

UML - Completo

Teoria e Exercícios

Professor Gabriel Pacheco

professor.gabrielpacheco@gmail.com

<https://www.instagram.com/professor.gabrielpacheco/>

<https://www.youtube.com/profgabrielpacheco>

<https://www.facebook.com/coachgabrielpacheco>

Outros Cursos no PTI:

<https://bit.ly/2mIWMX4>

[Professor]



Master Trainer/Facilitador de
Treinamentos

Especialista em Gerenciamento
de Projetos e Métodos Ágeis

Certificações na Área de Projetos
e Agilidade

Analista Comportamental DISC

Coach e Consultor das Áreas de
Concursos Públicos e
Certificações

UML

Conceitos



Grady BOOCH
James RUMBAUGH
Ivar JACOBSON

UML

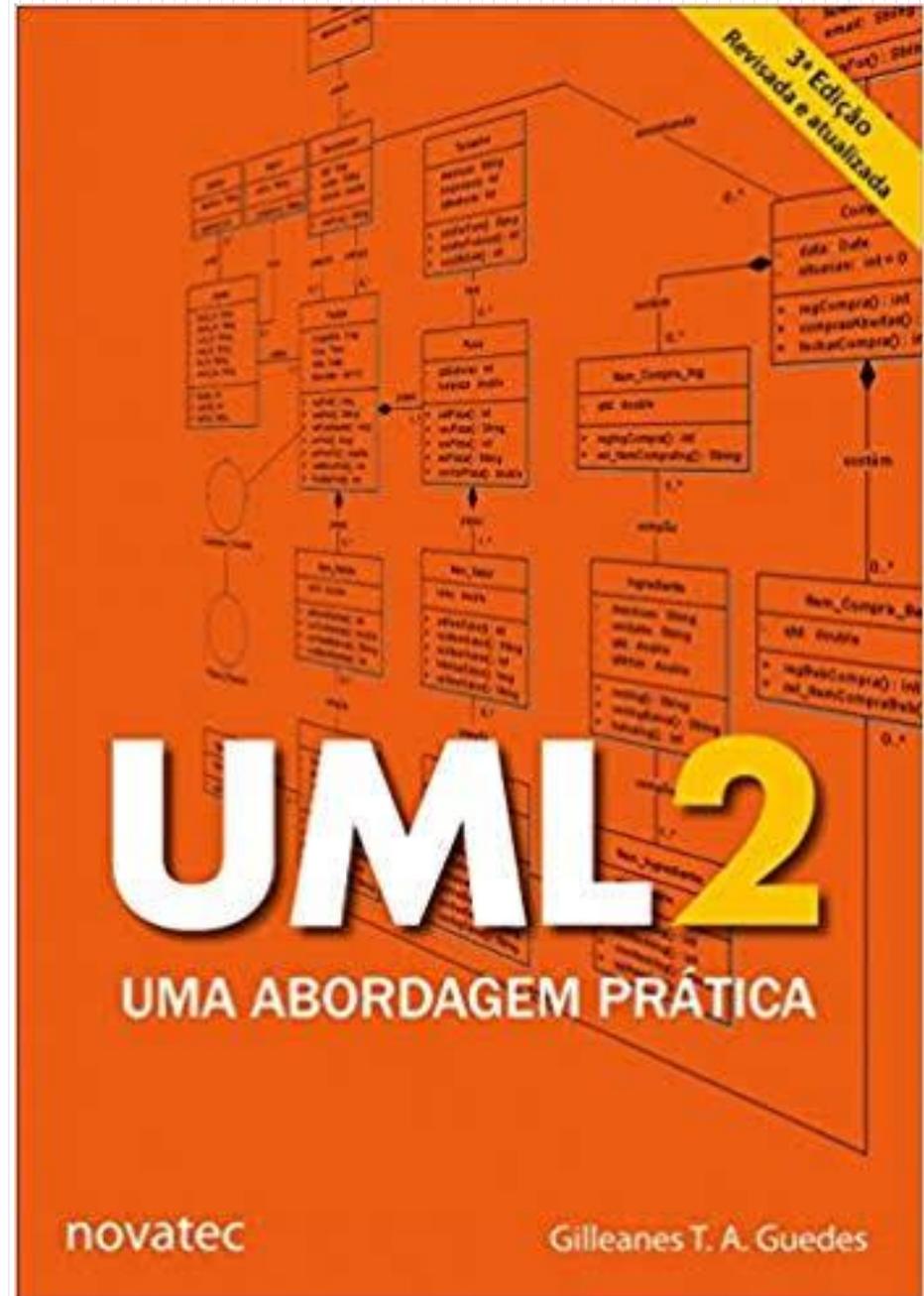
GUIA DO USUÁRIO

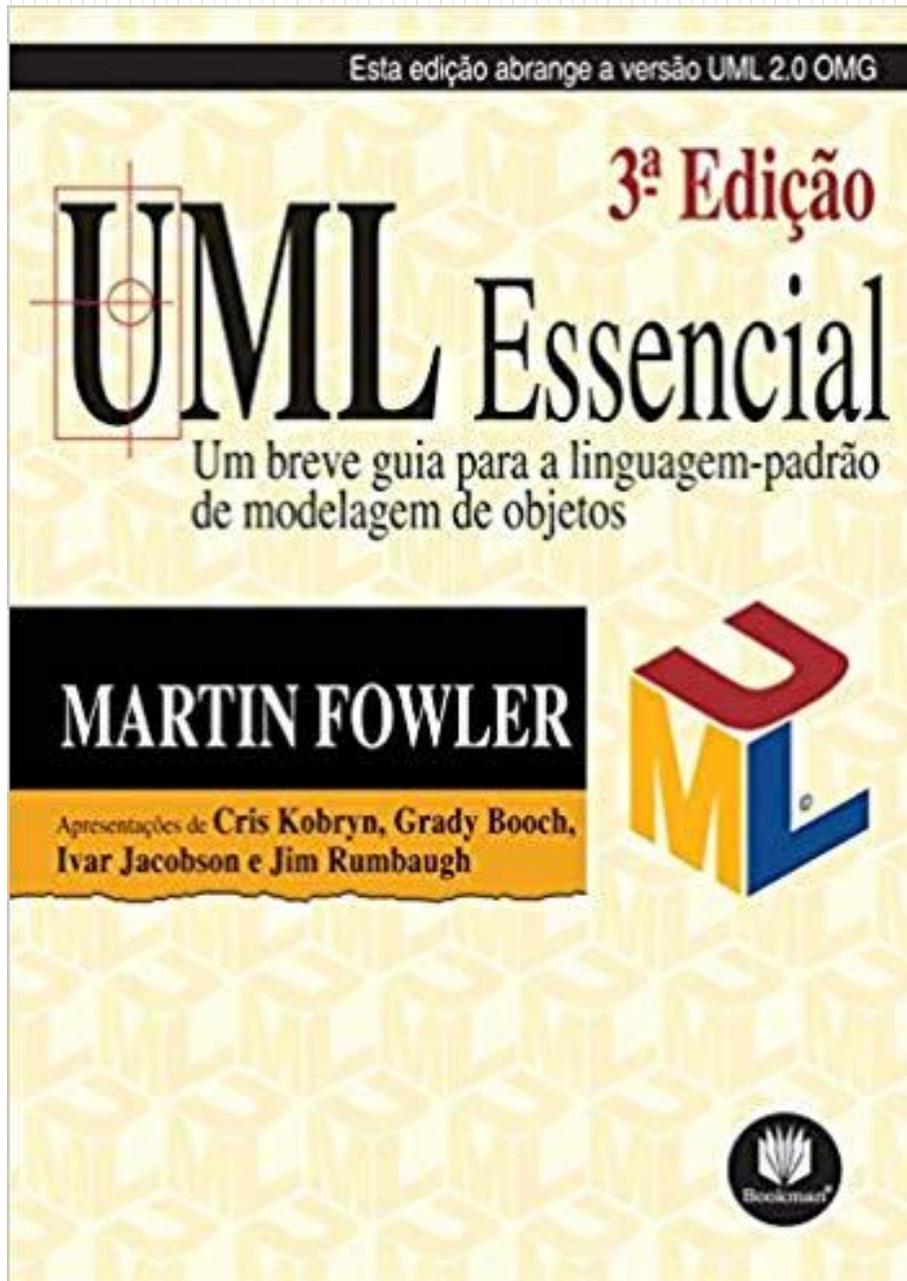
Segunda Edição, totalmente revista e atualizada

O mais avançado tutorial sobre Unified Modeling Language (UML), elaborado pelos próprios criadores da linguagem



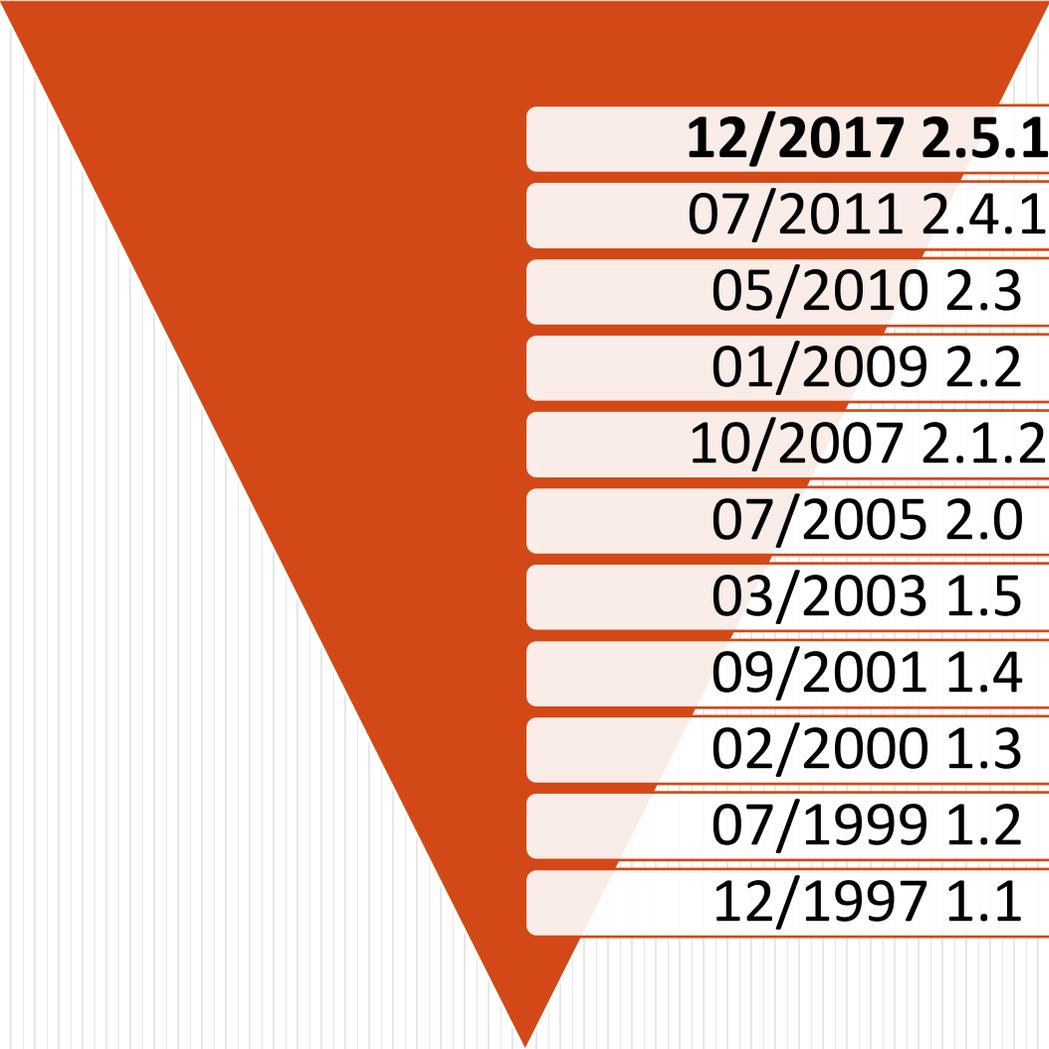
Abrange
UML 2.0





- <https://www.omg.org/spec/UML>

Histórico de versões oficiais



12/2017 2.5.1
07/2011 2.4.1
05/2010 2.3
01/2009 2.2
10/2007 2.1.2
07/2005 2.0
03/2003 1.5
09/2001 1.4
02/2000 1.3
07/1999 1.2
12/1997 1.1

Introdução à UML

- UML não é um processo de desenvolvimento de software e também não está ligada a um de forma exclusiva.
- Independente, pode ser utilizada por diversos processos de desenvolvimento diferentes ou mesmo da forma que o engenheiro de software considerar adequada.
- “Linguagem visual utilizada para modelar softwares baseados no paradigma de orientação a objetos. É uma linguagem de modelagem de propósito geral que pode ser aplicada a todos os domínios de aplicação.”
[Gilleanes]

🌐 Surgiu da união de três métodos de modelagem:

- 🌐 Método de Booch.
- 🌐 Método OMT (Object Modeling Technique) de Jacobson. E
- 🌐 Método OOSE (Object-Oriented Software Engineering) de Rumbaugh.
- 🌐 Eram os mais populares até meados de 90.
- 🌐 O trabalho dos “Três amigos” resultou no lançamento, em 1996, da primeira versão do UML propriamente dita.

- 🌐 Por que modelar software? (Para que projetar uma casa?). 😊
- 🌐 Comumente os sistemas de informação estão em constante evolução, pois:
 - 🌐 Clientes demandam modificações ou melhorias.
 - 🌐 O mercado está sempre mudando.
 - 🌐 O governo promulga novas leis e cria novos impostos e alíquotas.
- 🌐 Um sistema de informação precisa ter uma documentação extremamente detalhada, precisa e atualizada para que assim possa ser mantido.

 **Modelo de Software:** a modelagem implica diretamente em criar modelos de software. Um modelo de software captura uma visão de um sistema físico, é uma abstração do sistema com propósito de descrever aspectos estruturais ou comportamentais do software (o citado propósito irá delimitar as fronteiras do modelo).

Documentação Histórica.

-  Perspectiva histórica.

-  Permite à empresa:

 -  Saber se está evoluindo.

