



Sistema de Vendas

Equipe 5

PROJETO E DESENVOLVIMENTO DE SOFTWARE

Integrantes da Equipe 5



ANTONIO LISTER
MATRÍCULA: 2023098548



GABRIEL FERNANDO
MATRÍCULA: 2022024020



KELY SOUZA
MATRÍCULA: 2020051163



LUCAS TEIXEIRA
MATRÍCULA 2020056301



VICTOR COELHO
MATRÍCULA 2019050753

Sumário

- 01. Introdução
- 02. Objetivo
- 03. Estrutura do projeto
- 04. Requisitos
- 05. Requisitos funcionais
- 06. Requisitos não funcionais
- 07. Regras de Negócio
- 08. Diagrama de Caso de Uso
- 9. Modelagem de Classes de Análise
- 10. Modelagem de Interação e Classes de Projeto
- 11. Modelagem de Estados e Atividades
- 12. Conclusão

Introdução

SISTEMA DE VENDAS

A incessante busca por eficiência e automação no ambiente empresarial contemporâneo, impulsionada pela necessidade de otimização de processos e aprimoramento da gestão, reflete a visão de Philip Kotler (1998), sobre a importância de soluções integradas. Nesse contexto, a proposta de desenvolver um sistema abrangente de vendas surge como resposta à demanda por soluções capazes de gerenciar desde o cadastro de funcionários e produtos até a conclusão de vendas e o registro detalhado de transações, alinhando-se às perspectivas de Kotler sobre a eficiência operacional e a gestão estratégica





Objetivo

SISTEMA DE VENDAS

O principal objetivo deste projeto, alinhado à visão de especialistas em gestão de projetos como Harold Kerzner (2017), é desenvolver um sistema abrangente de gestão de vendas que ofereça uma solução completa e integrada para otimizar os processos comerciais. Isso será alcançado por meio da implementação de um conjunto de funcionalidades.

Estrutura do Projeto

01

BANCO DE DADOS:



SQL sever

02

CADASTRO DE USUÁRIO



adicionar usuário à base de dados

04

SISTEMA DE LOGIN:



verifica as credenciais do usuário

03

CADASTRO DE PRODUTOS:



adicionar produtos à base de dados

Estrutura do Projeto

05

VENDAS:



interface para realizar vendas

06

REGISTROS E RELATÓRIOS:



Informações de cada venda

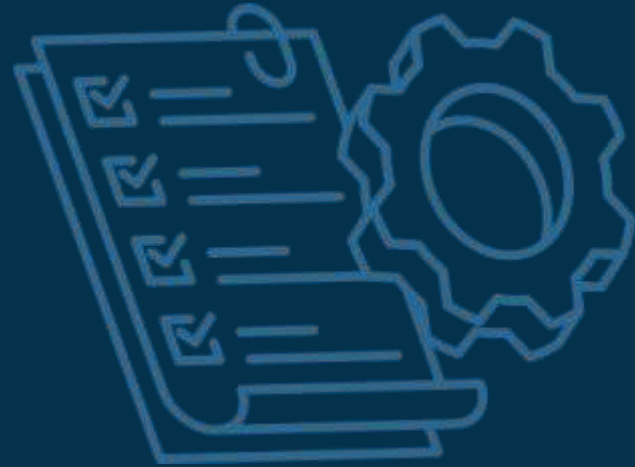
Requisitos



Requisito é uma característica do sistema ou descrição de algo que o sistema é capaz de fazer (Pleeger, 2004).

Requisitos são propriedades que um sistema de software deve ter em busca do seu sucesso no ambiente onde será utilizado (Goguen, 1994).

Níveis de requisitos



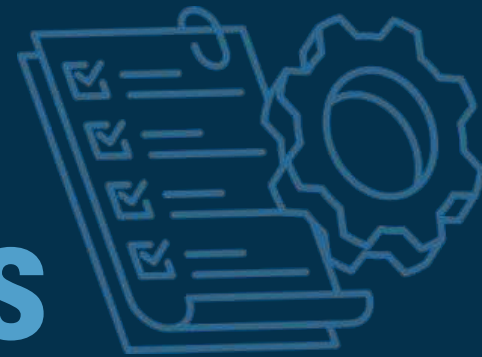
Requisitos de usuário:

- Declaração em linguagem natural e inteligível pelo cliente, podendo ser apoiado por diagramas. (SOMMERVILLE, 2008).

Requisito de sistema:

- Estabelecem detalhadamente as funções, serviços e restrições, em um formatado mais apropriado para implementação. (PRESSMAN, 2006)

Requisitos Funcionais



- Representação de algo que o sistema deve fazer
- Descrição de interações entre o sistema e o ambiente



Requisitos Funcionais

RF01

RF02

RF03

RF04

RF05

RF06

FAZER LOGIN

RF01: O sistema deve permitir que os usuários façam login usando credenciais válidas (nome de usuário e senha).

AUTENTICAR USUÁRIO

RF02: O sistema deve autenticar os usuários com base nas informações armazenadas no banco de dados.

GERENCIAR USUÁRIO

RF03: O sistema deve permitir o cadastro de usuários, incluindo informações como nome, cargo e credenciais de login. E também permitir a edição, visualização e exclusão.

GERENCIAR PRODUTO

RF04: O sistema deve permitir a adição, visualização, edição e exclusão do produto no banco de dados.

REALIZAR VENDA

RF05: O sistema deve permitir que os funcionários selecionem produtos para venda. Deve ser possível aplicar descontos em produtos específicos ou no valor total da venda. O sistema deve permitir o cancelamento da venda.

RELATÓRIOS DE VENDA

RF06: Os registros de vendas devem incluir informações completas sobre cada transação, como número da venda, produtos vendidos, valor, data e hora, e funcionário responsável.

Requisitos Não Funcionais

- Estão diretamente as qualidades específicas que um software deve ter.
- Ou ainda, alinhados a restrições adicionadas ao sistema.

Categorias

- Desempenho
- Segurança
- Confiabilidade
- Escalabilidade
- Compatibilidade
- Interoperabilidade (integrar)
- Documentação



Requisitos Não Funcionais

RNF01

RNF02

RNF03

RNF04

RNF05

RNF06

TEMPO DE RESPOSTA

O sistema deve responder rapidamente às solicitações dos usuários, levando no máximo **5 segundos**, garantindo tempos de resposta curtos para ações comuns.

ESCALABILIDADE

O sistema deve ser capaz de lidar com pelo menos 50 usuários simultâneos sem degradação significativa no desempenho.

AUTENTICAÇÃO E AUTORIZAÇÃO

O sistema deve ter acesso baseado em papéis para garantir que apenas usuários autorizados tenham acesso e permissões específicas, protegendo os dados sensíveis.

CRIPTOGRAFIA

As informações sensíveis, como senhas e informações de pagamento, devem ser armazenadas e transmitidas de forma segura.

BACKUP

O sistema deve ter um backup diário dos dados para evitar a perda de informações em caso de falha.

DISPONIBILIDADE

O sistema deve estar disponível a maior parte do tempo. Deve ser capaz de se recuperar automaticamente de falhas, minimizando o tempo de inatividade.

Requisitos Não Funcionais

RNF07

RNF08

RNF09

RNF10

RNF11

RNF12

INTERFACE

A interface do usuário deve ser intuitiva e fácil de usar, exigindo o mínimo de treinamento para os usuários.

CLAREZA

O sistema deve fornecer feedback claro e mensagens de erro compreensíveis para orientar os usuários em caso de problemas.

COMPATIBILIDADE

O sistema deve ser compatível com os principais navegadores web, como Chrome, Firefox, Safari e Edge. Assim como, com sistemas operacionais Windows, macOS e Linux.

INTEROPERABILIDADE

sistema deve ser capaz de se integrar com outros sistemas, como sistemas de contabilidade ou sistemas de gerenciamento de estoque.

MANUTENIBILIDADE

O sistema deve ser modular, permitindo a fácil adição ou remoção de funcionalidades sem afetar outras partes do sistema.

DOCUMENTAÇÃO

O sistema deve ter uma documentação abrangente, incluindo manual do usuário, guia de instalação e documentação técnica para facilitar o entendimento e a manutenção do sistema

Regras de negócio

Regras de negócio são uma nova categoria de requisitos do sistema que representam decisões sobre como executar o negócio, e são caracterizadas pela orientação do negócio e sua tendência às mudanças (Rosca et al., 1997).



Regras de negócio

RN01

RN02

RN03

RN04

RN05

RN06

AUTENTICAÇÃO DE USUARIOS

RN01: O sistema deve permitir que os usuários façam login usando credenciais válidas (nome de usuário e senha).

GERENCIAMENTO DE FUNCIONÁRIOS

RN02: O nome de usuário de um funcionário deve ser único no sistema.

GERENCIAMENTO DE PRODUTOS

RN03: O código de identificação único de um produto (SKU) deve ser único no sistema.

REALIZAÇÃO DE VENDAS

RN04: A venda só pode ser finalizada se houver pelo menos um produto na lista de itens vendidos.

GERAÇÃO DE RELATÓRIOS

RN05: Os relatórios diários de vendas devem incluir informações precisas sobre vendas, lucros e gastos do dia.

SEGURANÇA

RN06: As transações devem ser protegidas contra acessos não autorizados. Apenas usuários autenticados têm permissão para realizar vendas ou acessar informações confidenciais.

Modelagem Funcional

A modelagem funcional é uma abordagem na engenharia de software que se concentra na representação das funcionalidades e comportamentos de um sistema, destacando as interações entre seus componentes (JACOBSON et al. 2006)

Quais são as funcionalidades?

- Casos de uso

Quem fará uso dessas funcionalidades?

- Atores

Onde acontecerão as interações ator/caso de uso?

- Sistema

Diagrama Casos de Uso

Uma representação gráfica que ilustra as interações entre os atores (usuários, outros sistemas ou entidades externas) e os casos de uso em um sistema.

Tem o objetivo de ilustrar em um nível alto de abstração quais elementos externos interagem com que funcionalidades do sistema. (Booch et al. 2006)

Diagrama Caso de Uso

SISTEMA

- O bloco retangular que representa o sistema é conhecido como "sistema de fronteira" ou "fronteira do sistema". Ele age como uma caixa que envolve os elementos relacionados ao sistema.

