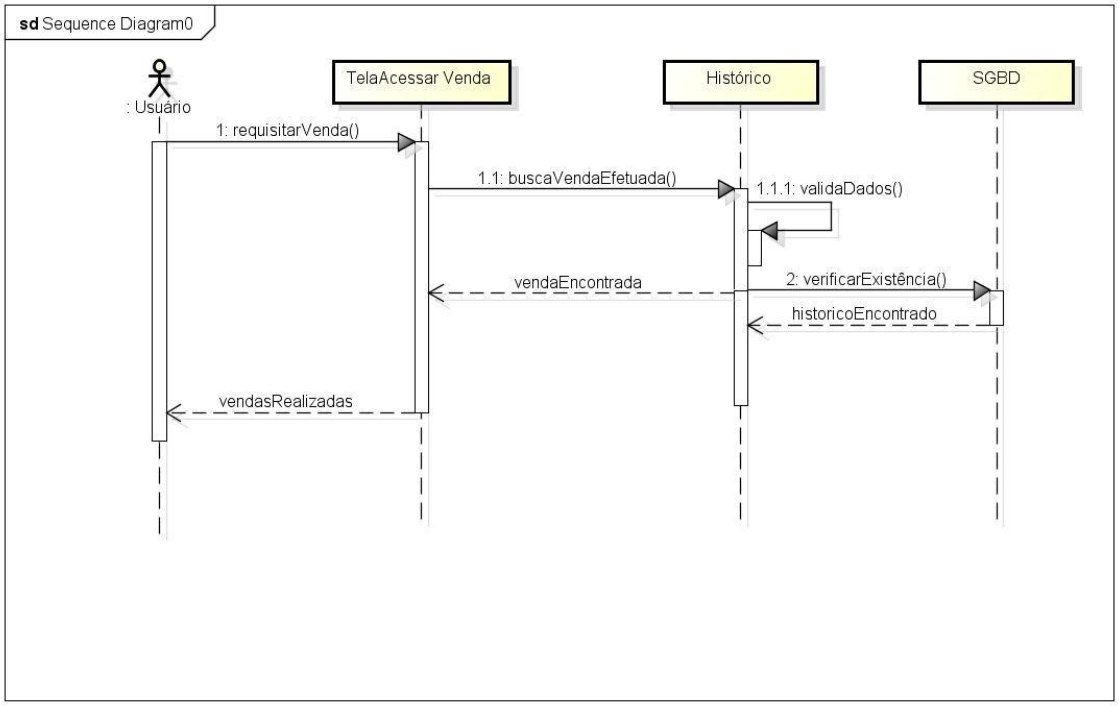


Diagrama de Sequência Cliente