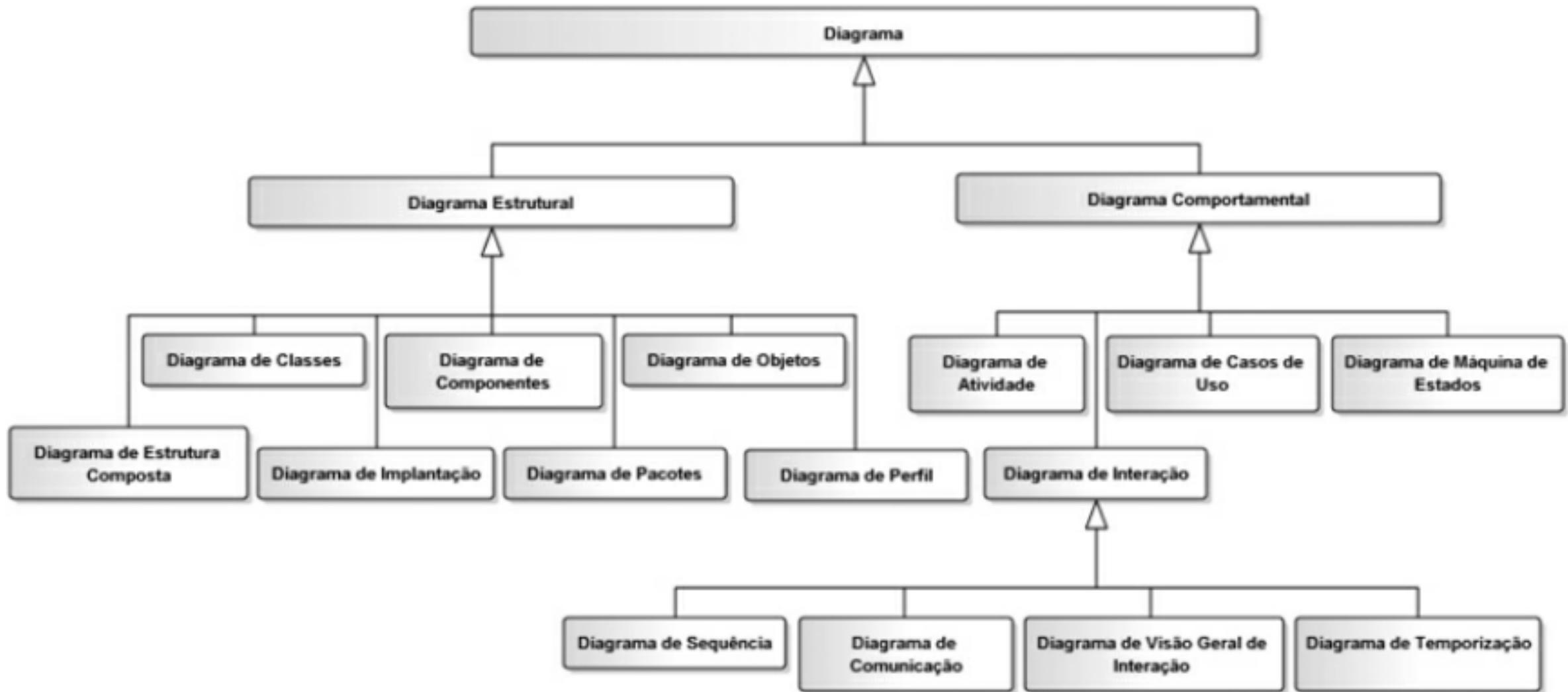
 -  Saber se o processo de desenvolvimento tornou-se mais rápido.

 -  Saber se as metodologias adotadas atualmente são superiores às práticas aplicadas anteriormente.

 -  Se a qualidade do software está melhorando.

🌐 **Documentação Histórica. (cont)**

- 🌐 A documentação detalhada do software se torna necessária que se tenha:
 - 🌐 Média de manutenções.
 - 🌐 Média de custo de modelagem.
 - 🌐 Média de custo de desenvolvimento.
 - 🌐 Média de tempo despendido até a finalização do projeto.
 - 🌐 Quantidade de profissionais necessários.
- 🌐 Torna-se muito útil na Reusabilidade.



[Questão]

(IF-PB – Professor – Informática – 2019 - IDECAN)

A UML (Unified Modeling Language) define dois tipos principais de diagramas: estruturais e comportamentais. Qual das opções abaixo lista apenas diagramas comportamentais da UML?

- A) Máquina de Estados; Atividades; Casos de Uso.
- B) Classes; Sequência; Comunicações.
- C) Tempo; Objetos; Pacotes.
- D) Sequência; Tempo; Perfil.
- E) Atividades; Casos de Uso; Classes.

[Questão]

(UFF - Analista de TI– 2019 – Coseac)

Na UML 2.0 existem os diagramas estruturais e os diagramas comportamentais. É um exemplo de diagrama estrutural o diagrama de:

- A) caso de uso.
- B) atividade.
- C) objetos.
- D) tempo.
- E) comunicação.

[Questão]

(BNB – Especialista Técnico – Analista de Sistema – 2018 – Cespe)

Considerando os conceitos de análise e projeto orientados a objetos, julgue o item subsecutivo.

A UML é um guia para análise e projeto orientados a objetos, e mostra ao desenvolvedor como realizar essas atividades.

UML

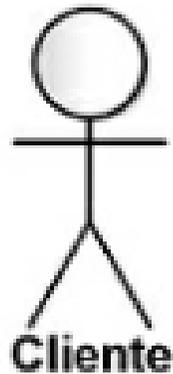
Diagrama de Casos de Uso

Diagrama de Caso de Uso.

- 🌀 Procura possibilitar a compreensão do comportamento externo do sistema por qualquer pessoa em uma perspectiva do usuário. Utilizado no início da modelagem principalmente nas etapas de levantamento de requisitos de forma abstrata, flexível e informal. Será consultado durante todo o processo de engenharia e também poderá ser modificado. Não se preocupa com o “como” as funcionalidades serão implementadas.
- 🌀 Muito útil para entendimento dos requisitos funcionais do sistema.
- 🌀 Poderá ser apresentado em reuniões iniciais com o cliente para ilustração do comportamento do sistema juntamente com o protótipo.
- 🌀 Composto de:
 - 🌀 Atores.
 - 🌀 Casos de Uso.

🌐 Atores.

- 🌐 Representam os papéis desempenhados pelos diversos usuários que poderão utilizar os serviços e funções do sistema. Pode ser um hardware, outro sistema ou uma pessoa representado por uma breve descrição logo abaixo dele.
- 🌐 Indentificados inicialmente como sendo as entidades externas



🌐 Casos de uso.

- 🌐 Devem capturar os requisitos funcionais do sistema (serviços, tarefas ou funcionalidades) que serão utilizados pelos atores.



Casos de uso (cont).

 Podem ser:

 **Primário:** se estiver relacionado a um requisito funcional do software.

 **Secundário:** se refere a um processo periférico (manutenção de cadastro).

 São representados por elipses contendo um texto sucinto sobre sua descrição.

 Eles terão o seu funcionamento documentado de maneira informal possibilitando identificar:

 Qual evento forçará sua execução.

 Quais atores poderão utilizá-los.

 Quais são suas possíveis restrições.

Casos de uso (cont).

-  Seu objetivo principal é fornecer um relatório ao cliente sobre o comportamento pretendido para o UC e quais funções ele executará.
-  São identificados pela determinação das funcionalidades necessárias ao software.

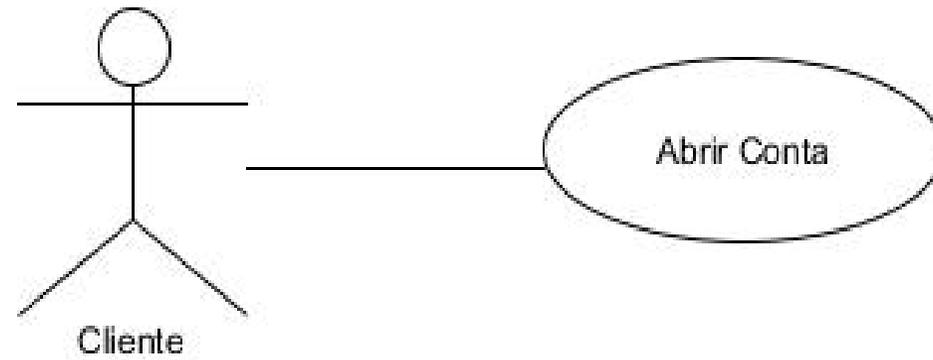
Documentação do UC.

-  Não existe um formato específico, mas indica-se que não sejam utilizados pseudocódigos na documentação.

Nome	Abrir Conta
Ator principal	Cliente
Atores secundários	Funcionário
Resumo	Descreve os passos a serem percorridos pelo cliente para a abertura de uma conta.
Pré-condições	Pedido de abertura de conta previamente aprovado.
Pós-condições	Realização de um depósito inicial.
Fluxo Principal.	Ações do Sistema.
Solicitar Abertura de Conta	
	Consultar cliente por seu CPF
Informar a senha da conta	
	Abrir a conta
Restrições	O cliente precisa ser maior de idade para abrir uma conta.
	O valor mínimo do depósito deverá ser de R\$ 500,00.
	O cliente precisa apresentar documentação própria no ato da abertura.
Fluxo Alternativo	(representam situações que fogem das situações ideais do caso de uso ou que representem opções que podem ser executadas ou não)
Fluxo de Exceção	(ações que deverão ser tomadas em situações em que o fluxo não possa ser concluído)

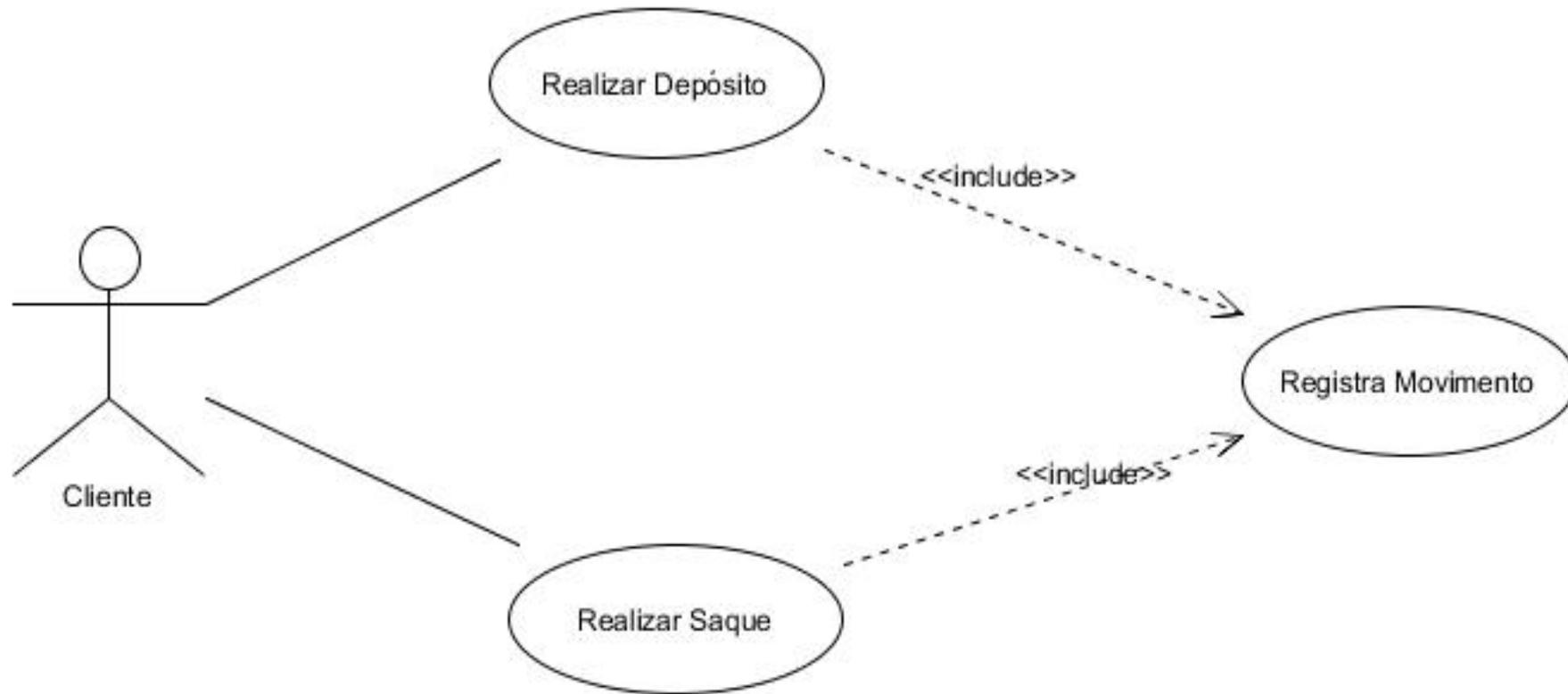
Associações.

-  Interações ou relacionamentos entre os atores, entre os atores e os UCs ou entre os UCs e outros UCs.
-  Podem ser de:
 -  Inclusão.
 -  Extensão.
 -  Generalização.
-  Quando entre um ator e um UC, será representada por uma linha ligando o ator ao caso de uso, podendo ocorrer na extremidade uma seta indicando o sentido em que as informações trafegam ou para indicarem quem inicia a comunicação.
-  Poderá ter uma descrição própria.