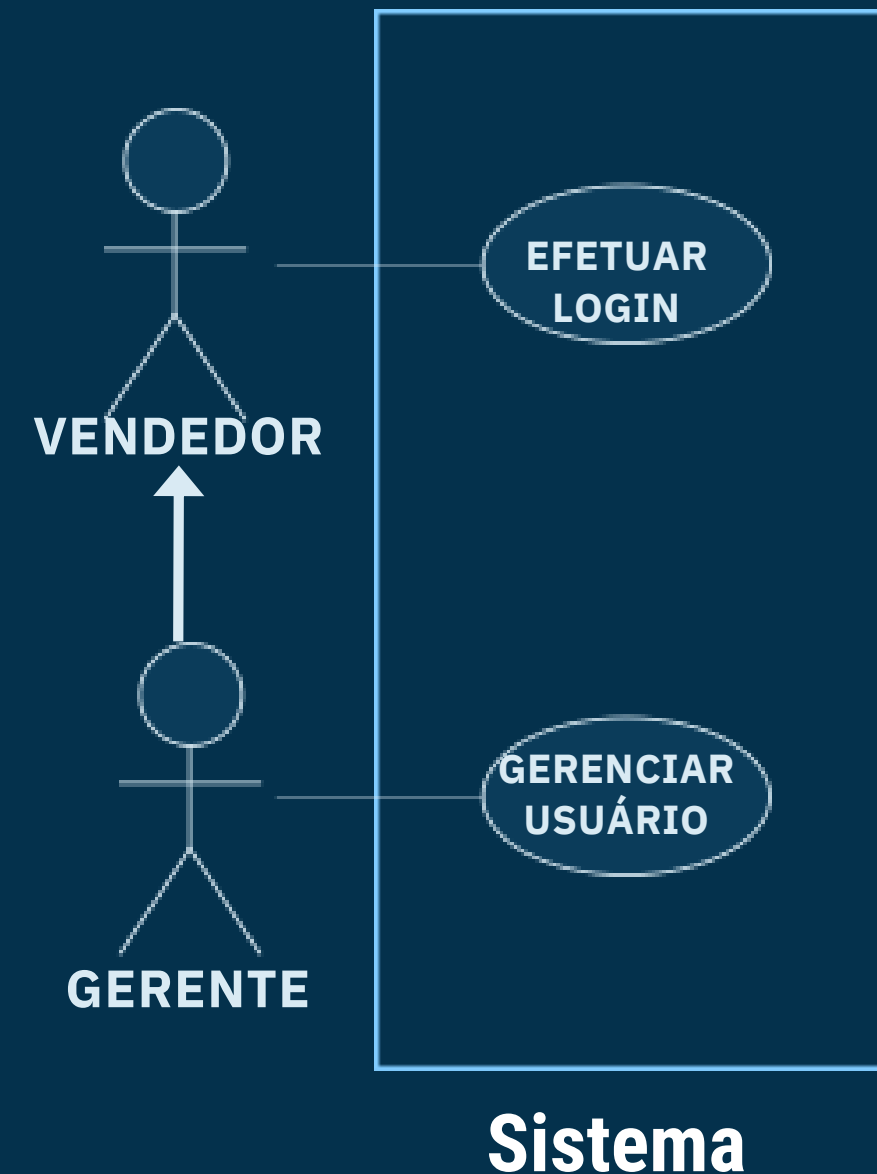


Diagrama Casos de Uso

ATOR



- Representado por um boneco palito;
- Qualquer elemento externo ao sistema que interage com ele;
- Corresponde a um papel representado em relação ao sistema;
- Troca informações com o sistema.

Diagrama Casos de Uso

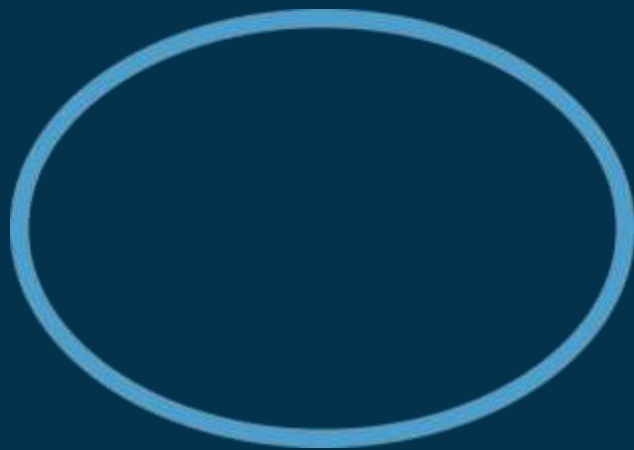
Exemplos de Tipo de atores:

| CARGOS | SETORES | SISTEMAS | EQUIPAMENTOS |
|--|---|--|---|
| <ul style="list-style-type: none">• Atendente• Vendedor• Gerente | <ul style="list-style-type: none">• Financeiro• Atendimento• Gerência | <ul style="list-style-type: none">• Faturamento• RH• Segurança | <ul style="list-style-type: none">• Servidor WEB• Central de Comunicação |



Diagrama Casos de Uso

CASO DE USO



- Representado por uma elipse;
- Descreve uma interação específica entre o ator e o sistema;
- Um caso de uso representa uma funcionalidade ou uma unidade de trabalho específica que o sistema deve realizar em resposta a uma solicitação do ator.

Diagrama Casos de Uso

RELACIONAMENTOS

- Associação entre um ator e um caso de uso
- Generalização de um ator
- Extensão do relacionamento entre dois casos de uso
- Inclusão da relação entre dois casos de uso
- Generalização de um caso de uso

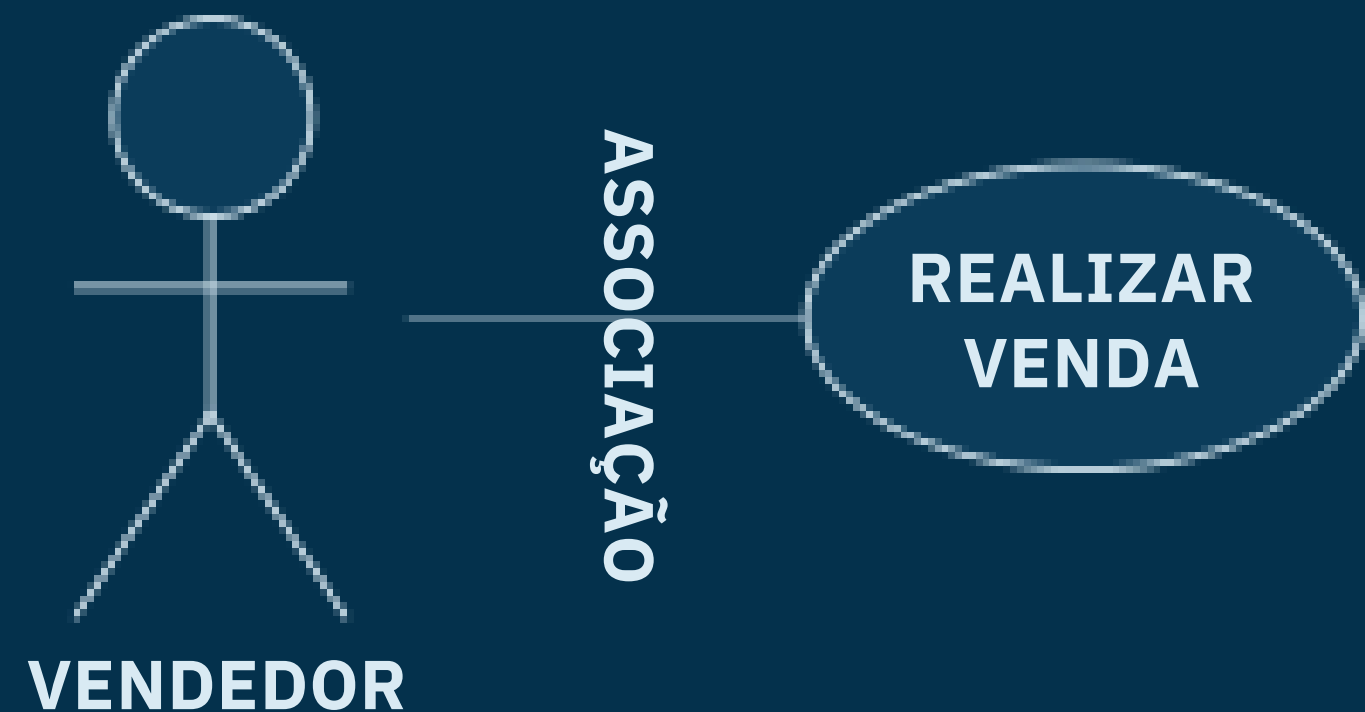


Diagrama de Casos de Uso

Relacionamento de Inclusão (Include)

A relação de inclusão é representada por uma seta com a notação <<include>> apontando para o caso de uso incluído.

<<include>> →

- Inclusão de Comportamento
- Reutilização de Funcionalidades Comuns.