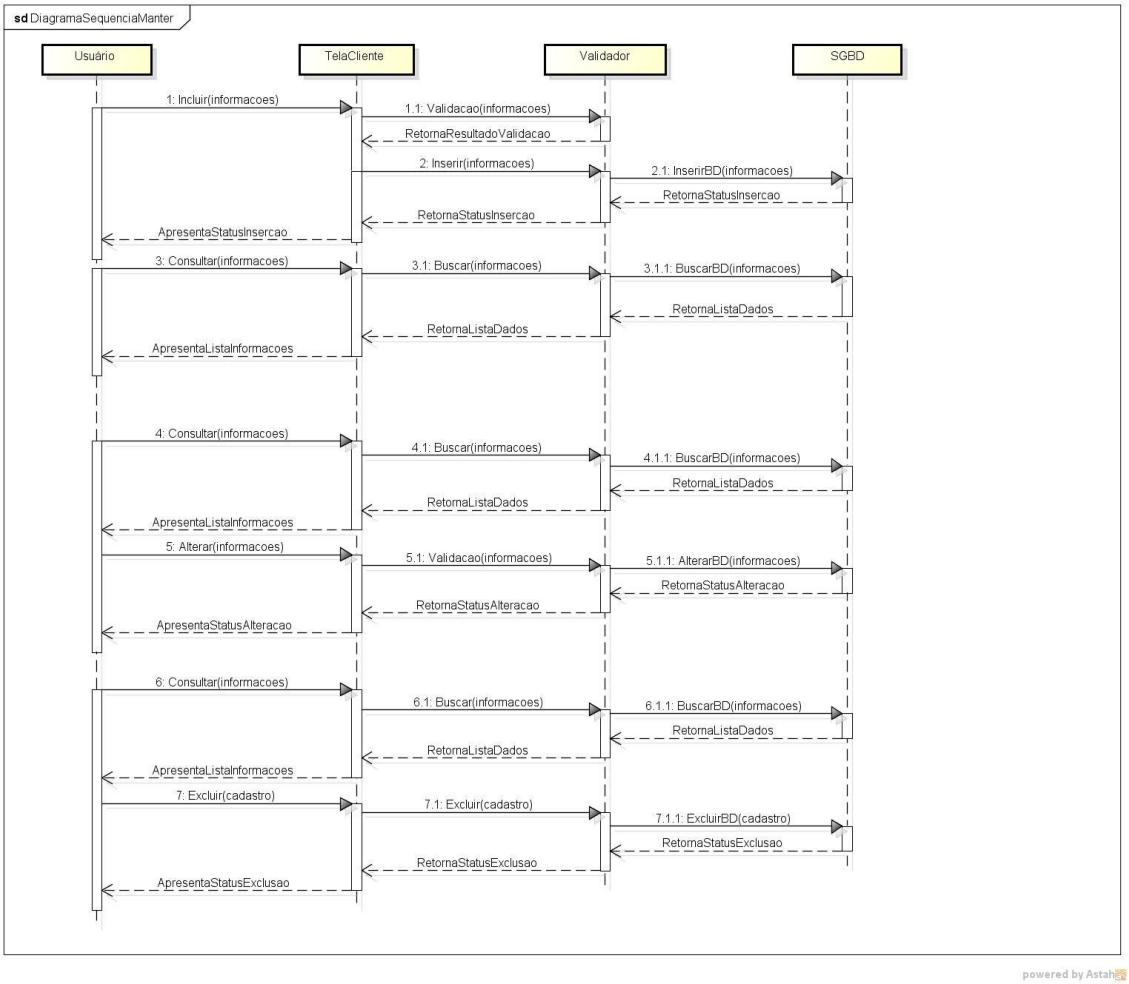


Diagrama de Sequência Fornecedor

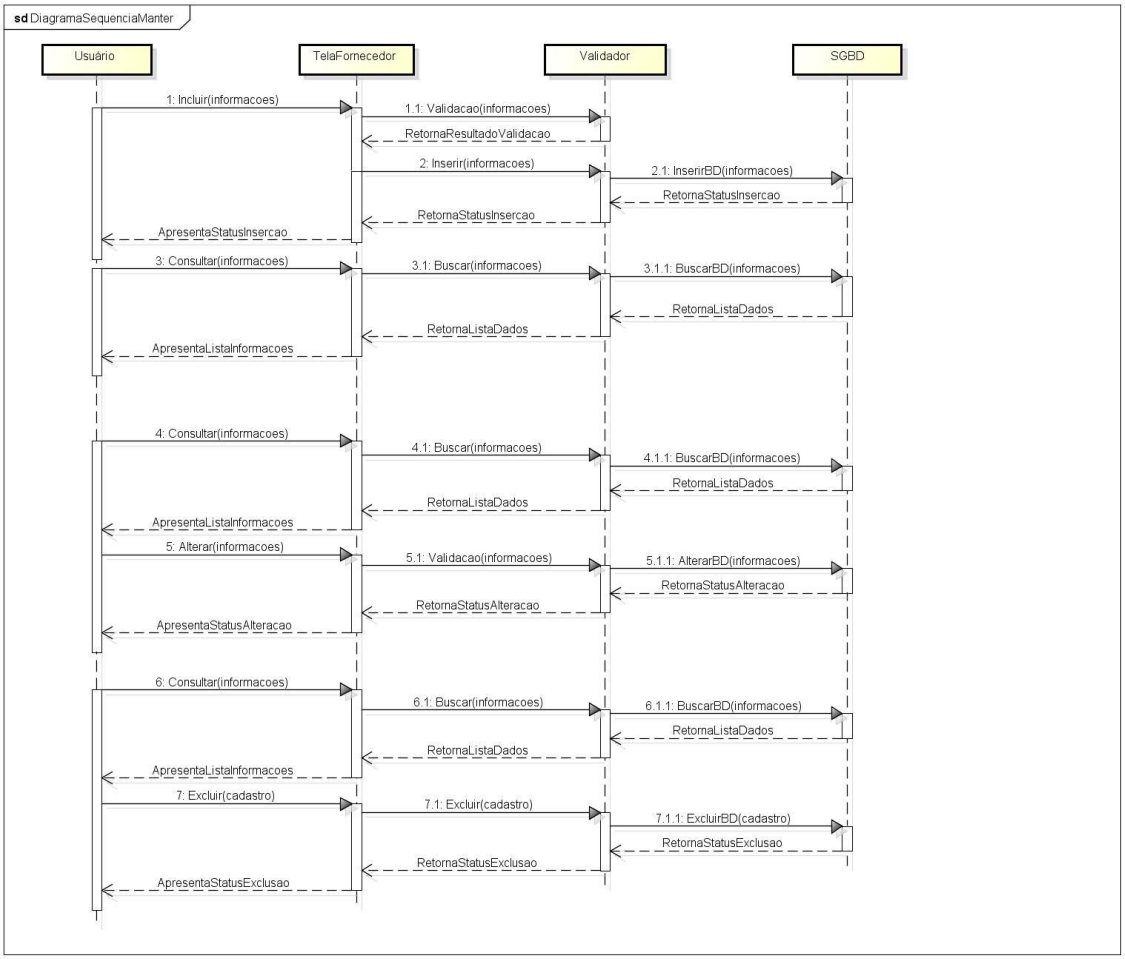