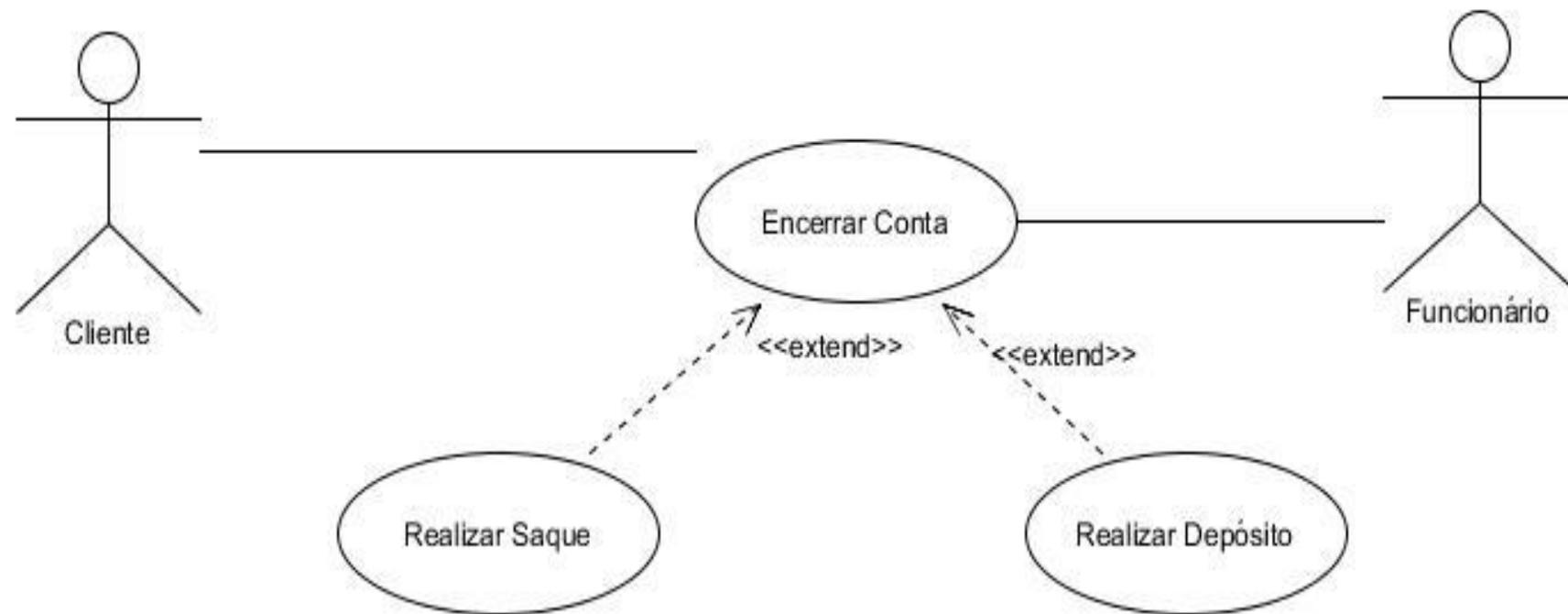
Inclusão.

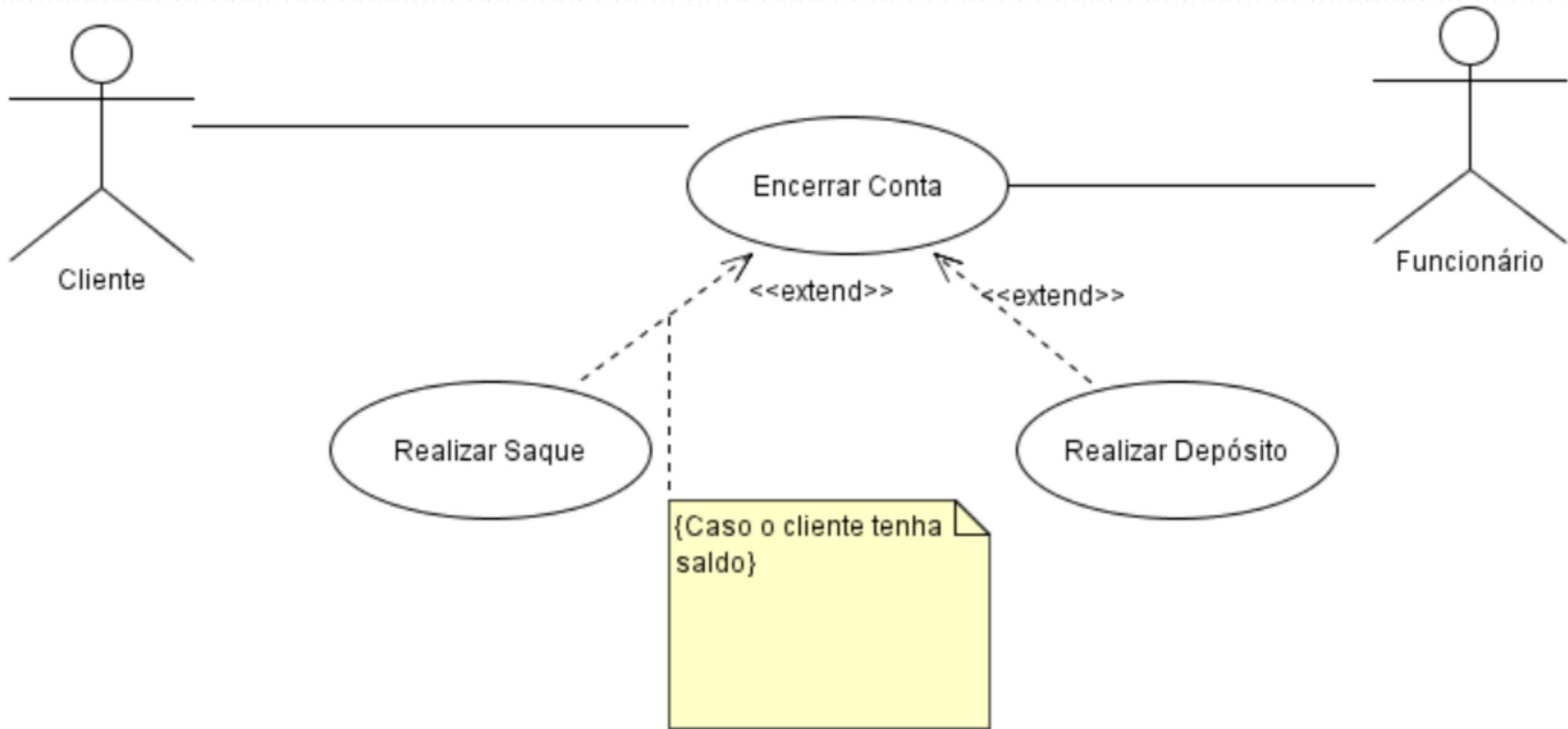
-  Utilizado quando temos situações ou cenários comuns a mais de um UC, onde sua documentação será colocada em um UC que será utilizada por outros UCs.
-  Este tipo de associação cria uma obrigação, na qual a execução do primeiro irá obrigar a execução do segundo.
-  Representada por uma linha tracejada contendo uma seta em uma de suas extremidades que irá apontar para o UC incluído.
-  Costumam apresentar um estereótipo com o texto include entre “<<>>”.



Extensão.

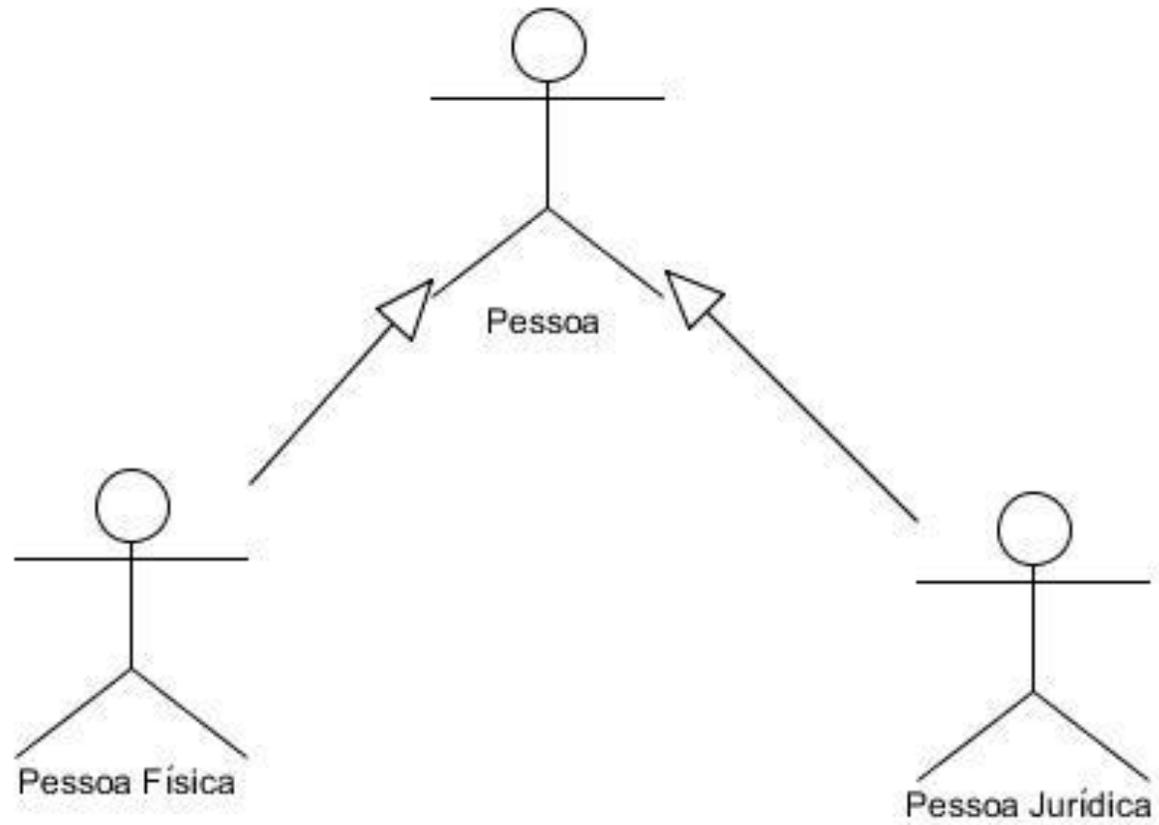
-  Utilizada para descrever cenários opcionais de um UC, descrevem cenários que apenas ocorrerão em uma situação específica se determinada condição for satisfeita condicionada a um teste.
-  Representada por uma linha tracejada, mas com a seta apontando para o UC que utiliza o UC estendido.
-  Costuma apresentar um estereótipo com o texto *extend* entre “<<>>”.





Generalizações/Especializações.

-  Utilização de um UC geral que descreve as características compartilhadas por todos os UCs em questão, a documentação destes últimos conterá somente as características específicas de cada um, visto que os casos de uso especializados irão herdar toda a estrutura do caso de uso generalizado.
-  Representada por uma linha com uma seta mais grossa, que indicará qual é o caso de uso mais geral (do lado da seta).
-  Tal tipo de relacionamento também pode ser aplicado em atores.

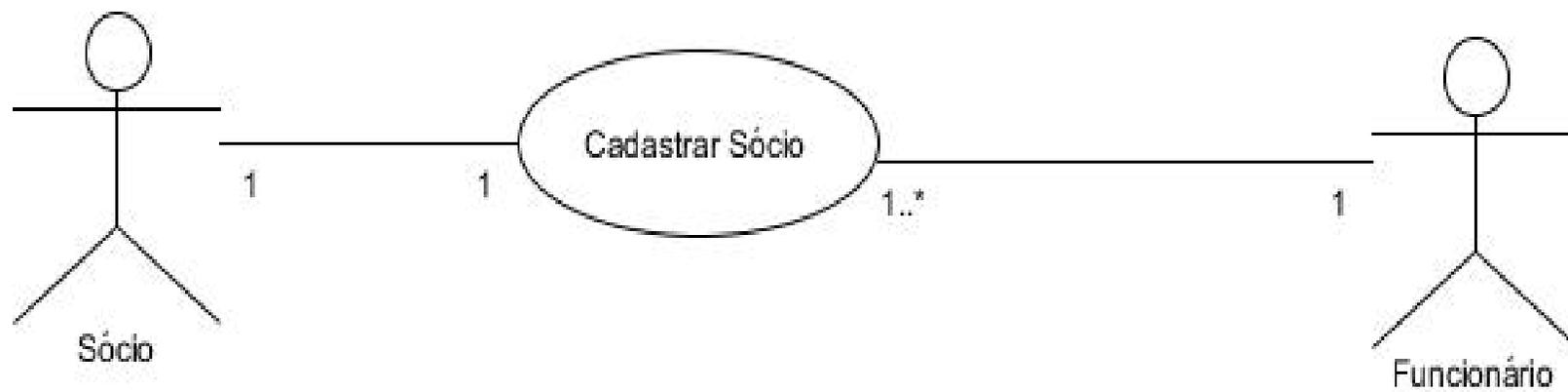


Pontos de extensão.

-  Identifica um ponto no comportamento de um UC a partir do qual esse comportamento poderá ser estendido pelo comportamento de outro UC.

Multiplicidade.

-  Representa quantas vezes um ator poderá utilizar um UC.