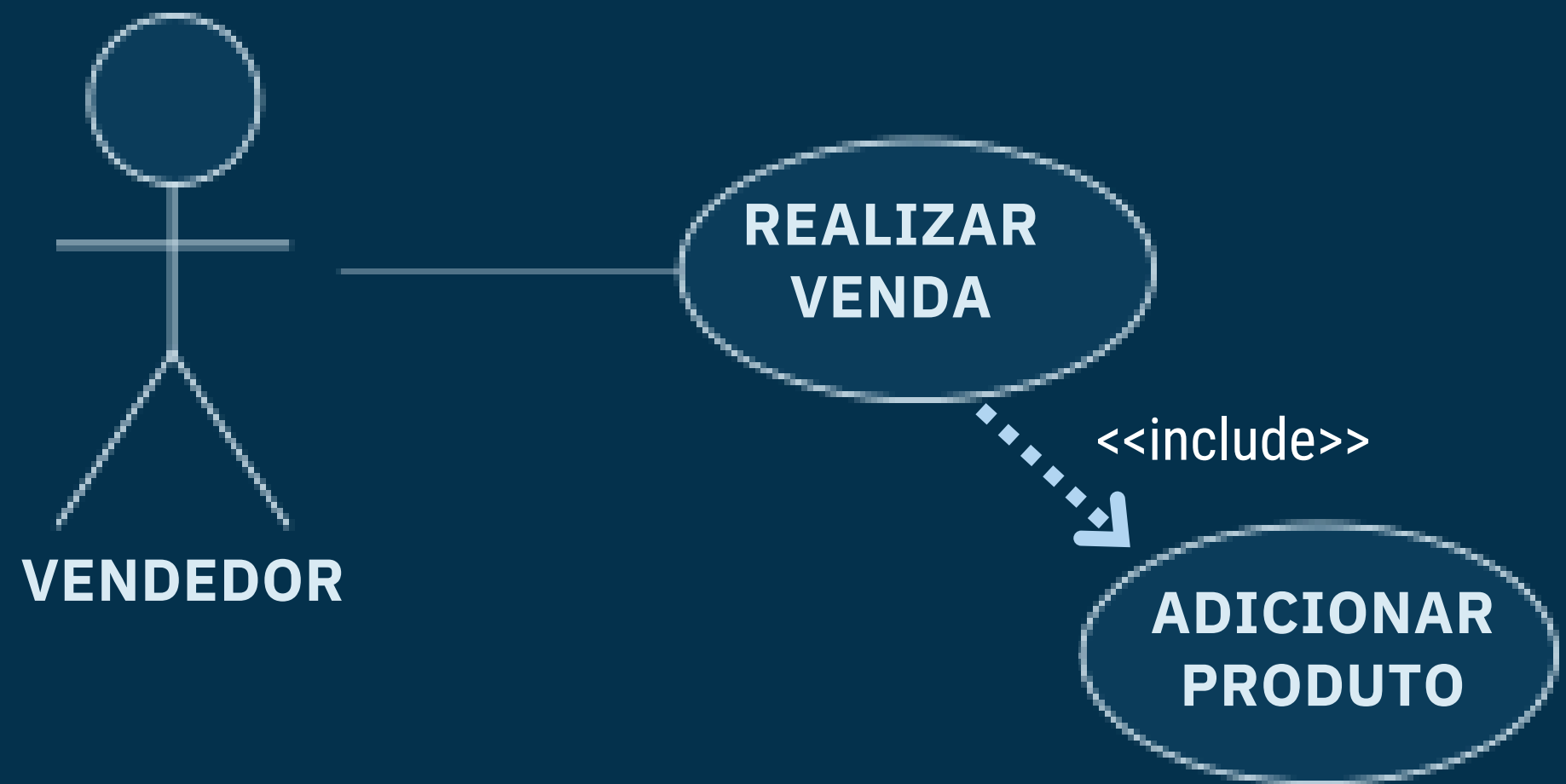


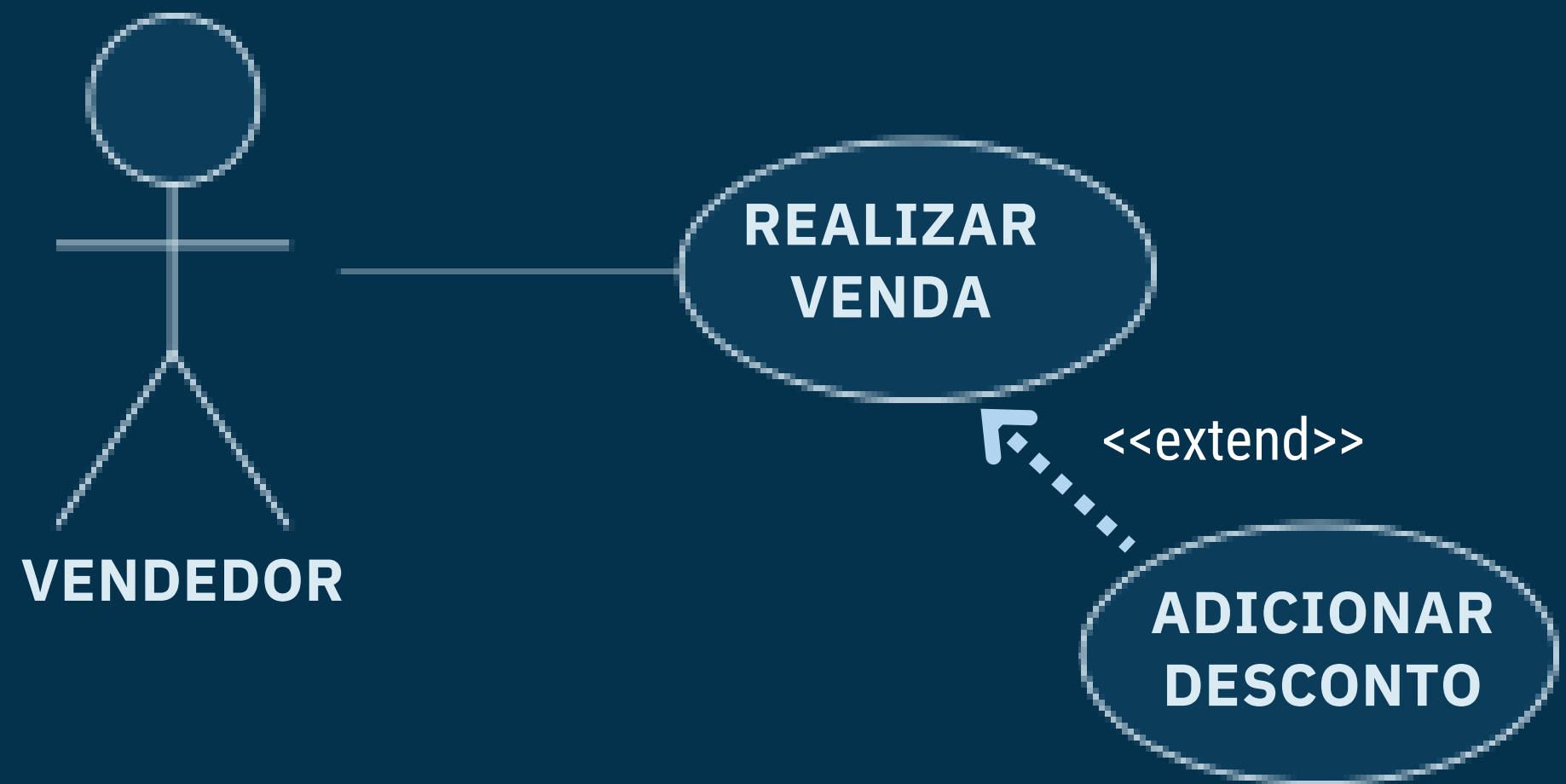
Diagrama de Casos de Uso

Relacionamento de Extensão (Extend)

A relação de extensão é representada por uma seta com o estereótipo <<extend>> apontando do caso de uso estendido para o caso de uso base.

←--->
<<extend>>


- Extensão de Comportamento.
- Ponto de Extensão.



No exemplo acima, o "Caso de Uso Estendido" estende o "Caso de Uso Base".

Diagrama Casos de Uso

Relacionamento de Generalização / Especialização

- 
- Aplica os princípios da herança da orientação a objetos
 - Indica que o "Caso de Uso Específico" herda características do "Caso de Uso Geral".
 - a generalização de atores indica que um ator mais específico herda comportamentos e características de um ator mais geral.

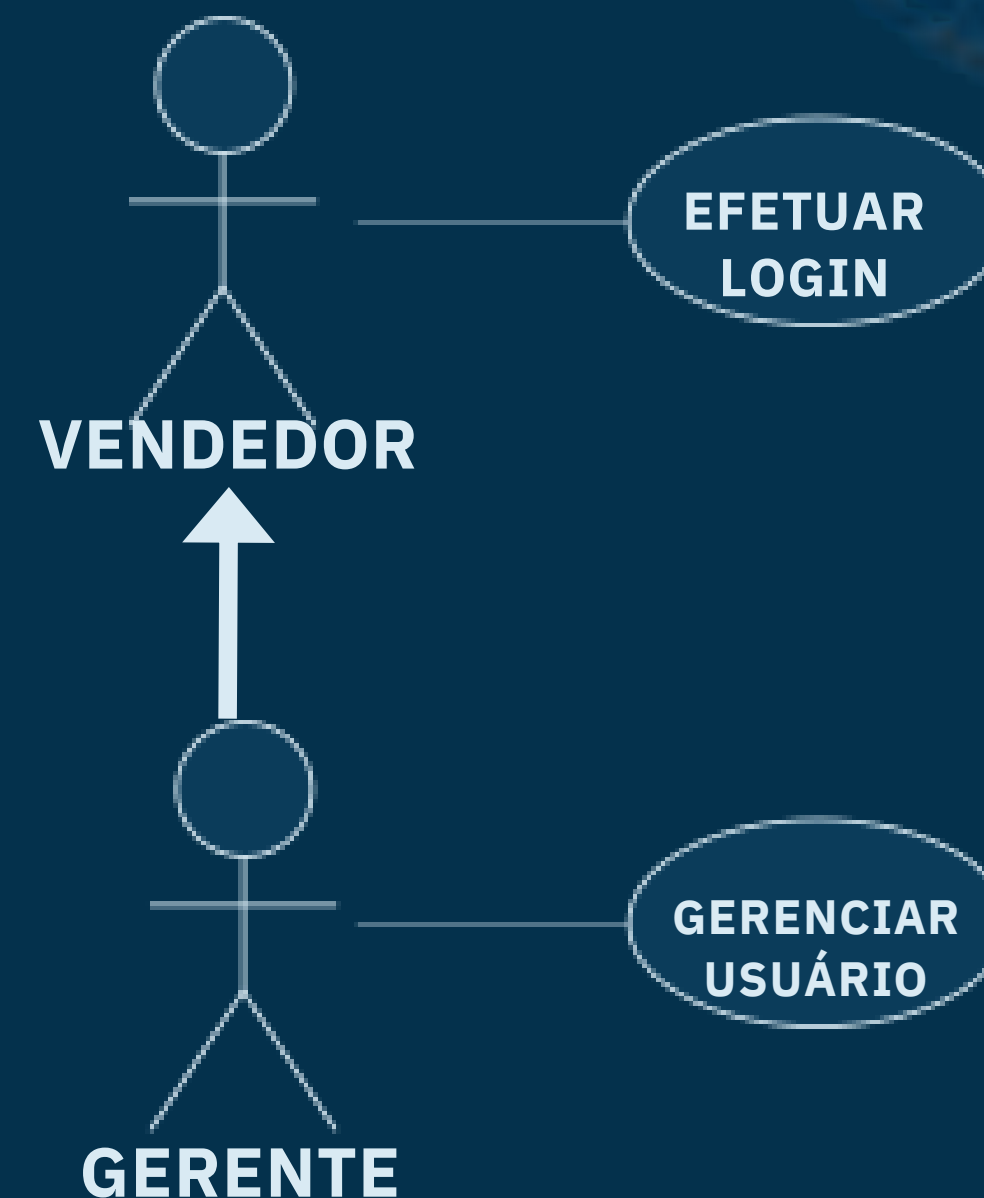
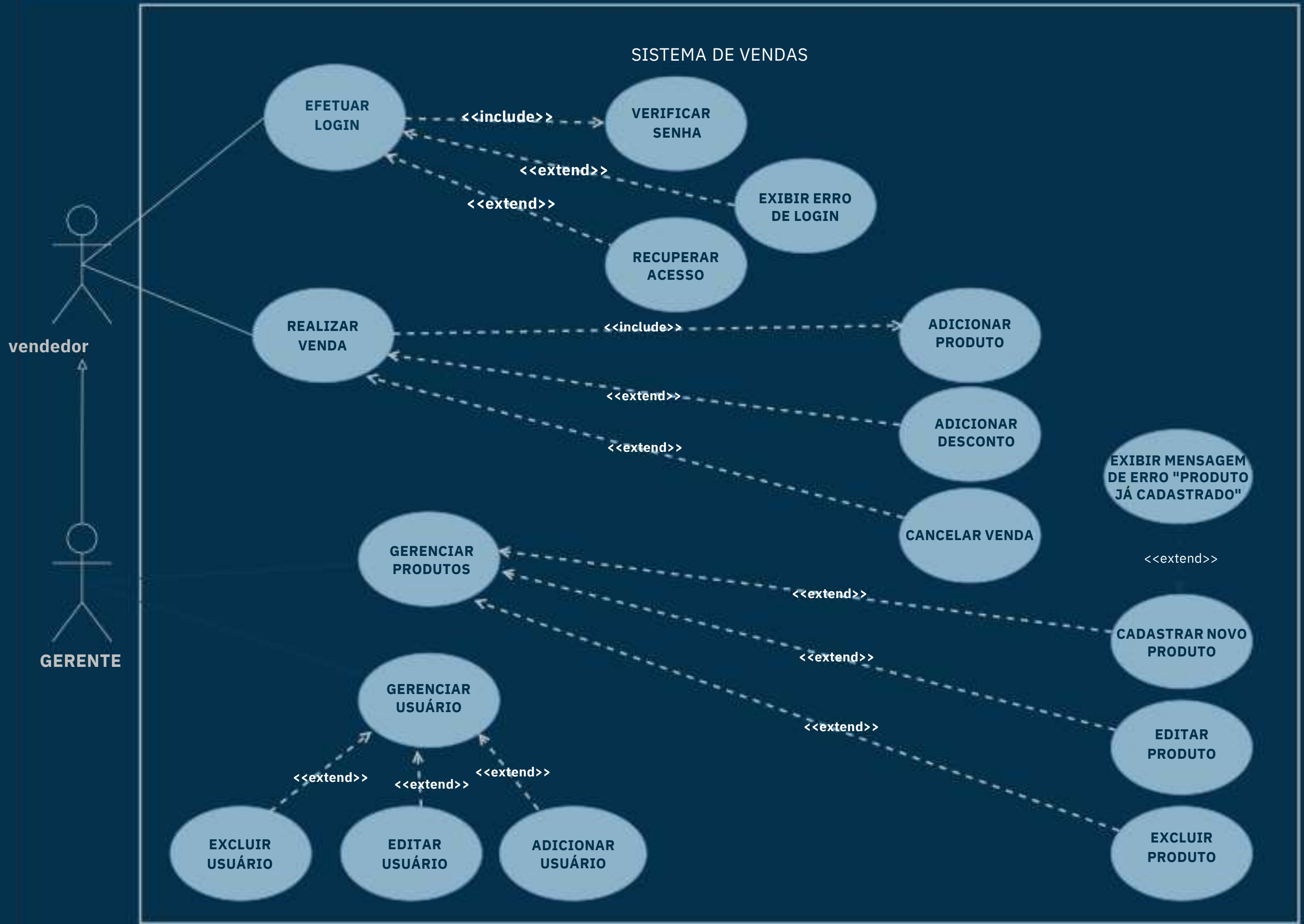