Diagrama de Sequência Unidade

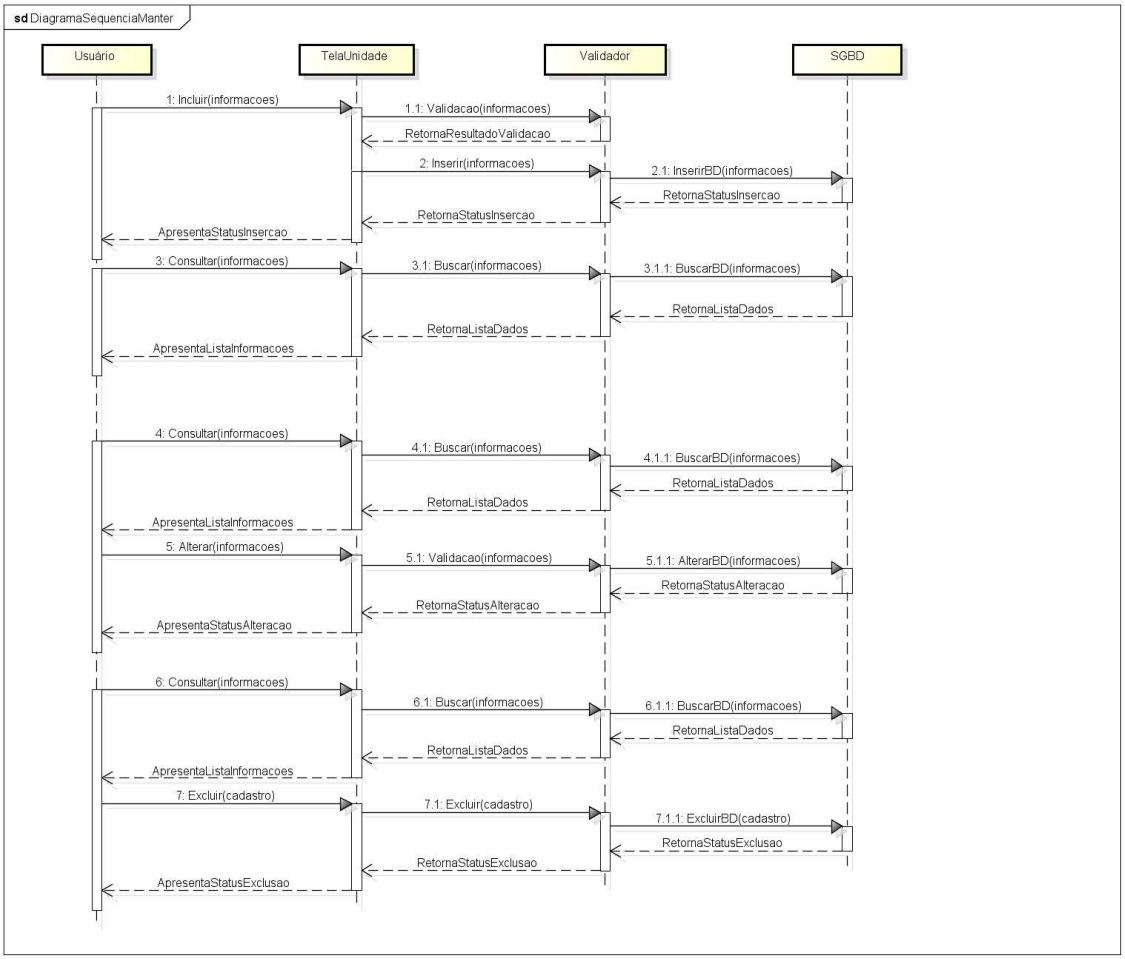