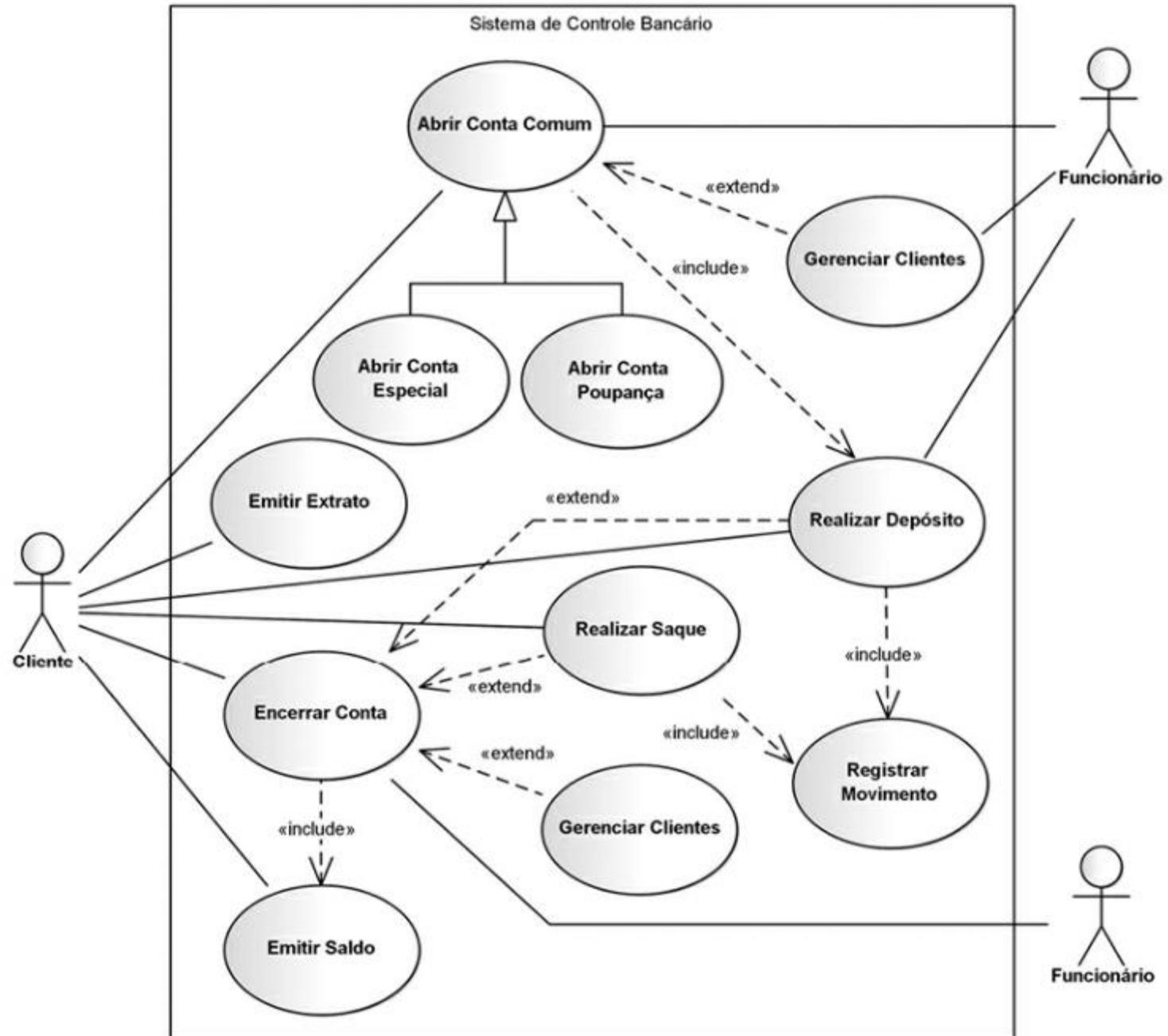


Estereótipo.

-  Possibilita extensibilidade aos componentes ou associações da UML.
-  Permite a identificação de componentes ou associações podendo atribuir funções extras ou diferentes a um componente.

Fronteira de Sistema.

-  Permite identificar um subsistema ou um sistema completo e ainda destacar o que está contido no sistema ou não.



[Questão]

(SLU-DF – Analista de Gestão de Resíduos Sólidos – Informática – 2019 - Cespe)

Acerca de conceitos e disciplinas da engenharia de software, julgue o item que se segue.

Na UML, o diagrama de caso de uso demonstra as interações entre os usuários e o software, por meio de uma sequência de eventos realizados pelos usuários.

[Questão]

(UFF - Analista de TI– 2019 – Coseac)

Na UML 2.0, a inclusão e a extensão são utilizadas na:

- A) realização.
- B) agregação.
- C) composição.
- D) associação.
- E) especialização.

[Questão]

(AFAP – Analista de Fomento – TI – 2019 – FCC)

Um Analista de Informática está desenhando um Diagrama de Caso de Uso usando a notação padrão da UML 2.5 e modela os seguintes requisitos:

Caso de Uso: Cadastrar Funcionário que:

- I. Exige um relacionamento incondicional com outro Caso de Uso denominado Validar Matrícula.
- II. Possui um relacionamento com o Caso de Uso denominado Cadastrar via Web que é uma variação de Cadastrar Funcionário.
- III. Deverá se relacionar com o Caso de Uso denominado Help On-line, se o usuário operador solicitar ajuda (Help) mediante seleção dessa opção.

Os requisitos especificados acima tipificam os respectivos relacionamentos entre Casos de Uso:

- A) Associação, Generalização e Especialização.
- B) Inclusão, Especialização e Exclusão.
- C) Inclusão, Generalização e Extensão.
- D) Extensão, Exclusão e Associação.
- E) Generalização, Especialização e Extensão.

[Questão]

(FUB – Técnico de TI – Analista de Fomento – TI – 2018 – Cespe)

Acerca dos processos de desenvolvimento de software, julgue o item a seguir.

O diagrama de casos de uso deve ser elaborado após o levantamento de requisitos, para refletir as principais funcionalidades do software.

[Questão]

(BNB – Especialista Técnico – Analista de Sistema – 2018 – Cespe)

Considerando os conceitos de análise e projeto orientados a objetos, julgue o item subsecutivo.

Dois métodos podem ser utilizados para identificar casos de uso: um que se baseia em atores, em que primeiro são identificados os atores e, depois, os eventos dos quais eles participam; e um que se baseia em eventos, em que primeiro são identificados os eventos e, depois, os atores relacionados.

[Questão]

(TCE-MG – Analista de Controle Externo – 2018 – Cespe)

Em um diagrama de caso de uso, o ator representa

- A) máquinas que interagem com o sistema.
- B) humanos e outros sistemas que interagem com o assunto ou com o sistema.
- C) uma elipse e um rótulo com o nome do caso de uso.
- D) papéis que os humanos tomam ao interagir com o sistema.
- E) humanos específicos que interagem com o sistema.

UML

Diagrama de Classes

Diagrama de Classes.

- Um dos mais importantes diagramas, tem seu foco em permitir a visualização das classes que irão compor o sistema com seus atributos e métodos e demonstrar como as classes do diagrama se relacionam, complementam e transmitem informações entre si.
- Apresenta uma **visão estática** de como as classes estão organizadas definindo assim uma estrutura lógica.
- Serve como base para a construção da maioria dos outros diagramas.
- Composto de:
 - Classes.
 - Associações.

- No PU temos a sua utilização na fase de análise para a produção de um modelo conceitual a respeito das informações necessárias ao software, momento no qual o engenheiro irá se preocupar em representar as informações que o software necessitará (classes, atributo e associações), os métodos ficam para definição do como e deverão ser considerados na fase de projeto a partir da modelagem de diagramas de interação (como o de seqüência).

Classe.

-  Representada por um retângulo com até três divisões:
 -  Descrição ou nome da classe.
 -  Atributos e seus tipos de dados. (obrigatória somente se a classe tiver atributos ou se precisar mostrá-los).
 -  Métodos da classe. (obrigatória somente se a classe tiver métodos ou se precisar mostrá-los).

Atributos e métodos.

-  Os atributos armazenam os dados dos objetos, que podem variar de instância para instância, e os métodos são operações/funções que uma instância da classe pode executar, sendo que seus métodos são idênticos.
-  Obs.: o diagrama de classes não irá definir as etapas a serem percorridas pelos métodos, isso é função do diagrama de atividades.
-  Os métodos podem receber valores como parâmetros e retornar valores que podem ser:
 -  Resultado produzido pela execução do método.
 -  Valor representado se o método foi realizado com sucesso ou não.

Conta

nro_conta : long

dt_abertura : Date

+ abrir_Conta(): long

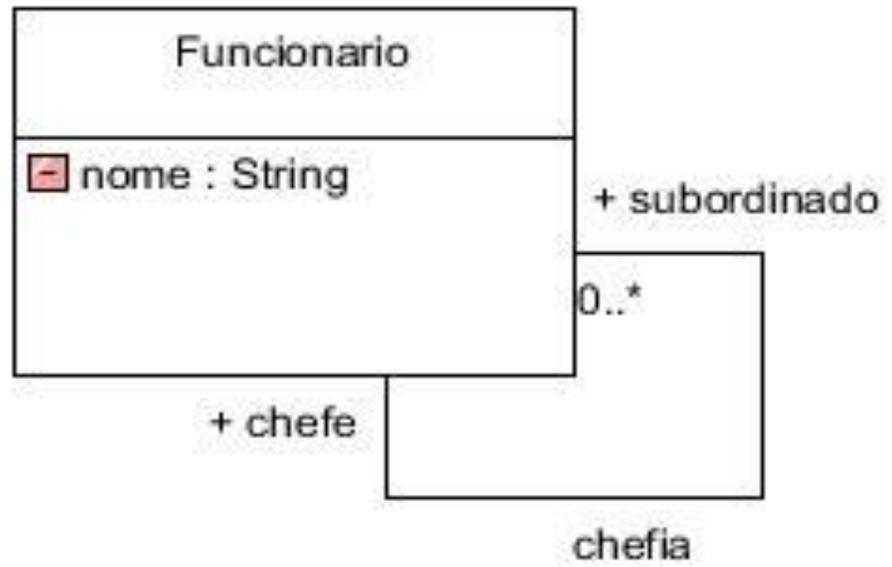
+ consultar_Conta(): long

Diagrama de Classes – Relacionamentos ou associações

- Permitem o compartilhamento de informações entre si e a colaboração para a execução dos processos executado pelo sistema.
- Descreve um vínculo que ocorre normalmente entre os objetos de uma ou mais classes.
- São representadas por linhas ligando as classes envolvidas que poderão ter nomes ou títulos para auxiliar a compreensão do tipo de vínculo estabelecido entre os objetos das classes envolvidas nas associações.

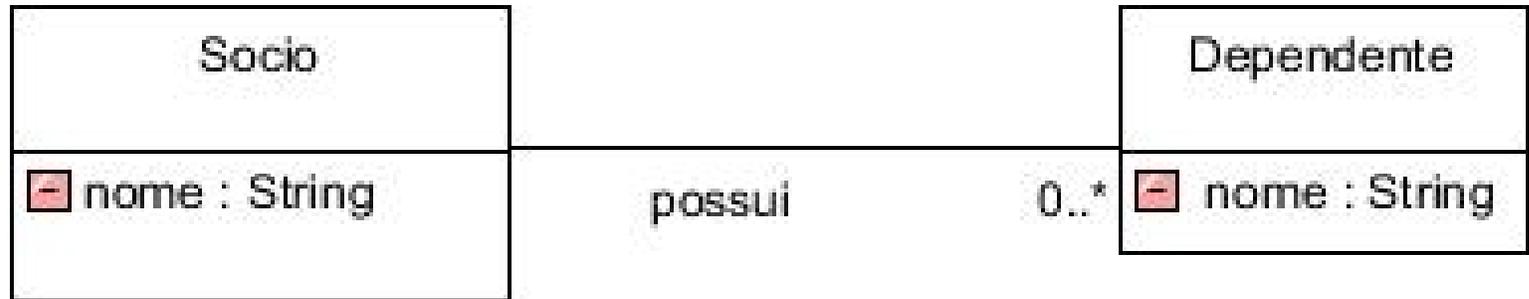
Associação unária ou reflexiva.

-  Ocorre quando existe um relacionamento de um objeto de uma classe com objetos da mesma classe.
-  Apresenta o seu nome, o papel (que não é obrigatório, mas ajuda a explicar a função de um objeto e possuem critérios de visibilidade) e também a multiplicidade, que determina o número mínimo e máximo de objetos envolvidos em cada extremidade da associação e também especifica o nível de dependência de um objeto para com os outros envolvidos na associação.
-  Multiplicidades: i) 0..1. ii) 1..1. iii) 0..*. iv) 1..*. v) 2..6.



Associação binária.

-  Quando identificamos relacionamentos entre objetos de duas classes distintas.
-  Sua navegabilidade é representada por uma seta em uma das extremidades da associação, identificando o sentido em que as informações são transmitidas entre os objetos, o sentido em que os métodos serão disparados, se não houver setas é sinal que as informações podem trafegar em qualquer uma das direções.

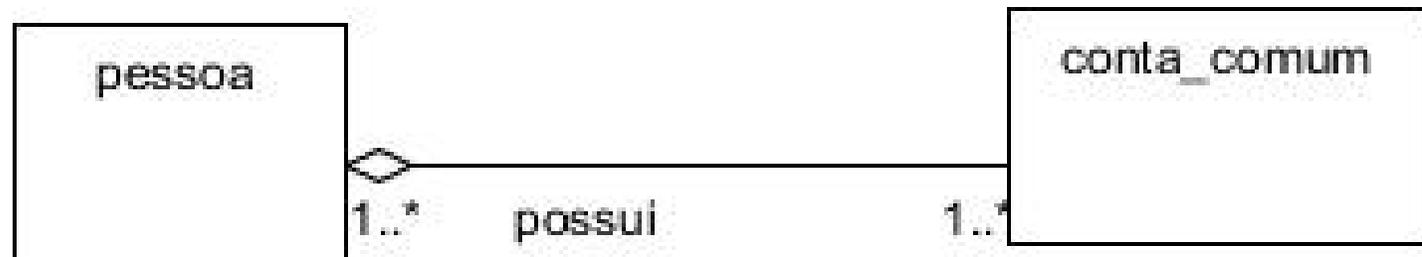


Associação ternária.

-  Conectam objetos de mais de duas classes e são representados por um losango para onde convergem as ligações da associação.

Agregação.

-  Demonstra que as informações de um objeto-todo precisam ser complementadas pelas informações de um ou mais objetos-parte.
-  Representada por uma linha com um losango vazio na ponta conectado ao objeto-todo.
-  Obs.: poderá ser substituída por uma associação binária, desde que não se tenha a obrigatoriedade de complementação de um objeto-todo por um objeto-parte.