Diagrama de Caso de Uso



09. Modelagem de Classes de Análise

O diagrama de classes, um componente essencial da UML, é um diagrama estrutural que revela a composição interna das entidades envolvidas. É possivelmente o diagrama mais utilizado na UML. (BOOCH, 2006).



09. Modelagem de Classes de Análise

- **Identificador:** valor de uma característica que o identifica para reconhecimento
- **Atributos:** qualidades, características
- **Comportamento:** funções ou procedimentos, os resultados dessas funções determinam o comportamento do objeto

identificador Da Classe

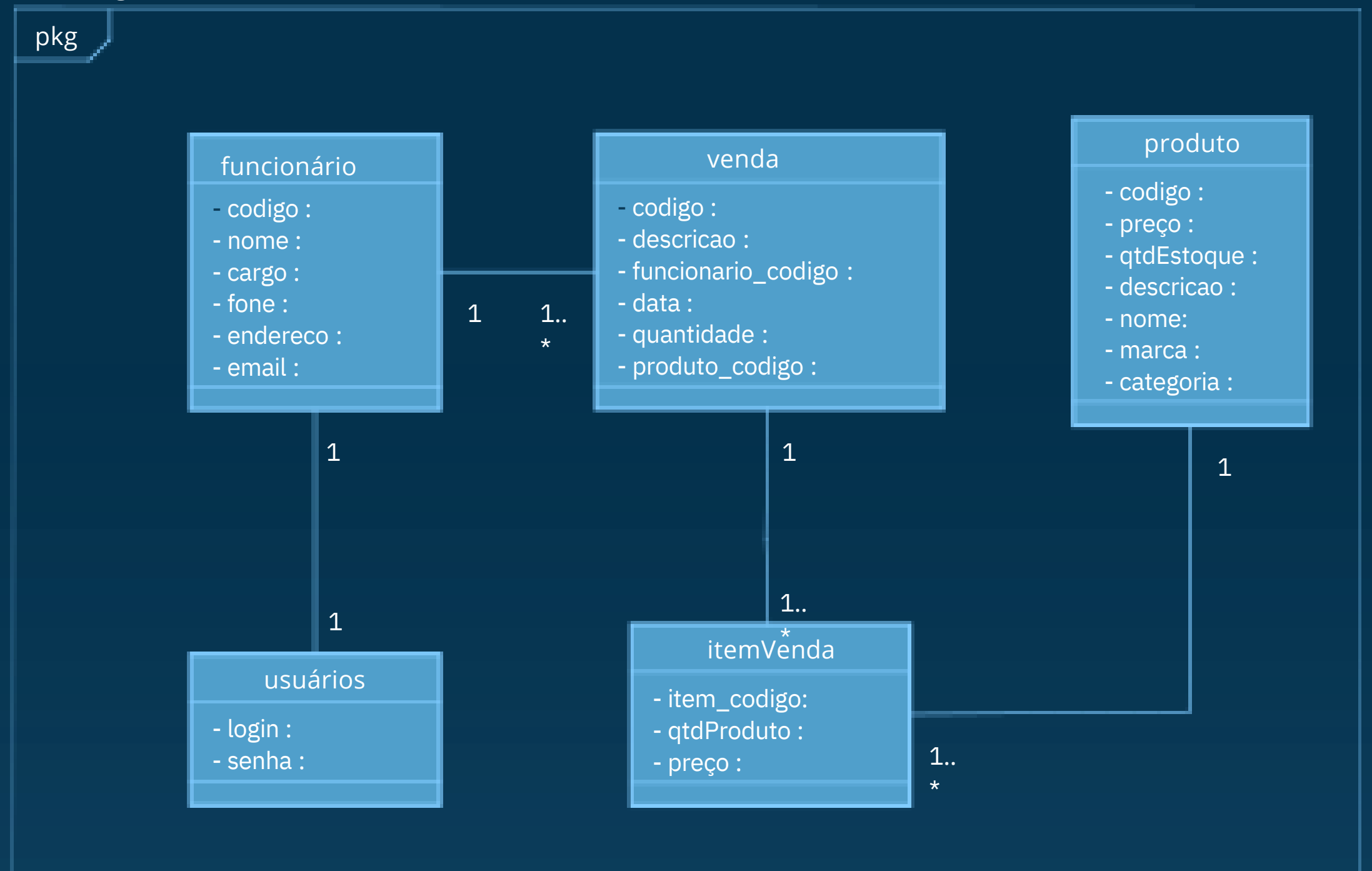
- Atributos : void

+ Métodos() : void

09. Modelagem de Classes de Análise

Nesse modelo, cada classe pode ser vista como um conceito ou um tipo, e os métodos são identificados numa fase posterior.