Diagrama de Sequência Produto

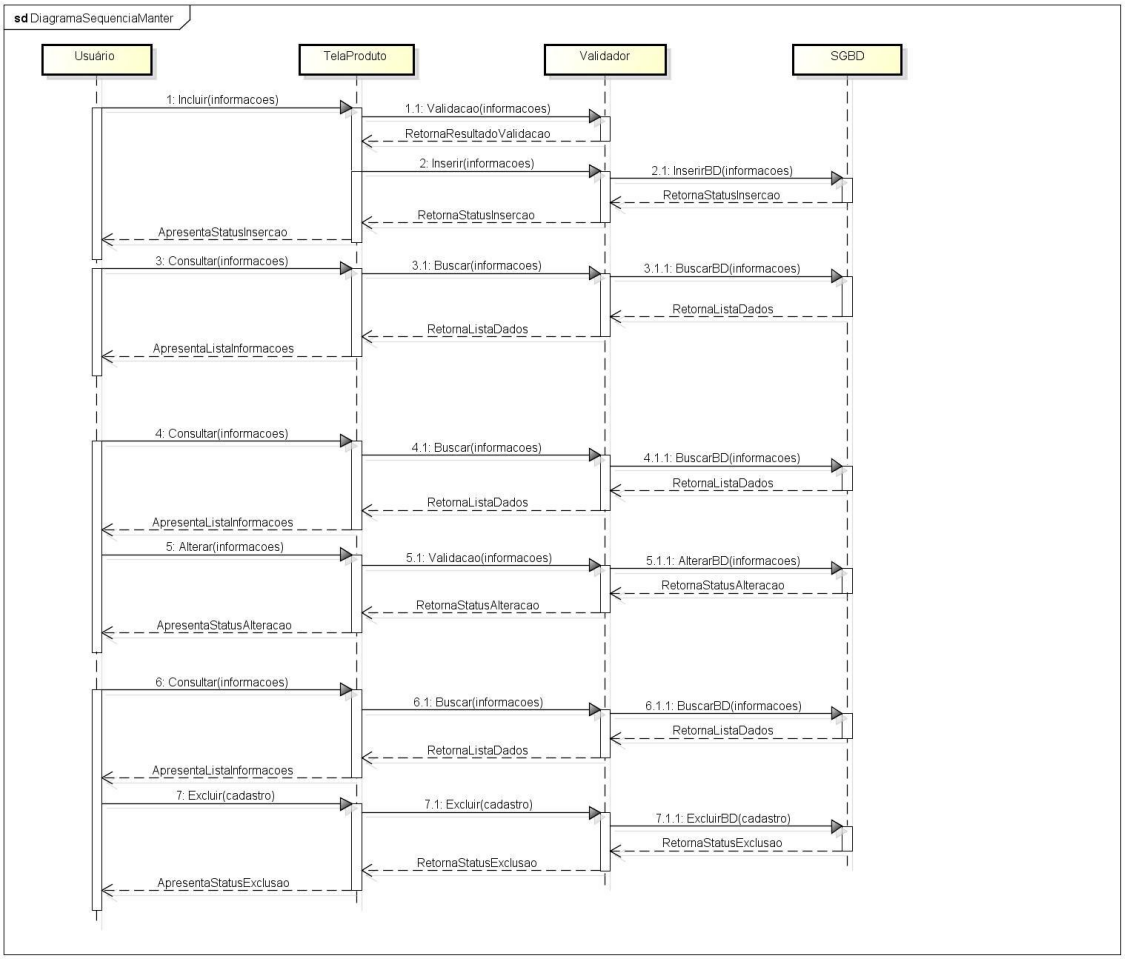


Diagrama de Sequência Funcionário