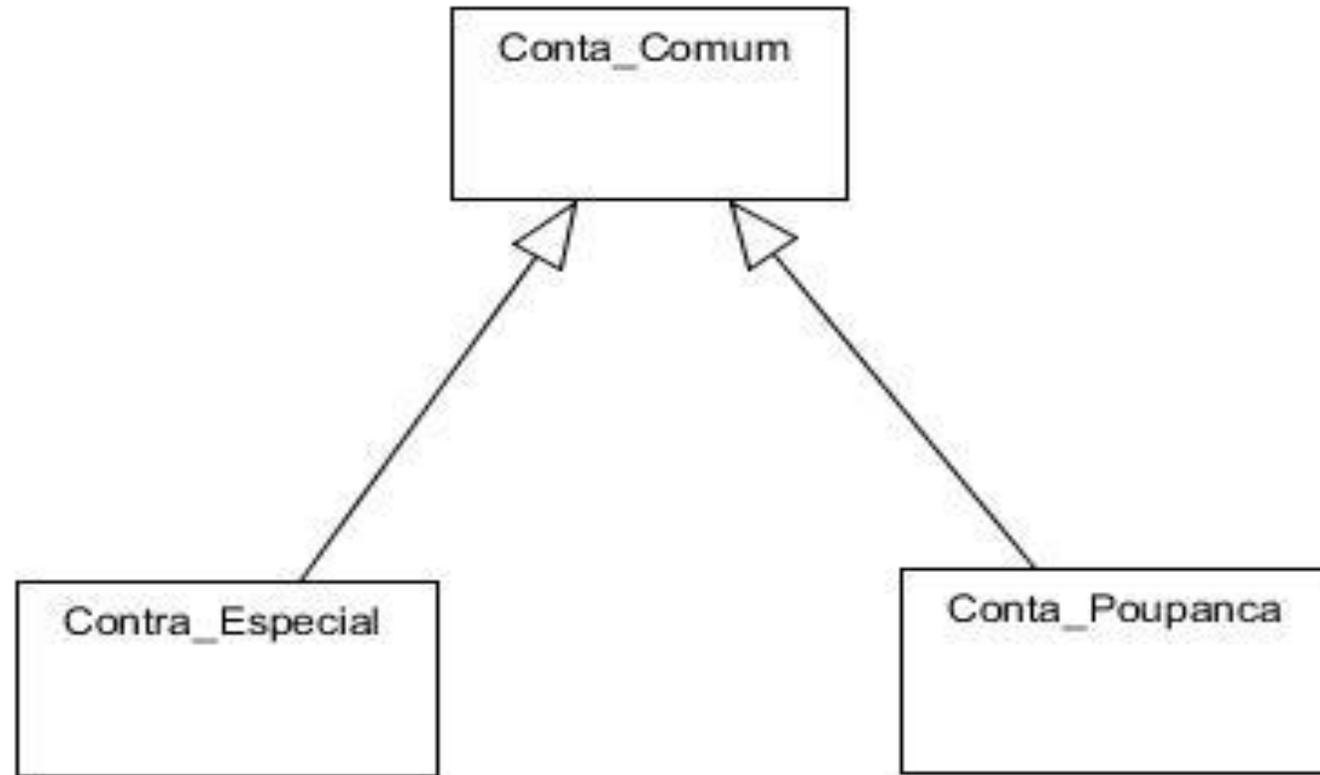
Composição.

-  Variação da agregação que considera uma relação mais forte entre o todo e a parte, na qual o objeto-parte deverá estar associado a um único objeto-todo e só poderá ser destruído por ele.
-  Representada por uma linha com um losango preenchido na ponta conectado ao objeto-todo.



Generalização/Especialização.

-  Similar ao utilizado no Diagrama de UC.
-  Tem como objetivo identificar as classes mães (gerais) e filhas (especializadas) de uma associação mediante herança.
-  Demonstra possíveis métodos polimórficos nas classes especializadas e ocorrem quando temos duas ou mais classes com características muito semelhantes.
-  Os métodos poderão ser declarados com o mesmo nome em uma classe filha assumindo novos comportamentos e sem a necessidade de alterarmos o código da classe mãe.



Classe Associativa.

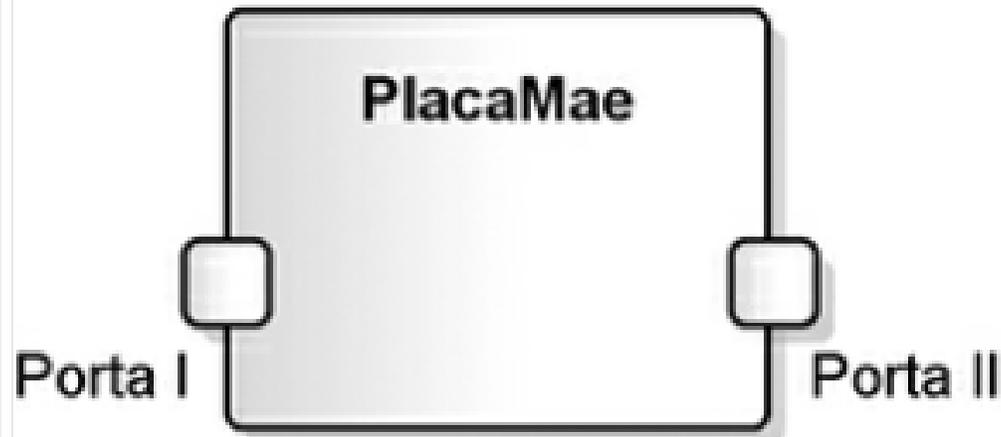
-  Produzida quando ocorrerem classes com multiplicidade muitos (*) em todas as extremidades, são necessárias no caso em que existam atributos relacionados à associação que não podem ser armazenados por nenhuma das classes envolvidas.

Dependência.

-  Identificar o grau de dependência de uma classe em relação à outra.
-  Representada por uma linha tracejada entre duas classes, contendo uma seta que aponta para a classe da qual a classe posicionada na outra extremidade do relacionamento é dependente.

🌐 Portas.

- 🌐 Característica estrutural que especifica uma interação entre o classificador e seu ambiente ou entre o classificador e suas partes internas.
- 🌐 É um ponto de comunicação.



 **Interfaces:** Meio para descrever os serviços que estão disponíveis de classes específicas, podem ser:

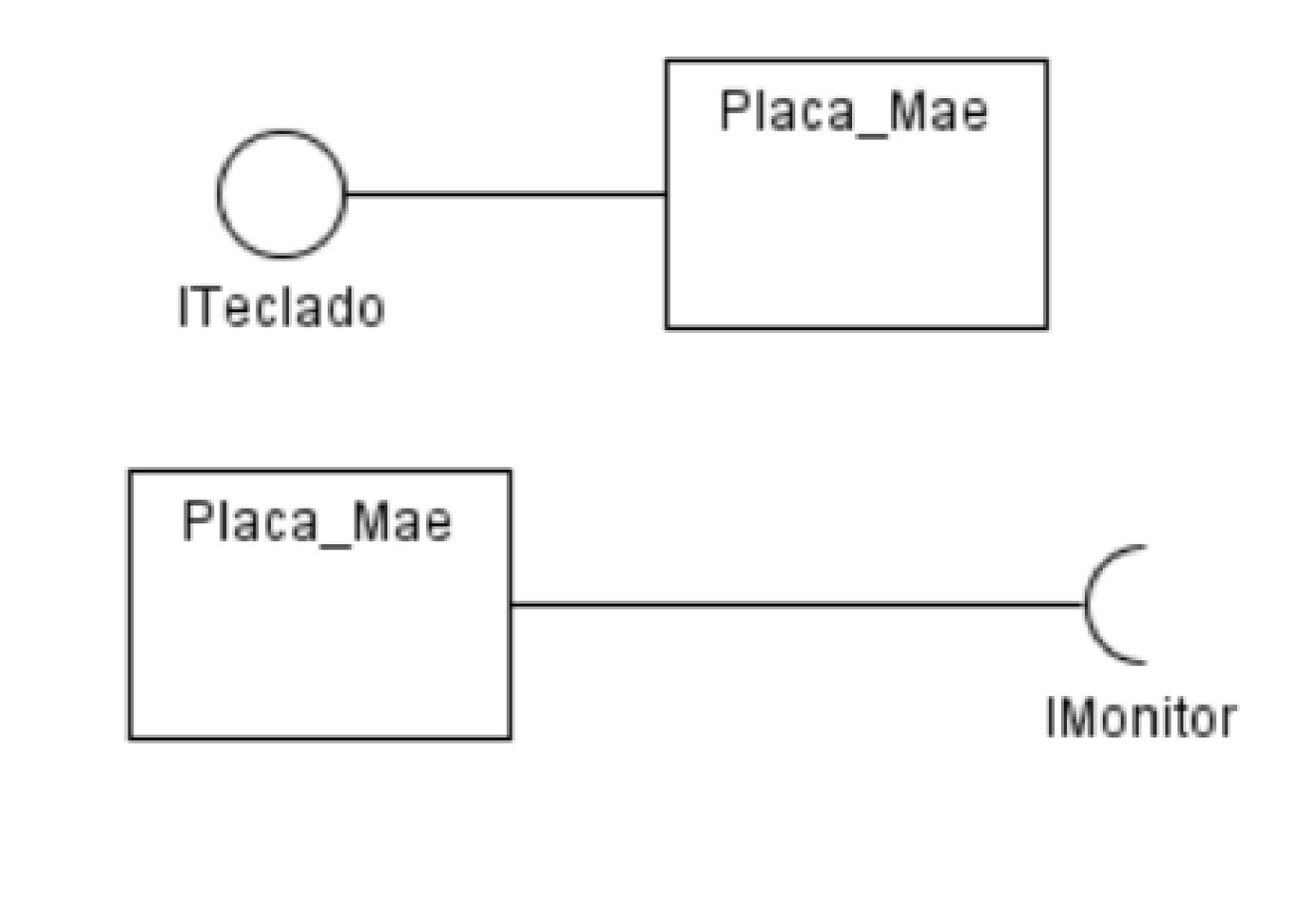
 **Fornecidas:**

-  Descreve um serviço implementado por uma classe.
-  Representa o conjunto de serviços que a classe oferece a seus clientes.
-  São representadas por um círculo fechado ligado à classe por uma linha sólida.

Interfaces (cont).

Requeridas:

-  Descreve os serviços que outras classes devem fornecer a uma determinada classe e não precisa saber quais classes irão implementar esses serviços.
-  São representadas por um semicírculo ligado a uma classe por uma linha sólida.
-  Comumente serão encontradas interfaces fornecidas que serão requeridas de outras.
-  Algumas vezes as classes fornecidas e requeridas poderão ser substituídas pelos relacionamentos de dependência e realização.



Estereótipos de Classes.

 <<enumeration>>: valores enumerados (situação, estado).

 Projeto Navegacional.

 <<server page>>



 <<client page>>



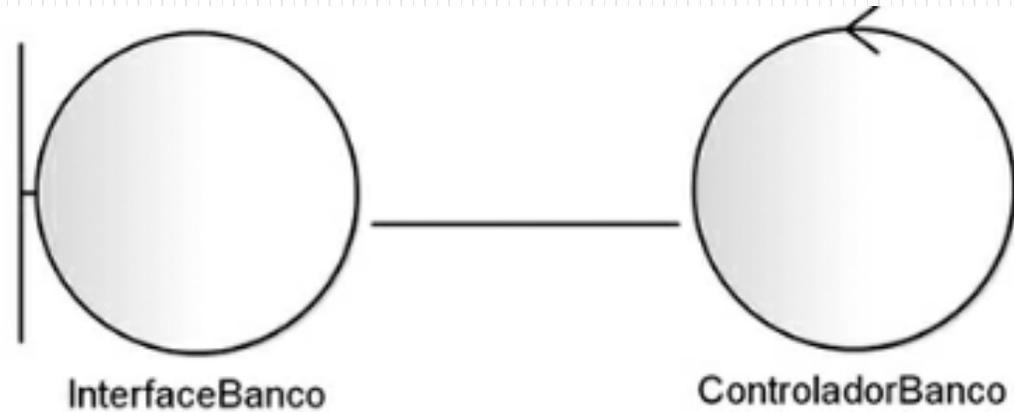
Estereótipos de Classes.

 <<form>>

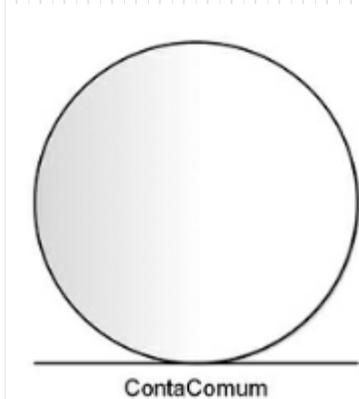


Estereótipos de Classes.

<<boundary>> e <<control>>



<<entity>>



Restrições.

-  Informações extras que definem condições a serem validadas.
-  São representadas por textos limitados por chaves.
-  Podem ser utilizadas para detalhar requisitos não-funcionais, incluindo regras de negócio.
-  OCL (Object Constraint Language) pode ser utilizada para documentar casos mais complexos de restrições.
 -  Coleção:
 -  Conjunto de comunidades associadas a um usuário, restrições que podem ser aplicadas a coleções:
 -  Unique: um elemento na coleção não pode se repetir.
 -  Bag: um mesmo elemento pode aparecer mais de uma vez.
 -  Sequence ou seq: representa uma seqüência, uma bag ordenada.

Modelo conceitual X modelo de domínio.

-  O modelo conceitual é produzido durante a fase de análise e refere-se ao domínio do problema, enquanto o modelo de domínio refere-se ao domínio da solução.
-  O modelo conceitual apresenta somente as informações que o sistema irá necessitar, enquanto o modelo de domínio toma o modelo conceitual e detalha questões como métodos, navegabilidade e até mesmo pode inserir novas classes, se isso for considerado necessário. [GILEANES].
-  O modelo de domínio só deverá ser desenvolvido na fase de projeto e deve ser modelado junto com o detalhamento dos casos de uso por meio de diagramas de interação, como o de seqüência.

☎ Persistência e Mapeamento de Classes em Tabelas.

- ☎ Cabível em situações que temos a real necessidade de termos uma classe sendo persistida em tabelas.
- ☎ Preocupação visível quando se usa um SGBD relacional, pois se fosse OO não precisaria termos tal preocupação.
- ☎ A identificação das classes que serão persistentes é feita pela identificação de seus estereótipos ou de suas restrições.
- ☎ A forma como as instâncias serão preservadas dependerá diretamente da forma como o sistema será implementado que poderão ser armazenadas em disco ou mantidas em memória e registradas no disco apenas quando do encerramento do sistema (uso da camada de persistência).

⌚ Persistência e Mapeamento de Classes em Tabelas (cont.)

- ⌚ Existe ainda a necessidade de identificarmos se todos os atributos de uma classe são persistentes ou não.
- ⌚ Teremos em vários casos as classes persistentes com o seu valor equivalente em forma de tabela ou dividida em mais de uma tabela:
 - ⌚ Cada atributo será uma coluna na tabela.
 - ⌚ Cada objeto será uma linha.
- ⌚ Frameworks (Hibernate, Toque, Castor).