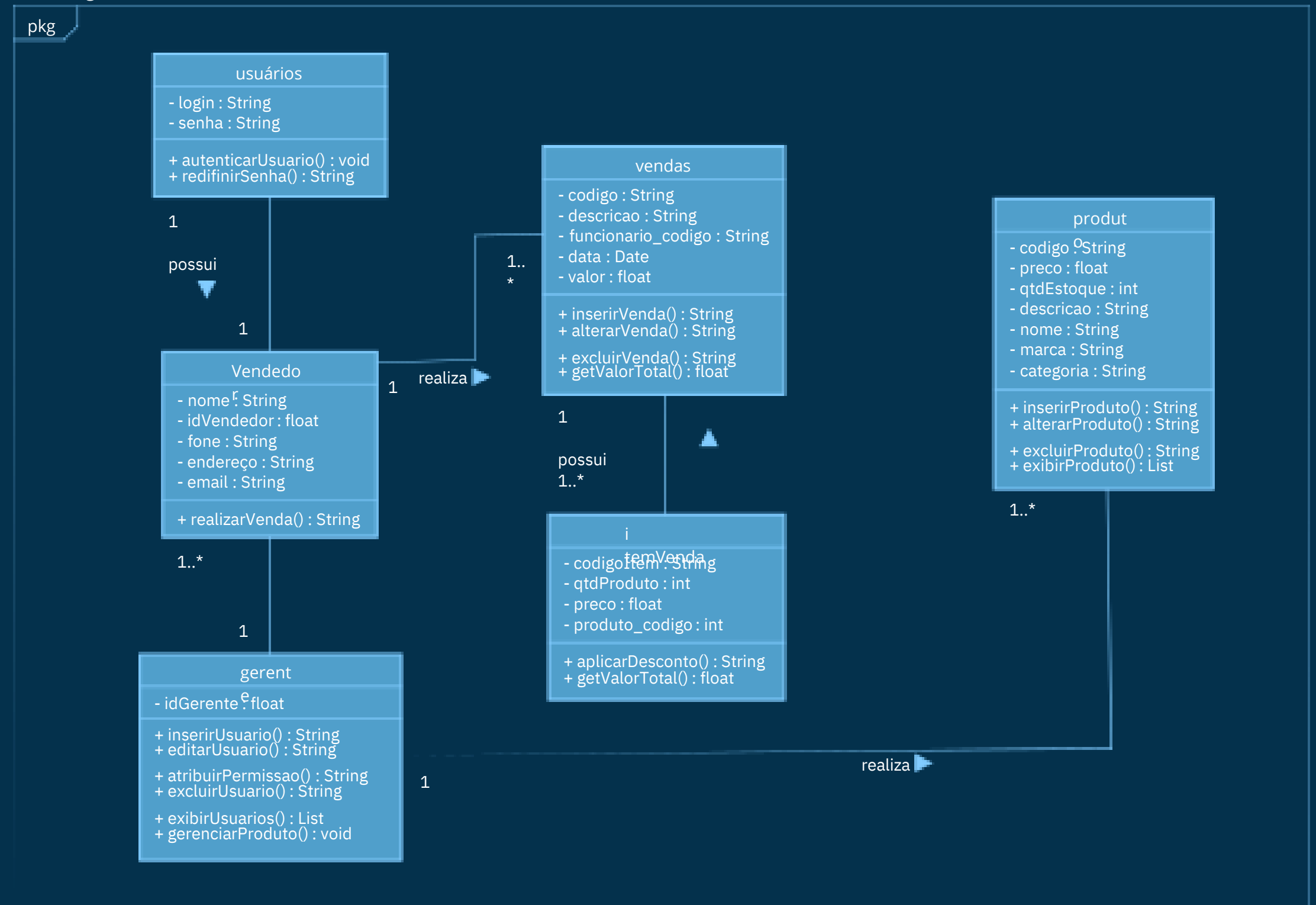
Class Diagram0



Modelagem de Classes de Projeto

Na fase de projeto os métodos são adicionados e o Modelo Conceitual é refinado gerando o Diagrama de Classes

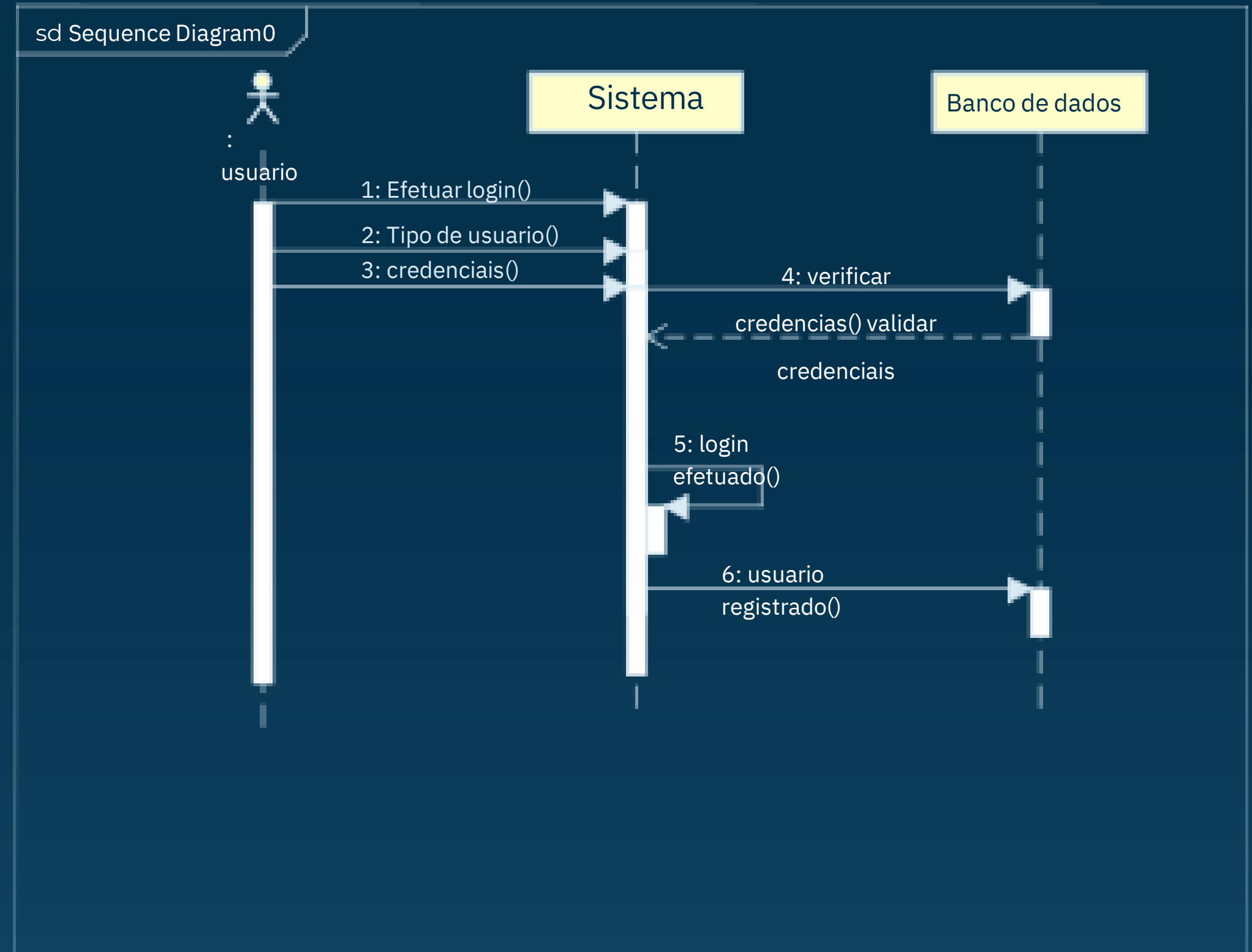
Class Diagram0



Modelagem de Interação e Classes de Projeto

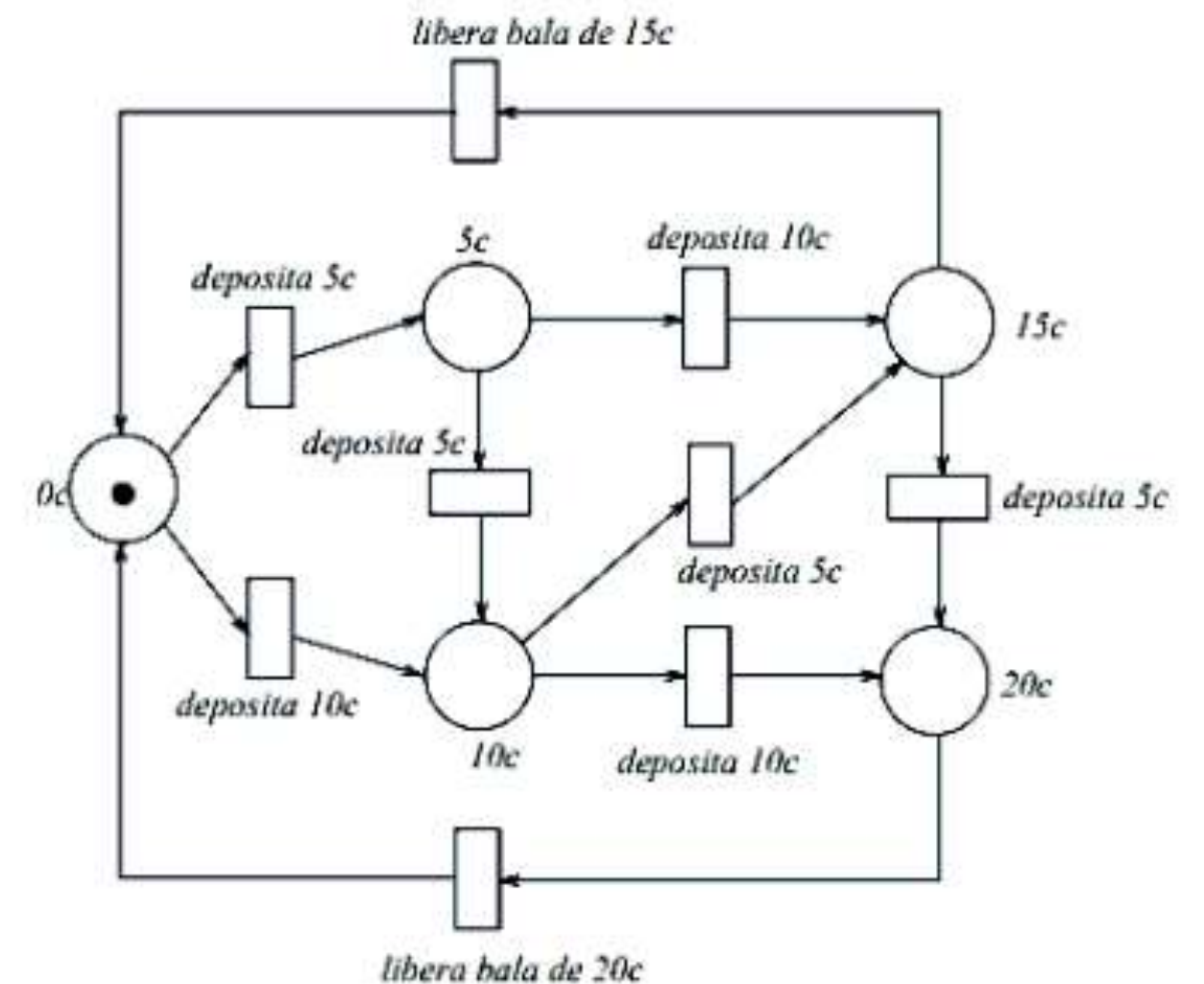
Diagrama de sequência “Efetuar Login”

Sequence Diagram0



Entendendo um exemplo classico de Máquina de Estados

Seguindo os princípios da modelagem de sistemas reativos, conforme proposto por David Harel, uma máquina de estado finito (FSM) é um modelo matemático abstrato que descreve o comportamento dinâmico de um sistema em termos de estados, transições entre esses estados e eventos externos. Essa abstração é frequentemente utilizada na modelagem de sistemas reativos e controlados por eventos" (HAREL, 1987).