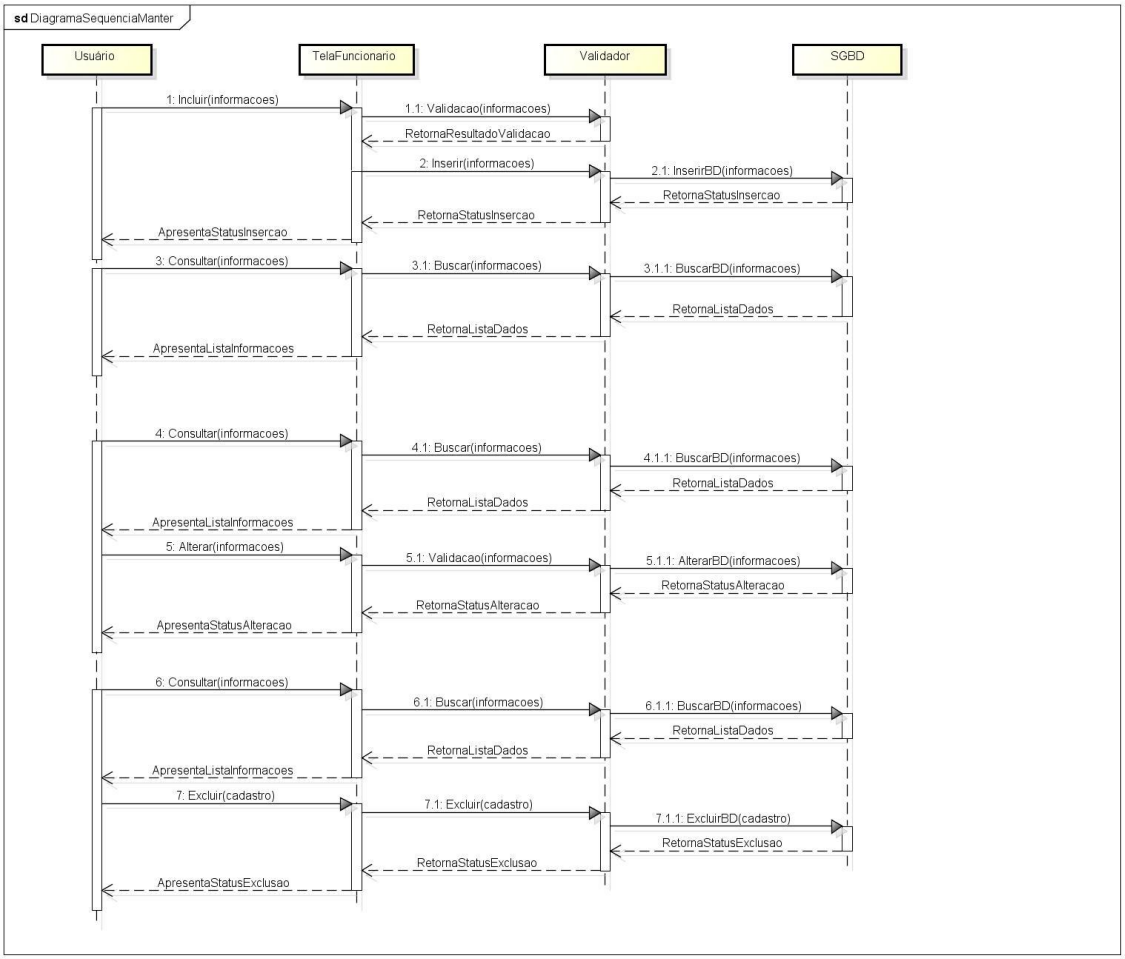


Diagrama de Sequência Tipo de Pagamento