🌐 Associações e chaves estrangeiras.

🌐 Associação de 1 para 1.

- 🌐 Quando existente deve-se adicionar uma chave estrangeira em uma das tabelas envolvidas na associação para referenciar a chave primária da tabela localizada na outra extremidade da associação.

🌐 Associação 1 para muitos.

- 🌐 Quando existente deve-se adicionar uma chave estrangeira na extremidade muitos (*) da associação para se referenciar à chave primária da tabela da outra extremidade.

🌀 Associações e chaves estrangeiras (cont.).

🌀 Associação muitos para muitos.

- 🌀 Dentro de OO é aceitável, no relacional deve-se criar uma tabela intermediária (forma normal).
- 🌀 Deve-se criar uma tabela intermediária contendo duas chaves estrangeiras, cada relacionada à chave primária de uma das classes com a qual se relaciona.
- 🌀 Chaves estrangeiras podem constituir a própria chave primária da tabela intermediária ou pode-se criar uma chave primária própria.
- 🌀 Quando já existir uma classe associativa, cria-se uma tabela para representá-la contendo seus atributos e depois se cria uma chave estrangeira para relacionar-se com cada chave primária das tabelas com a qual essa tabela se relaciona.

⌘ Associações e chaves estrangeiras (cont.).

⌘ Associações ternárias.

- ⌘ Cria-se uma tabela para representar a associação ternária e uma chave estrangeira para esta se relacionar com cada tabela associada.

⌚ Associações e chaves estrangeiras (cont.).

⌚ Herança.

⌚ Divide-se em três procedimentos possíveis:

⌚ Primeira estratégia.

⌚ Toda a hierarquia de classes será representada por uma única tabela no banco de dados.

⌚ Deverá conter uma coluna para identificar o tipo do objeto e também uma chave primária.

⌚ Segunda estratégia.

⌚ Cria-se uma tabela para cada classe concreta existente.

⌚ Terceira estratégia.

⌚ Cria-se uma tabela para cada classe da hierarquia, relacionando-as por meio de chaves estrangeiras.

[Questão]

(SANASA Campinas – ATI - 2019)

No que se refere aos diagramas da análise orientada a objetos, julgue o item.

O diagrama de classe retrata diversos elementos estáticos, juntamente com suas associações, suas construções de herança, sua agregação etc.

[Questão]

(Prefeitura de Manaus –AM – Tecnologia - FCC - 2019)

Um técnico deve, em um diagrama de classes da UML 2.0, utilizar a notação para declarar um atributo denominado 'produto', de uma classe, podendo tal atributo conter de 2 a 6 valores. Dessa forma, a maneira correta de declarar esse atributo é

- A) produto [2 // 6]
- B) produto [2 .. 6]
- C) produto [2/3/4/5/6]
- D) produto [2 ↔ 6]
- E) produto [2 until 6]

[Questão]

(Prefeitura de Manaus –AM – Tecnologia - FCC - 2019)

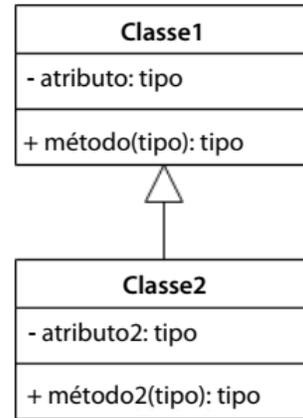
Um técnico, ao utilizar a UML 2.0, deve atentar que os atributos e operadores de uma classe exibem propriedades de visibilidade, sendo correto que o nível de visibilidade

- A) protegido (protected) aplica-se a árvores de herança.
- B) pacote (package) aplica-se a sistemas.
- C) privado (private) aplica-se a pacotes.
- D) público (public) aplica-se a classes.
- E) privado (private) aplica-se a sistemas.

[Questão]

(Prefeitura de Manaus –AM – Tecnologia - FCC - 2019)

Considere o diagrama de classes representado pelas classes Classe1 e Classe2:



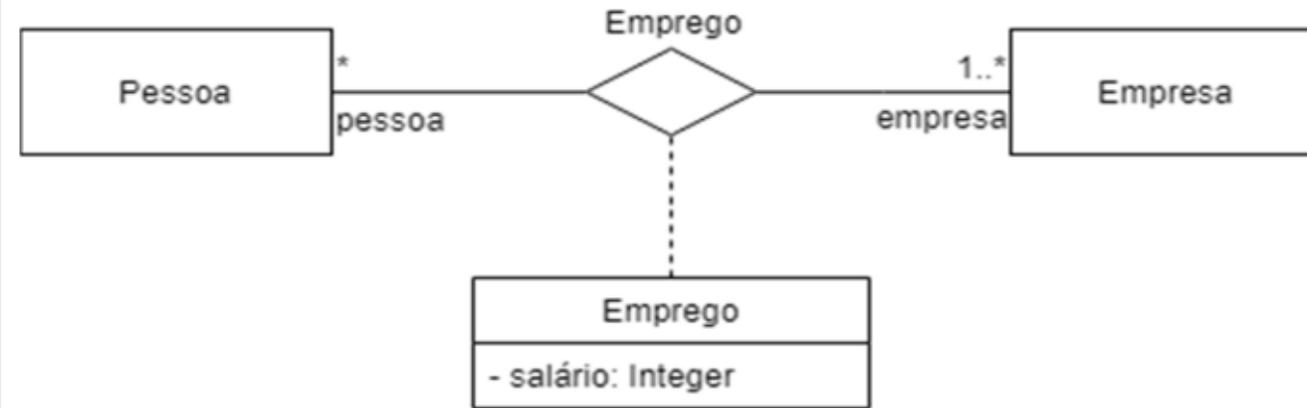
Assinale a alternativa que representa a relação entre as classes de acordo com o paradigma de orientação a objetos.

- A) Herança.
- B) Refatoração.
- C) Acoplamento.
- D) Coesão.

[Questão]

(IF-PB – Professor – Informática – 2019 - IDECAN)

Considere o seguinte diagrama de classes.



Sobre o diagrama acima, é correto afirmar que

- A) as representações gráficas das classes Pessoa e Empresa não seguem o padrão da especificação UML.
- B) salário é um método da classe Emprego.
- C) um objeto da classe Pessoa pode estar associado a zero ou mais objetos da classe Empresa.
- D) a classe Emprego é uma classe de associação.
- E) uma empresa deve ter pelo menos um funcionário.

[Questão]

(IF-PA – ATI – Desenvolvimento de Sistemas – 2019 – IF-PA)

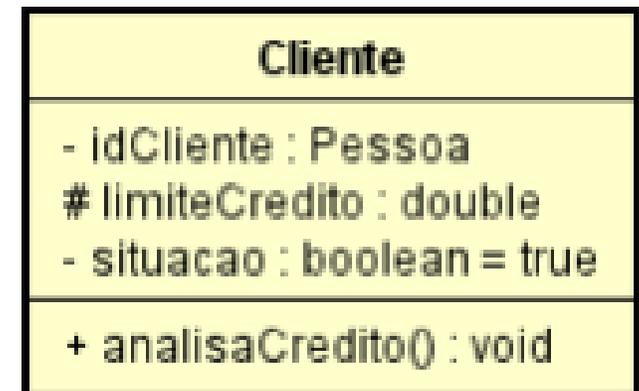
Considere o modelo abaixo, baseado de acordo com a especificação da UML (Unified Modeling Language), versão 2.5.1.

Com base nesse modelo, podemos afirmar que:

- I) o atributo “idCliente” é do tipo “Pessoa”, que se refere a uma outra classe já modelada.
- II) o atributo “limiteCredito” é o único atributo que é público, ou seja, não está encapsulado como os demais.
- III) o atributo “situacao”, do tipo “boolean” possui a restrição de obrigatoriedade do seu conteúdo.
- IV) o “void” do método “analisaCredito()” irá indicar que o método não possui parâmetros.

O CORRETO está em:

- A) I e III somente.
- B) I, II e IV somente.
- C) II e III somente.
- D) I somente.
- E) III somente.



[Questão]

(UF-PI – Técnico - TI – 2018 – Copese)

Na terminologia UML (Unified Modeling Language), aplicada ao processo de modelagem de dados, os tipos relacionamento e as instâncias de relacionamento são conhecidas, respectivamente, como

- A) associações e links.
- B) associações e atributos.
- C) referências e links.
- D) modelos e associações.
- E) atributos e referências.

[Questão]

(Câmara de Palmas – Analista de Sistemas – 2018 – Copese)

A Linguagem de Modelagem Unificada (UML) é uma linguagem visual para especificar, construir e documentar os artefatos dos sistemas. Nesse sentido, assinale a alternativa CORRETA.

- A) A UML define apenas um perfil UML que especializa subconjuntos da notação para áreas de assunto comum, tais como diagramação de Enterprise JavaBeans
- B) A UML é uma notação diagramática padrão, de fato, para desenhar ou apresentar figuras relacionadas ao software.
- C) Em engenharia reversa uma ferramenta UML lê o código fonte ou o código binário e gera apenas diagramas UML de pacotes, não permitindo a geração de pacotes de classes e de sequência.
- D) A UML descreve tipos de esboço de diagramas, tais como diagramas de classes e diagramas de sequência. Ela superpõe a eles uma perspectiva de modelagem. Por exemplo, a mesma notação UML de diagrama de classes não pode ser usada para desenhar imagens de conceitos do mundo real ou de classes de software em Java.

UML

Diagrama de Objetos

Diagrama de objetos.

- Fornece uma visão dos valores armazenados pelos objetos das classes em um determinado momento do sistema.
- **Objeto:**
 - Semelhante a um componente classe.
 - Não apresentam métodos.
 - Seus atributos armazenam os valores contidos nos objetos em uma determinada situação.
 - Nome dos objetos na primeira divisão que poderá ser apresentado de três formas:
 - Todo em letras minúsculas seguido de “:” e o nome da classe com a primeira letra maiúscula.
 - Nome omitido, mas mantendo o símbolo “:” e o nome da classe.
 - Somente o nome do objeto.

pesfis1: PessoaFisica

nomePessoa = "José da Silva"
enderecoPessoa = "Av. Brasil, 2017"
cepPessoa = 90860-510
telefonePessoa = "(55) 3527-7263"
rendaPessoa = 5000,00
situacaoPessoa = 1
cpfPessoa = 71689347095
rgPessoa = 1096453125
idadePessoa = 27

🌀 Vínculos:

- 🌀 São instâncias das associações.

- 🌀 Possui a mesma forma gráfica de uma associação em uma classe, mas sem multiplicidade, visto que representa o número de instâncias de uma determinada classe.

🌀 Dependência com estereótipo <<instanceOf>>.

- 🌀 Representação dos objetos instanciados a partir de classes por meio de uma associação de dependência junto com o estereótipo <<instanceOf>>.

[Questão]

(Prefeitura de Manaus –AM – Tecnologia - FCC - 2019)

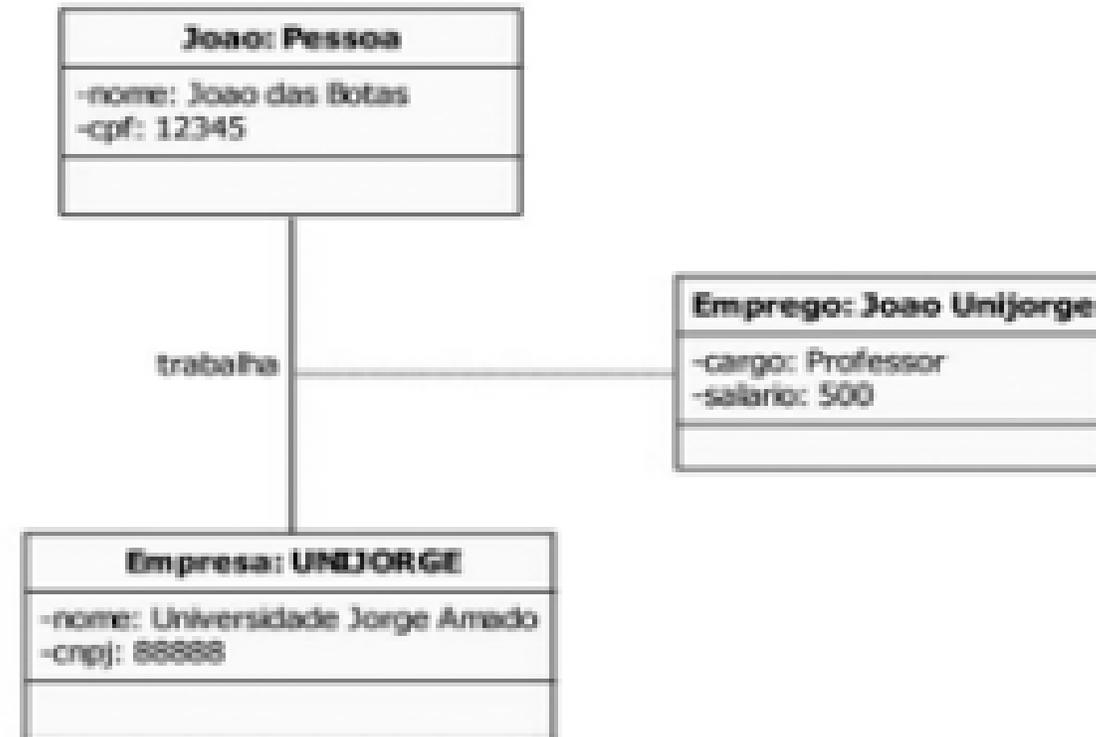
Um programador necessita fazer a representação de um diagrama de objetos da UML 2.0, sendo que as sintaxes do nome de objeto e do valor de atributo nesse tipo de diagrama são:

- A) nome-objeto IS nome-classe e nome-atributo AS valor
- B) nome-objeto = nome-classe e nome-atributo := valor
- C) nome-objeto → nome-classe e nome-atributo → valor
- D) nome-objeto : nome-classe e nome-atributo = valor
- E) nome-objeto / nome-classe e nome-atributo // valor

[Questão]

(Prefeitura de Lagoa Santa – Analista de Sistemas – Fundep - 2019)

Qual diagrama UML é representado a seguir?