Modelagem de Diagramas de Estados

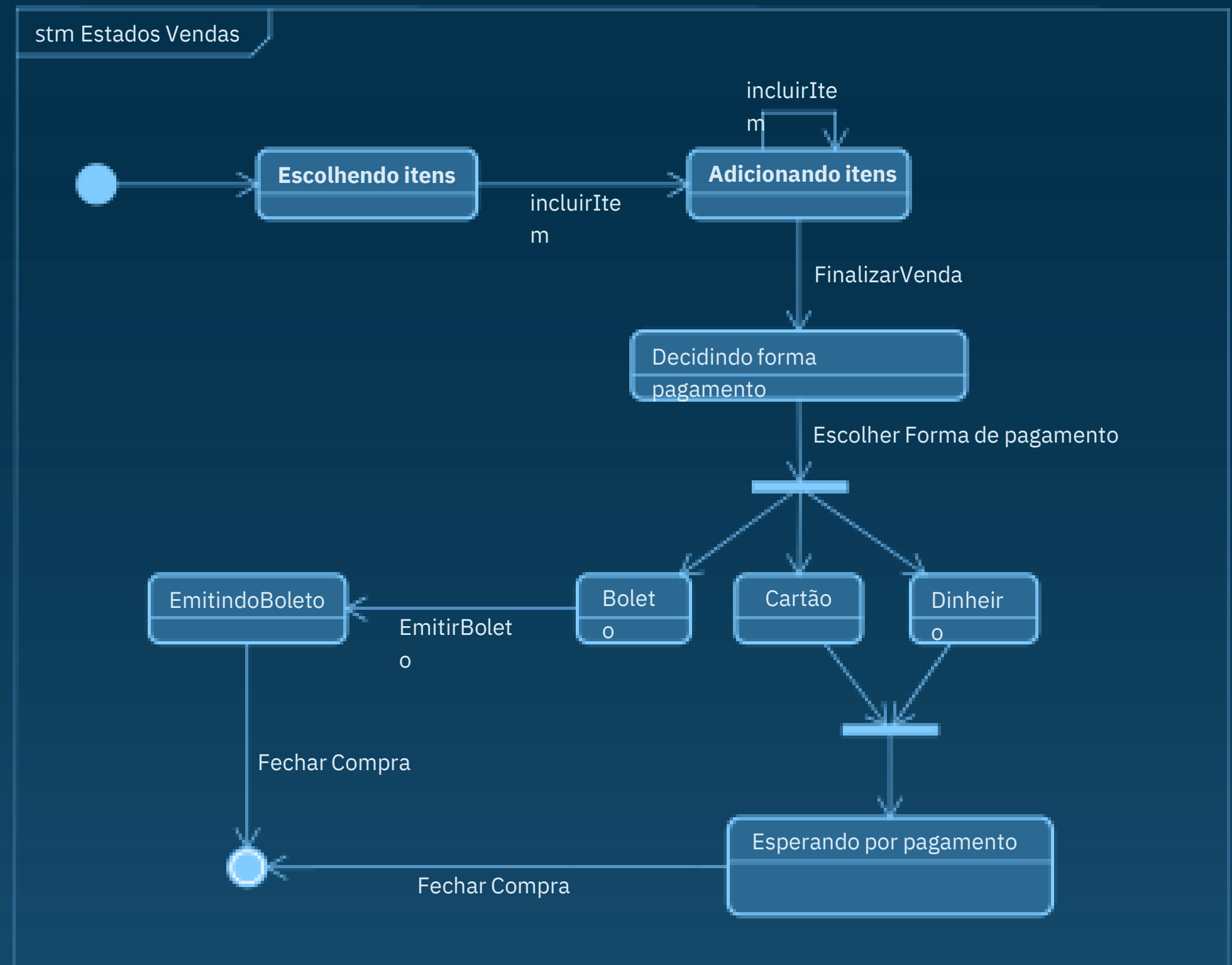
Um diagrama de Estado representa o comportamento dinâmico de um objeto ao longo do tempo. Ele irá mostrar através de diagrama o comportamento do ciclo de vida de um único objeto.

Uma máquina de estado consiste em estados, vinculados por transições. Um estado é uma condição de um objeto em que ele realiza alguma tarefa ou espera um evento. Uma transição é um relacionamento entre dois estados que é disparado por algum evento, que executa determinadas ações ou avaliações e que resulta em um estado final específico. (HAREL, 1987).

Exemplo de Modelagem de estado

Finalizar vendas

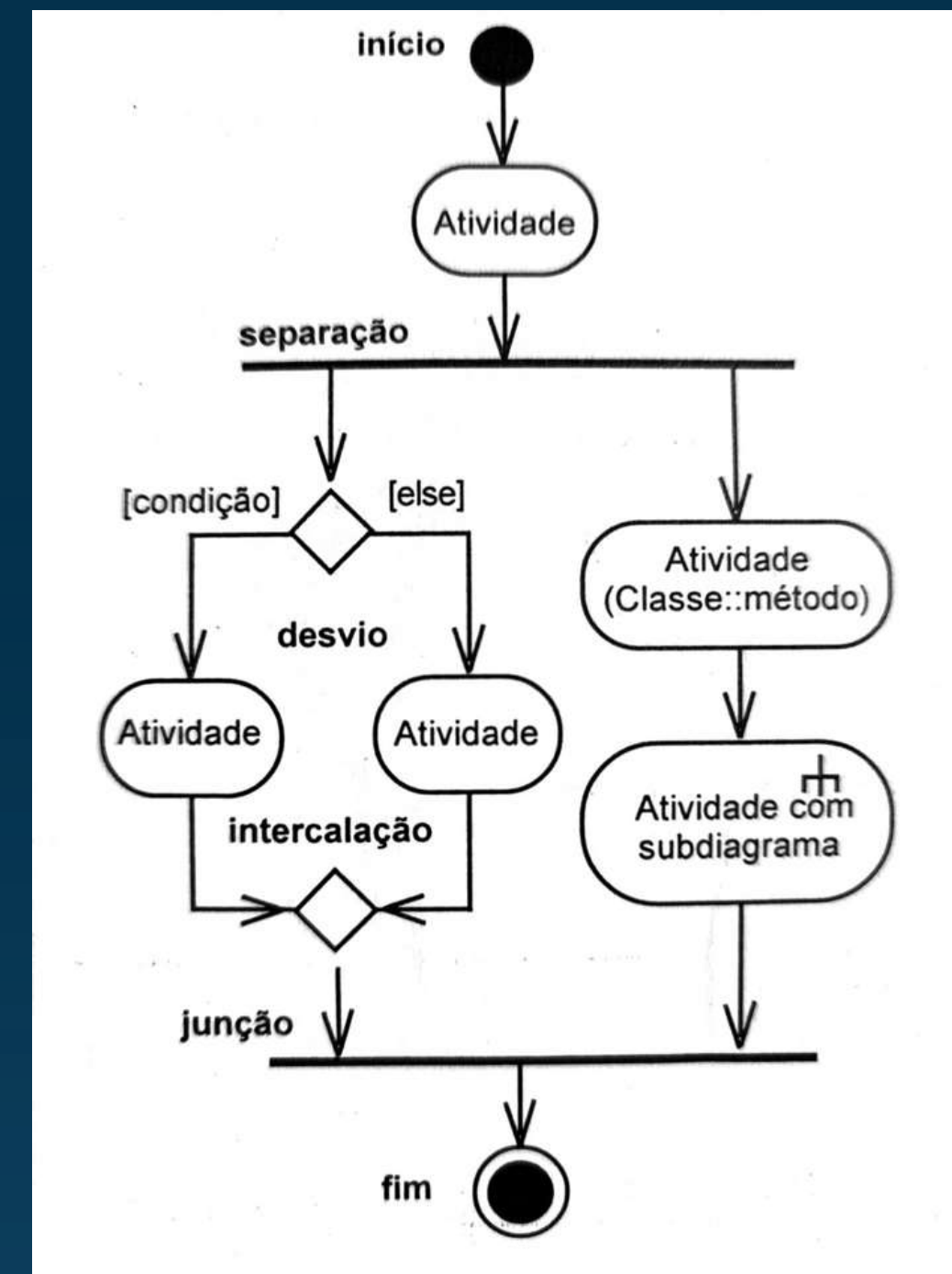
Estados Vendas



O que é o Diagrama de Atividade?

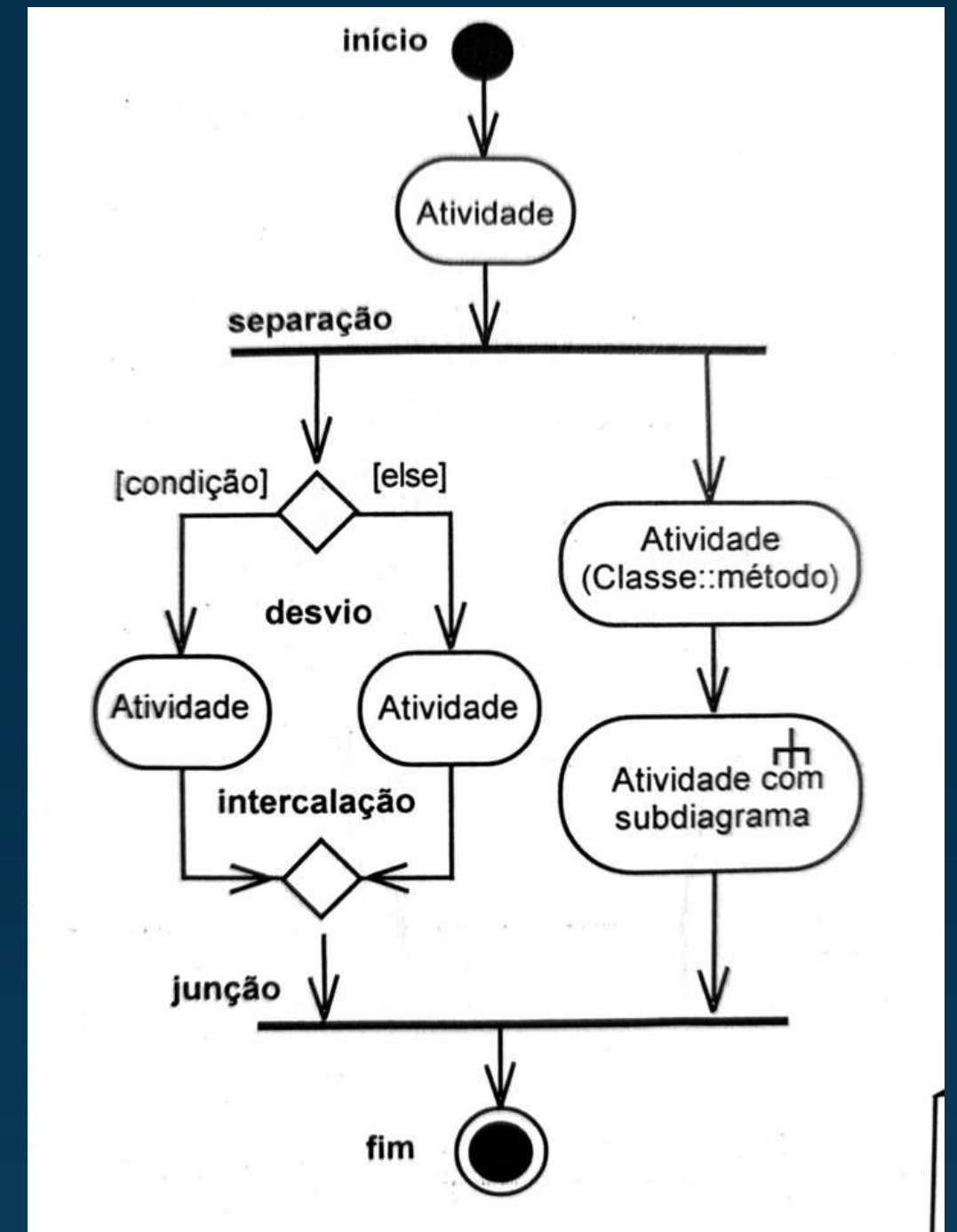
Na UML, um diagrama de atividade fornece uma visualização do comportamento de um sistema descrevendo a seqüência de ações em um processo. Os diagramas de atividades são semelhantes a fluxogramas porque mostram o fluxo entre as ações em uma atividade; no entanto, os diagramas de atividades também podem mostrar fluxos paralelos ou simultâneos e fluxos alternativos.