- A) Diagrama de classes
- B) Diagrama de sequência
- C) Diagrama de objetos
- D) xDiagrama de relacionamentos

UML

Diagrama de Pacotes

Diagrama de pacotes

- Descreve a organização dos elementos do modelo em pacotes e demonstra a dependência entre eles.
- Útil para modelagem de subsistemas e para a modelagem de subdivisões da arquitetura de uma linguagem.
- Útil também para representar um conjunto de sistemas integrados, ou seus submódulos ou para demonstrar uma arquitetura de uma linguagem.
- **Pacotes:**
 - São utilizados para agrupar elementos e fornecer denominações para esses grupos.
 - Pode representar um sistema, um subsistema, uma biblioteca ou uma etapa de um processo de desenvolvimento, entre outras alternativas.
 - Pode conter outro pacote.

Dependência.

-  Informa que o elemento dependente necessita de alguma forma de elemento do qual depende.
-  Pode ter dois estereótipos:
 -  <<merge>>: os elementos do pacote serão unidos aos elementos do outro pacote.
 -  <<import>>: o pacote que utiliza essa dependência está importando alguma característica ou elemento do outro pacote.
-  **Pacotes contendo pacotes.**
 -  Muito comum tal ocorrência.

Estereótipos aplicados a pacotes.

 <<system>>: texto.

 <<framework>>: texto.

 <<subsystem>>: gráfico.

 <<model>>: gráfico.



UML

Diagrama de Sequencia

Diagrama de sequência.

- Diagrama comportamental que determina a seqüência de eventos que ocorrem em um determinado processo, identificando quais mensagens devem ser disparadas entre os elementos envolvidos e em que ordem.
- Determina:
 - Ordem que os eventos ocorrem.
 - Mensagem que são enviadas.
 - Métodos que são chamados.
 - Como os objetos interagem dentro de um determinado processo.

- ☰ Baseia-se no diagrama de UC havendo comumente um diagrama de sequência para cada diagrama de UC, visto que UC refere-se a um processo disparado por um ator.
- ☰ Depende também do diagrama de classes, pois as classes dos objetos utilizados nele estão descritas no diagrama de classes.
- ☰ Ao modelar o diagrama de seqüência é que se detecta quais métodos são necessários declarar em que classes.
- ☰ **Atores.**
 - ☰ Instâncias dos atores declarados nos UCs.
 - ☰ Representam entidades externas que interagem com o sistema e que solicitam serviços.
 - ☰ Geram os eventos que iniciam processos.



Cliente



Lifelines.

-  Participante individual em uma interação.
-  Irá se referir a uma instância de uma classe que participa de uma interação.
-  Pode existir desde o início do processo ou ser criado durante o decorrer da execução dele.



Linha de vida.

-  Representa o tempo em que um objeto (lifeline) existe durante um processo.
-  Representadas por linhas finas verticais tracejadas, partindo do retângulo que representa o objeto.
-  É interrompida com um “X” quando o objeto é destruído.
-  Um objeto não precisa necessariamente existir quando um processo é iniciado, podendo ser criado ao longo da sua execução.

Foco de controle ou ativação.

-  Indica os momentos que um objeto está executando um ou mais métodos de um processo específico.
-  Representados dentro da linha de vida de um objeto, porém, enquanto as linhas de vida são representadas por tracejados finos, ele será representado por uma linha mais grossa.

Mensagens e estímulos.

-  Utilizados para demonstrar a ocorrência de eventos, que normalmente forçam a chamada de um método em algum dos objetos envolvidos no processo.
-  Pode representar a comunicação entre dois atores, não disparando métodos.
-  Comumente um diagrama de sequência é iniciado por um evento externo.
-  Pode haver mensagens entre:
 -  Dois atores.
 -  Um ator e um objeto. (inicia um método).
 -  Dois objetos. (mais comum).
 -  Um objeto e um ator. (mensagem de retorno).

Mensagens e estímulos (cont.).

-  Mensagens são representadas por linhas na posição horizontal entre dois componentes (suas linhas da vida), com setas que indica qual componente enviou a mensagem e qual recebeu. Sua ordem sequencial é demonstrada de cima para baixo.
-  Quando dirigida a um objeto existente ela irá engrossar a linha da vida para representar que o objeto está com o foco, quando for a criação de um objeto, ela irá atingir o retângulo.

Mensagens de retorno.

-  Identifica a resposta a uma mensagem para o objeto ou ator que a chamou.
-  Pode retornar informações específicas do método chamado ou apenas um valor indicando se o método foi executado com sucesso ou não.
-  Representadas por uma linha tracejada contendo uma seta fina que aponta para o objeto que recebe o resultado do método chamado.

Auto chamadas ou auto delegações.

-  Mensagens que um objeto envia para si mesmo.

Detalhes de tempo.

-  Trabalhado quando temos um tempo máximo de espera até que uma mensagem seja disparada.

Mensagens perdidas e encontradas.

-  Mensagem perdida representa uma mensagem que foi enviada de sua confirmação de recebimento não foi recebida.
-  Mensagem encontrada representa o recebimento de uma mensagem enviada por um elemento desconhecido ou um elemento não representado no diagrama, ou o recebimento de uma mensagem que fora dada como perdido.
-  Ambas são representadas por um círculo preenchido, quando perdida o círculo é atingido pela mensagem, quando encontrara a mensagem parte do círculo.

Portas.

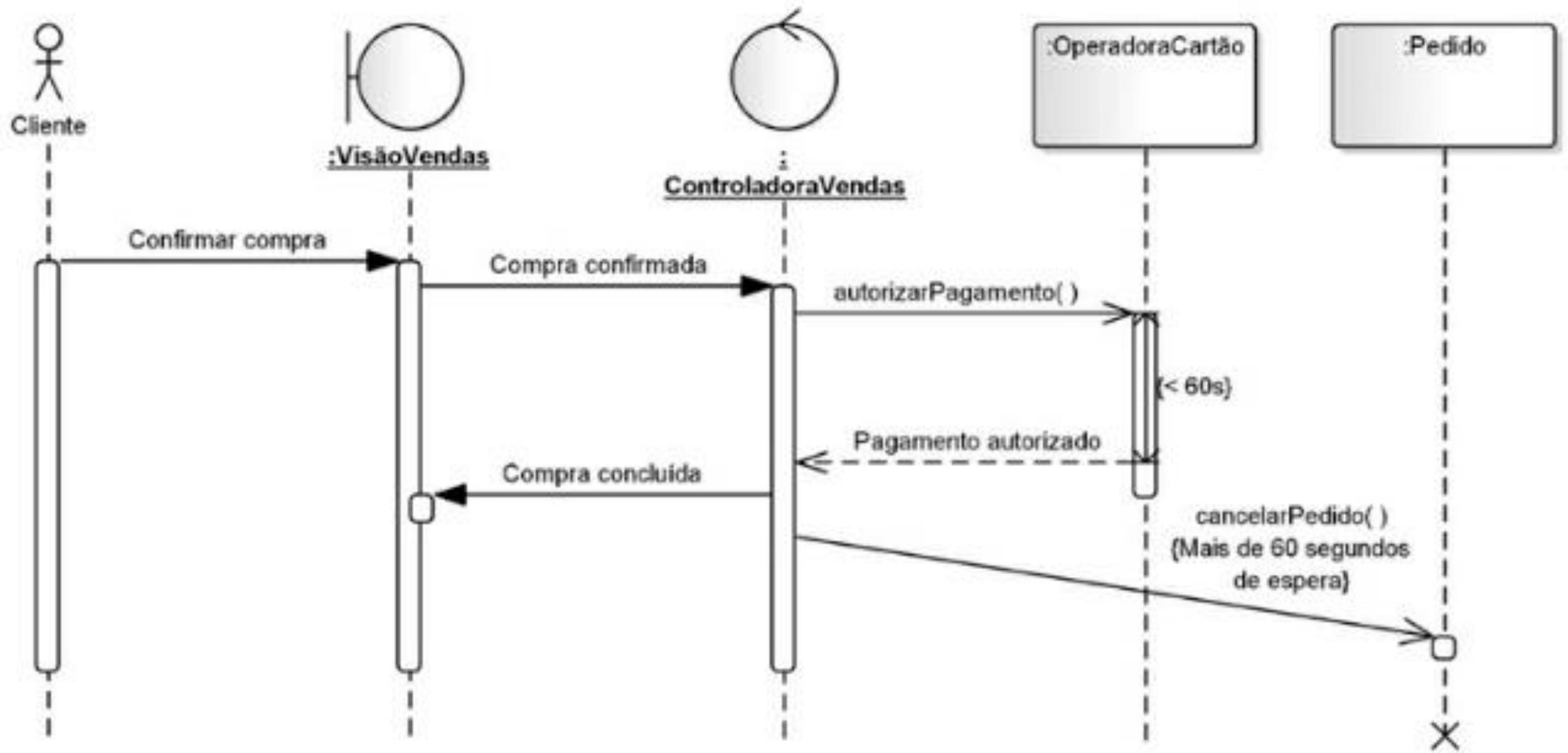
-  Possibilita a existência de mais de uma linha de vida, e permite assim a representação de mensagens externas e internas no objeto, como por exemplo, uma placa-mãe que pode ter solicitações internas.

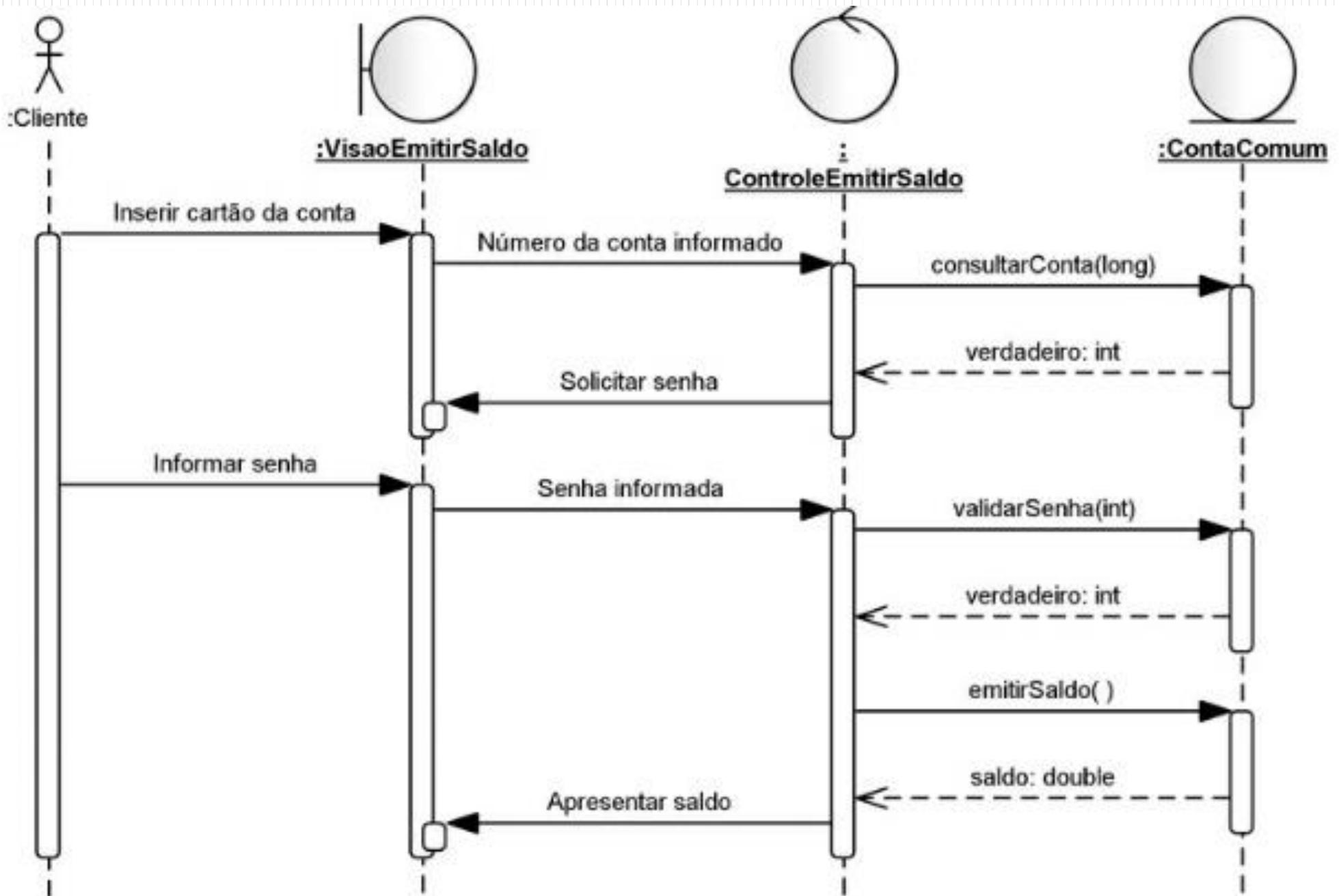
Fragmento de interação.

-  Noções abstratas de unidades de interação geral.
-  É uma parte de uma interação.
-  Cada fragmento de uma interação é considerado uma interação independente.
-  É representado por um retângulo que envolve toda a interação.

Portões (Gates).

-  Interface entre fragmentos, um ponto de conexão para relacionar uma mensagem fora de um uso de interação com uma mensagem dentro do uso de interação.