Nos diagramas de atividades, os nós de atividades e extremidades de atividades são utilizados para modelar o fluxo de controle e os dados entre as ações.



Elementos do Diagrama de Atividade

- Atividade
- fluxo de controle
- Decisões
- Fork e Join
- Estado Inicial/final
- Nodo de ação



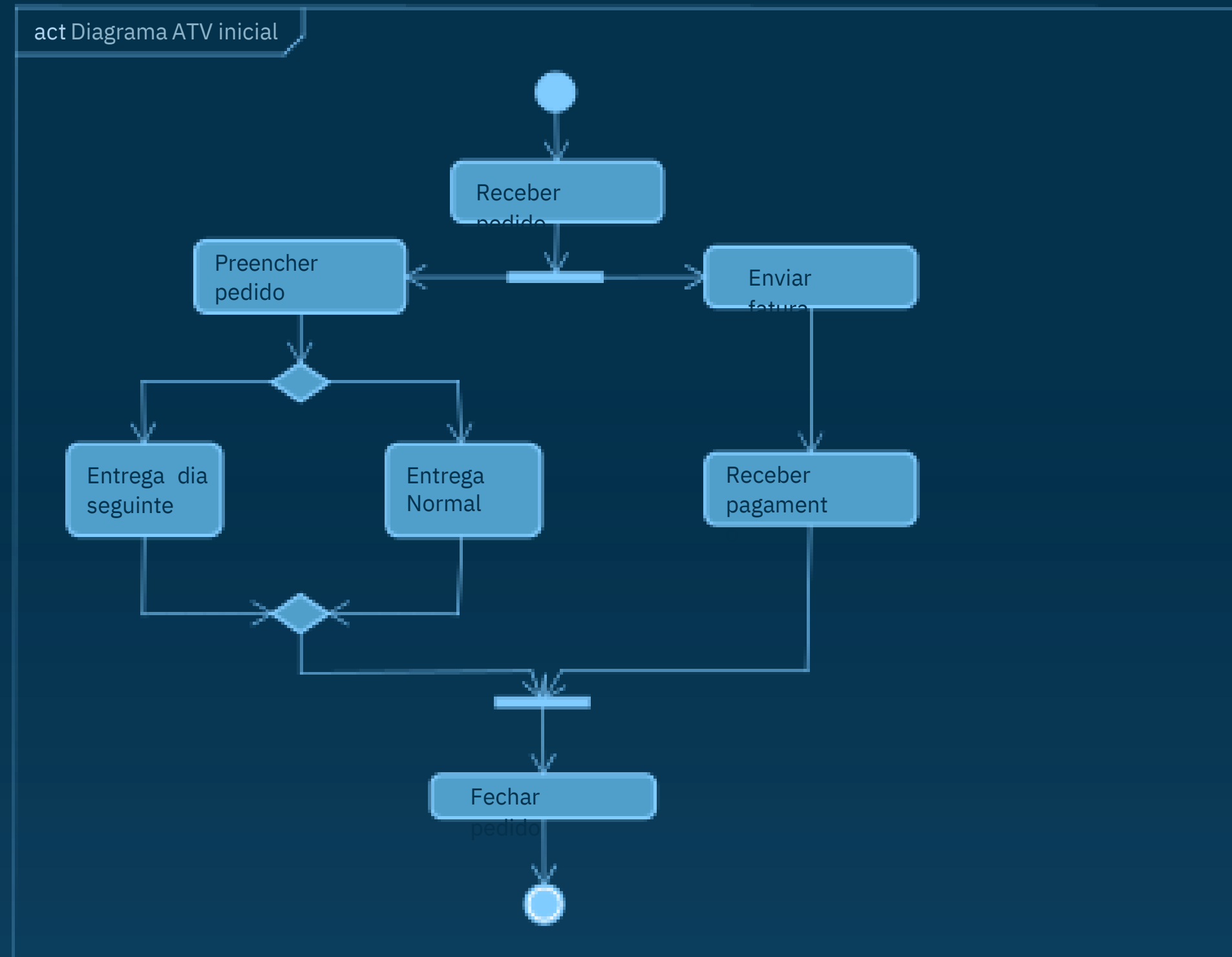
Modelando um Diagrama de Atividade

- Uma atividade se refere a uma sequencia de ações;
- Quando temos um comportamento paralelo, precisamos sincronizar, este elemento é chamado de junção;
- Comportamento condicional é delineado por decisões e intercalações.
- Uma decisão, também chamada de desvio, tem um único fluxo de entrada, e os fluxos de saída vigiados.

Ciclo do pedido

Diagrama de Atividade

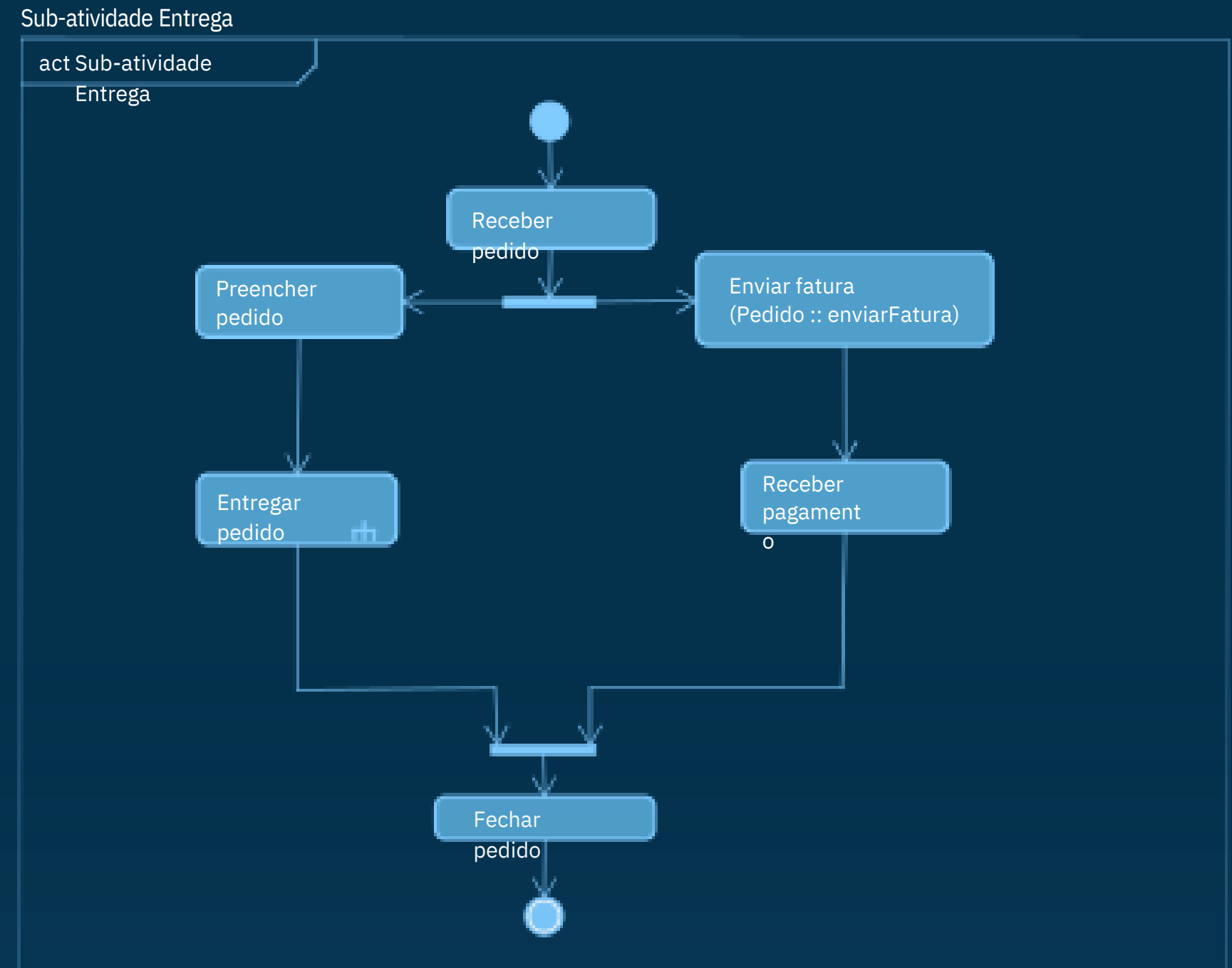
Diagrama ATV inicial



Decomposição de uma ação

Diagrama de Atividade

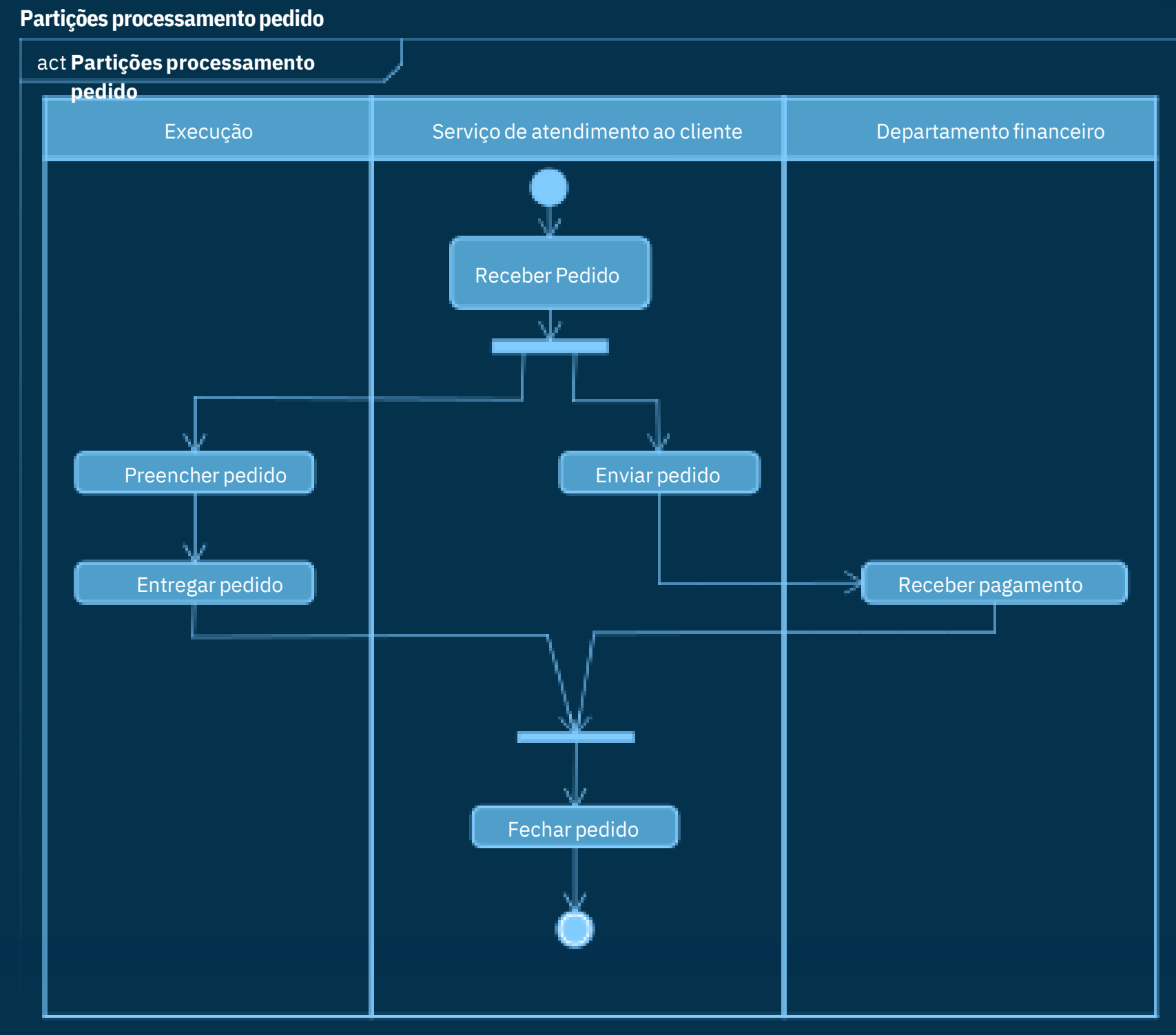
- As ações podem ser decompostas em sub-atividades.
- As sub-atividades podem ser implementadas como métodos.



Quem faz o quê?

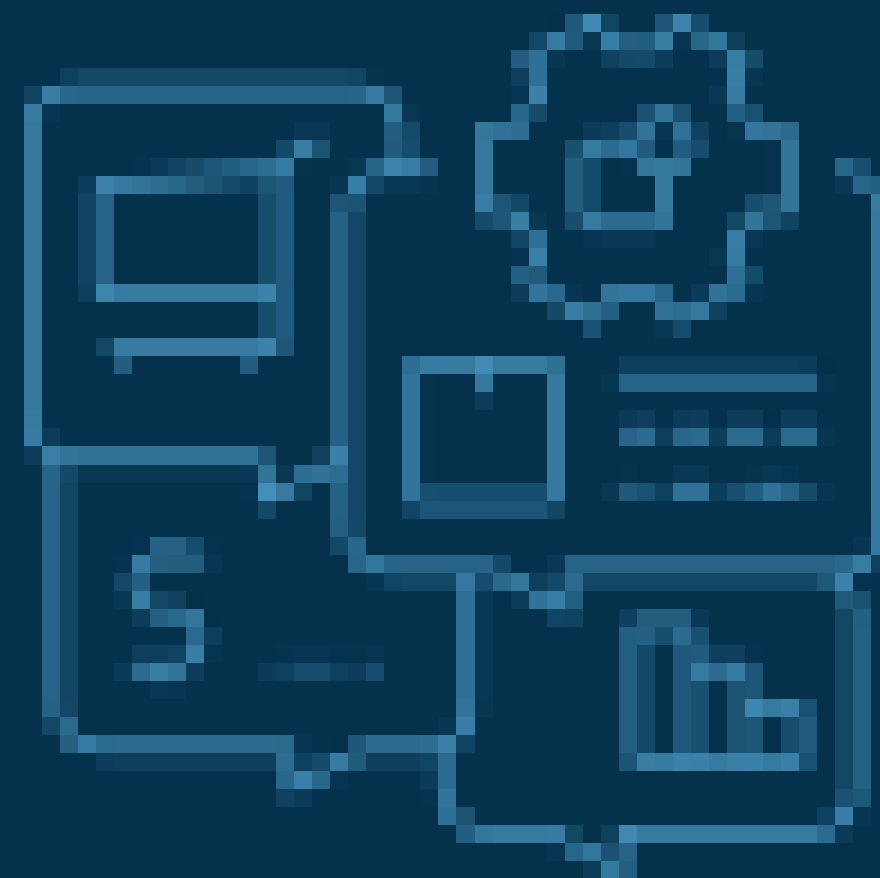
Diagrama de Atividade

No diagrama de atividade até agora vimos as ações acontecerem, mas quem faz o que? Podemos representa-los com partições para entender qual agente está manipulando determinada atividade.



Conclusão

Em conclusão, o projeto de um sistema de vendas com cadastro de funcionários, produtos, e processamento de vendas representa uma solução robusta para gerenciar eficientemente as operações comerciais. O uso de um banco de dados relacional, juntamente com um conjunto abrangente de classes e relacionamentos, proporciona uma estrutura sólida para atender às necessidades do negócio.



Referências

Bibliograficas

ERRA, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML** . 3. ed. Rio de Janeiro: Elsevier, 2015.

BOOCH, Grady , RUMBAUGH, James, JACOBSON, Ivar; **UML: Guia do Usuário**. 2. ed. Campus, 2006

HAREL, David. “**Statecharts: A Visual Formalism for Complex Systems. Science of Computer Programming**”, 8(3): 231-274, June 1987.

KOTLER, Philip . "**Administração de Marketing: Análise, Planejamento, Implementação e Controle**.5. ed. Atlas, 1998.

Kerzner, H. "**Gestão de Projetos: Uma Abordagem Sistêmica para o Planejamento, Programação e Controle**". 2. ed. Editora Blucher, 2021.

PFLEEGER, Shari Lawrence. “**Engenharia de Software: teoria e prática; tradução Dino Franklin; revisão técnica Ana Regina Cavalcanti da Rocha**”. – 2. Ed. – São Paulo – Prentice Hall, 2004.

PRESSMAN, Roger S. **Engenharia de Software**. 6 ed. São Paulo: McGraw Hill/Nacional, 2006.

ROSCA D. GREENSPAN S., FEBLOWITZ M., WILD C. (1997). A decision Making Methodology in support of business rules Lifecycle. In Proceeding of RE 97: IEEE International Symposuim on Requeriments Engineering, IEEE Computer Society Press, p. 236 -246

SOMMERVILLE, Ian. Engenharia de Software. 8 ed. Rio de Janeiro: Prentice-Hall, 2008



Thank You

Agradecemos pela atenção !

Reconhecimentos e Direitos Autorais

Autores: ANTONIO LISTER, GABRIEL FERNANDO, KELY SOUZA, LUCAS TEIXEIRA, VICTOR COELHO.

contato: lister.wd@hotmail.com ; gabrielfernandocarneiro798@gmail.com; kely.souza@discente.ufma.br ;lucas.teixeira@discente.ufma.br

data última versão: 04/12/2023

versão: 1.0

outros repositórios: <https://github.com/kelyazuos?tab=repositories>

Agradecimentos: Universidade Federal do Maranhão (UFMA), Professor Doutor Thales Levi Azevedo Valente, e colegas de curso.

Copyright/License

Este material é resultado de um trabalho acadêmico para a disciplina PROJETO E DESENVOLVIMENTO DE SOFTWARE, sobre a orientação do professor Dr. THALES LEVI AZEVEDO VALENTE, semestre letivo 2023.2, curso Engenharia da Computação, na Universidade Federal do Maranhão (UFMA). Todo o material sob esta licença é software livre: pode ser usado para fins acadêmicos e comerciais sem nenhum custo. Não há papelada, nem royalties, nem restrições de "copyleft" do tipo GNU. Ele é licenciado sob os termos da licença MIT reproduzida abaixo e, portanto, é compatível com GPL e também se qualifica como software de código aberto. É de domínio público. Os detalhes legais estão abaixo. O espírito desta licença é que você é livre para usar este material para qualquer finalidade, sem nenhum custo. O único requisito é que, se você usá-los, nos dê crédito.

Copyright © 2023 Educational Material.

Este material está licenciado sob a Licença MIT. É permitido o uso, cópia, modificação, e distribuição deste material para qualquer fim, desde que acompanhado deste aviso de direitos autorais.

O MATERIAL É FORNECIDO "COMO ESTÁ", SEM GARANTIA DE QUALQUER TIPO, EXPRESSA OU IMPLÍCITA, INCLUINDO, MAS NÃO SE LIMITANDO ÀS GARANTIAS DE COMERCIALIZAÇÃO, ADEQUAÇÃO A UM DETERMINADO FIM E NÃO VIOLAÇÃO. EM HIPÓTESE ALGUMA OS AUTORES OU DETENTORES DE DIREITOS AUTORAIS SERÃO RESPONSÁVEIS POR QUALQUER RECLAMAÇÃO, DANOS OU OUTRA RESPONSABILIDADE, SEJA EM UMA AÇÃO DE CONTRATO, ATO ILÍCITO OU DE OUTRA FORMA, DECORRENTE DE, OU EM CONEXÃO COM O MATERIAL OU O USO OU OUTRAS NEGOCIAÇÕES NO MATERIAL.