[Questão]

(UFRN – Técnico – TI – 2019 – Conperve)

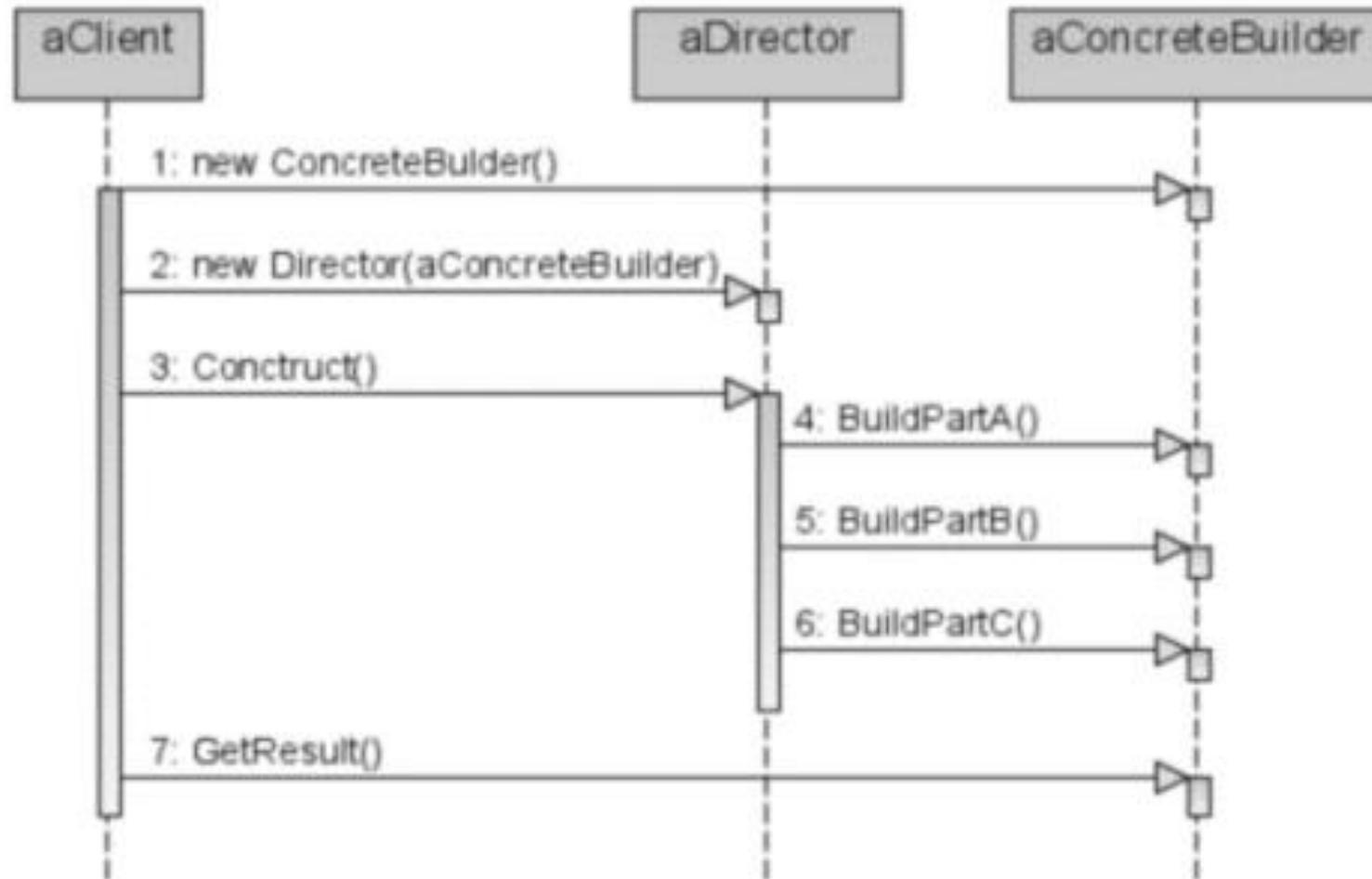
Há um tipo de diagrama da UML que dá ênfase à ordenação temporal em que as mensagens são trocadas entre os objetos de um software. Além disso, apresenta conceitos de atores, objetos, gate, fragmento e linha de vida. Esse diagrama é

- A) diagrama de componentes.
- B) diagrama de pacotes.
- C) diagrama de sequência.
- D) diagrama de estrutura composta.

[Questão]

(UFMA – ATI – 2019 – UFMA)

A figura a seguir ilustra um diagrama de sequência UML. Qual das alternativas abaixo está correta com relação a este tópico?



[Questão]

(UFMA – ATI – 2019 – UFMA)

- A) O diagrama de sequência UML é útil apenas quando se quer modelar a conexão entre clientes e servidores, não sendo necessário em aplicações que executam apenas no cliente.
- B) O diagrama de sequência UML mostra interações de objetos arranjados em uma sequência de tempo. Ele apresenta os objetos e classes envolvidos no cenário e a sequência de mensagens trocadas entre tais objetos necessárias para executar a funcionalidade do cenário.
- C) Recentemente o diagrama de sequência foi descontinuado de UML por ter caráter ambíguo e abstrato.
- D) Diagramas de sequência UML são criados derivados diretamente do diagrama de classes e não faz sentido eles serem originados dos diagramas de caso de uso.
- E) É desnecessário utilizar diagramas de sequência UML para modelar ou documentar interações entre classes ou objetos que já estão presentes no diagrama de classes. Os mesmos devem aparecer em apenas um dos dois diagramas.

[Questão]

(IF-SP - Informática – 2019 – IF-SP)

A UML possui diversos diagramas, cada qual com sua finalidade, propiciando a modelagem mais adequada de um sistema de software. Dentre os diagramas previstos na UML, um deles se destaca pelas seguintes características:

- I. É utilizado para indicar as comunicações dinâmicas entre objetos durante a execução de uma tarefa, mostrando a ordem temporal na qual as mensagens são enviadas entre os objetos para executar esta tarefa;
- II. Pode-se utilizar o diagrama para mostrar as interações em um caso de uso ou em um cenário de um sistema de Software.

Assinale a alternativa que contenha diagrama UML que apresenta as características presentes no item I e II.

- A) Diagrama de Classe.
- B) Diagrama de Estado.
- C) Diagrama de Sequência.
- D) Diagrama de Caso de Uso.

[Questão]

(Prefeitura de Lagoa Santa – Analista de Sistemas – Fundep - 2019)

Qual diagrama UML deve ser utilizado para se representar a interação entre os objetos por meio dos seus métodos, em um cenário no qual também é apresentada a ordem em que essas interações acontecem?

- A) Diagrama de classes
- B) Diagrama de objetos
- C) Diagrama de sequência
- D) Diagrama de atividades

UML

Diagrama de Comunicação

Diagramas de Comunicação.

- ☰ Conhecido como diagrama de colaboração até a versão 1.5 da UML e mudou na 2.0.
- ☰ Relacionado ao diagrama de Sequencia, só que aqui ele não se preocupa com a temporalidade do processo.
- ☰ Não suporta ocorrências de interação ou fragmentos combinados.

UML

Diagrama de Máquina de Estados

Diagramas de Máquina de Estados.

- ☎ Conhecido como diagrama de gráfico de estados ou diagram de estados nas versões anteriores e mudou na 2.0.
- ☎ Demonstra o comportamento e um elemento por meio de um conjunto finito de transições de estado.
- ☎ Expressa o comportamento de uma parte do sistema, quando recebe o nome de máquina de estados comportamental.
- ☎ Estado:
 - ☎ Representa a situação em que um elemento se encontra em determinado momento durante o período em que participa de um processo. Um objeto pode passar por diversos estados em um mesmo processo.

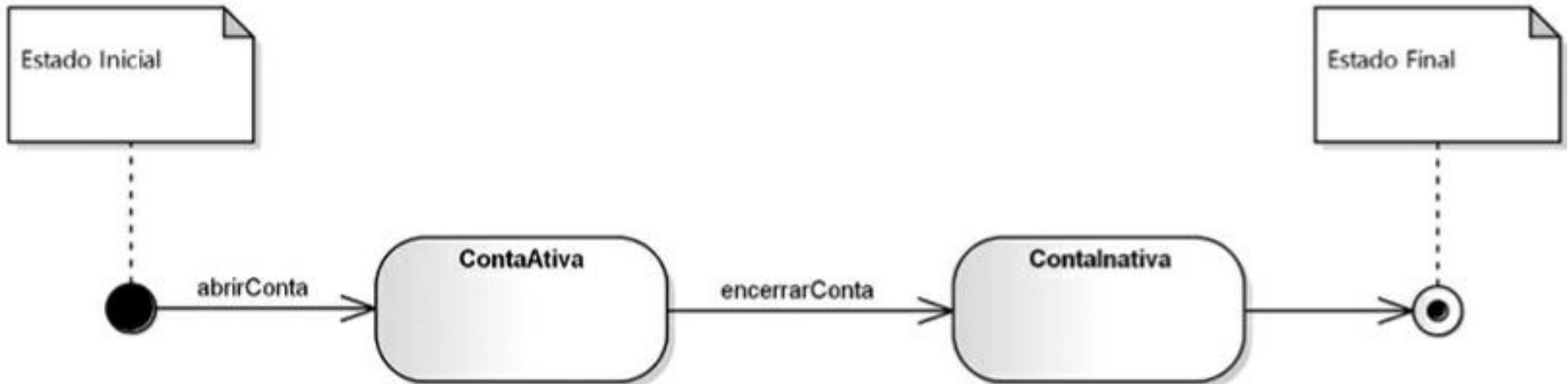
Transições:

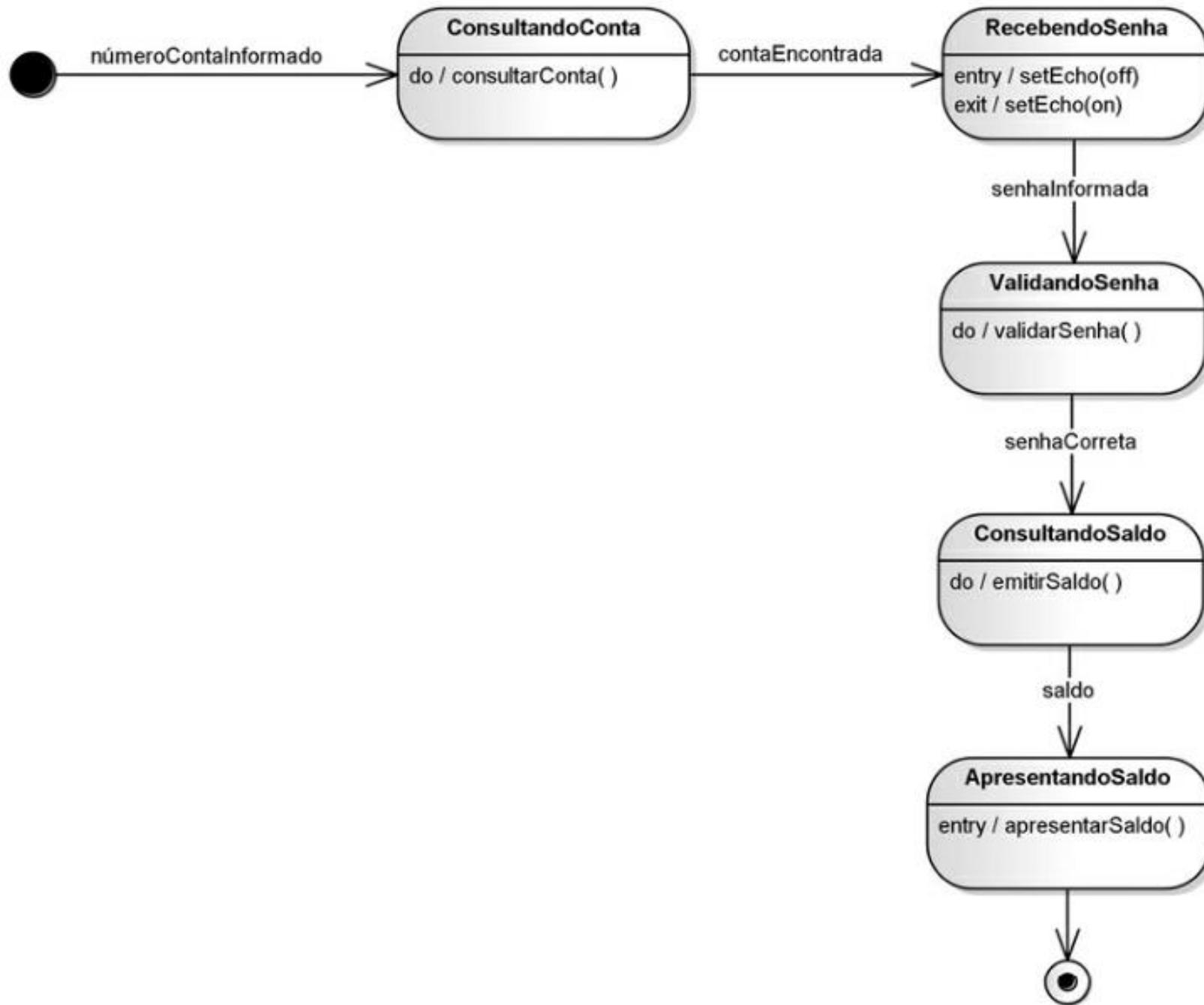
- Representa um evento que causa uma mudança no estado de um objeto, gerando um novo estado.
- Representada por uma linha ligando dois estados, com uma seta em uma de suas extremidades, que aponta para o estado novo.



Estado Inicial e Final:

- Inicial tem como função determinar o início da modelagem dos estados de um elemento.
- Final tem como função indicar o final dos estados modelados.





UML

Diagrama de Atividades

Diagrama de atividades.

- ⌚ Considerado anteriormente como um caso especial do antigo diagrama de gráfico de estados, chamado atualmente de máquina de estados.
- ⌚ É o que dá maior ênfase atualmente ao nível de algoritmo da UML e provavelmente o mais detalhista.
- ⌚ Utilizado para modelar atividades que podem ser um método ou um algoritmo, ou até mesmo um processo completo.
- ⌚ Pode modelar mais de uma atividade.
- ⌚ Utilizado para modelar dois tipos de fluxo:
 - ⌚ De controle.
 - ⌚ De objetos.

Atividades:

-  Composta por um conjunto de ações, ou seja, os passos necessários para que a atividade seja concluída.
-  Sempre conterá ações, no entanto, não necessariamente essas estarão representadas dentro da atividades, como quando for necessário fazer referência a uma atividades já modelada ou para poupar espaço no diagrama.

Nó de ação:

-  Representa um passo, uma etapa que dever ser executada em uma atividade.
-  Representado por um pequeno retângulo.

Fluxo de Controle:

-  Conector que liga dois nós, enviando sinais de controle de um nó para o outro.
-  Representado por uma linha contendo uma seta apontando para o novo nó e partindo do antigo.

Nó inicial:

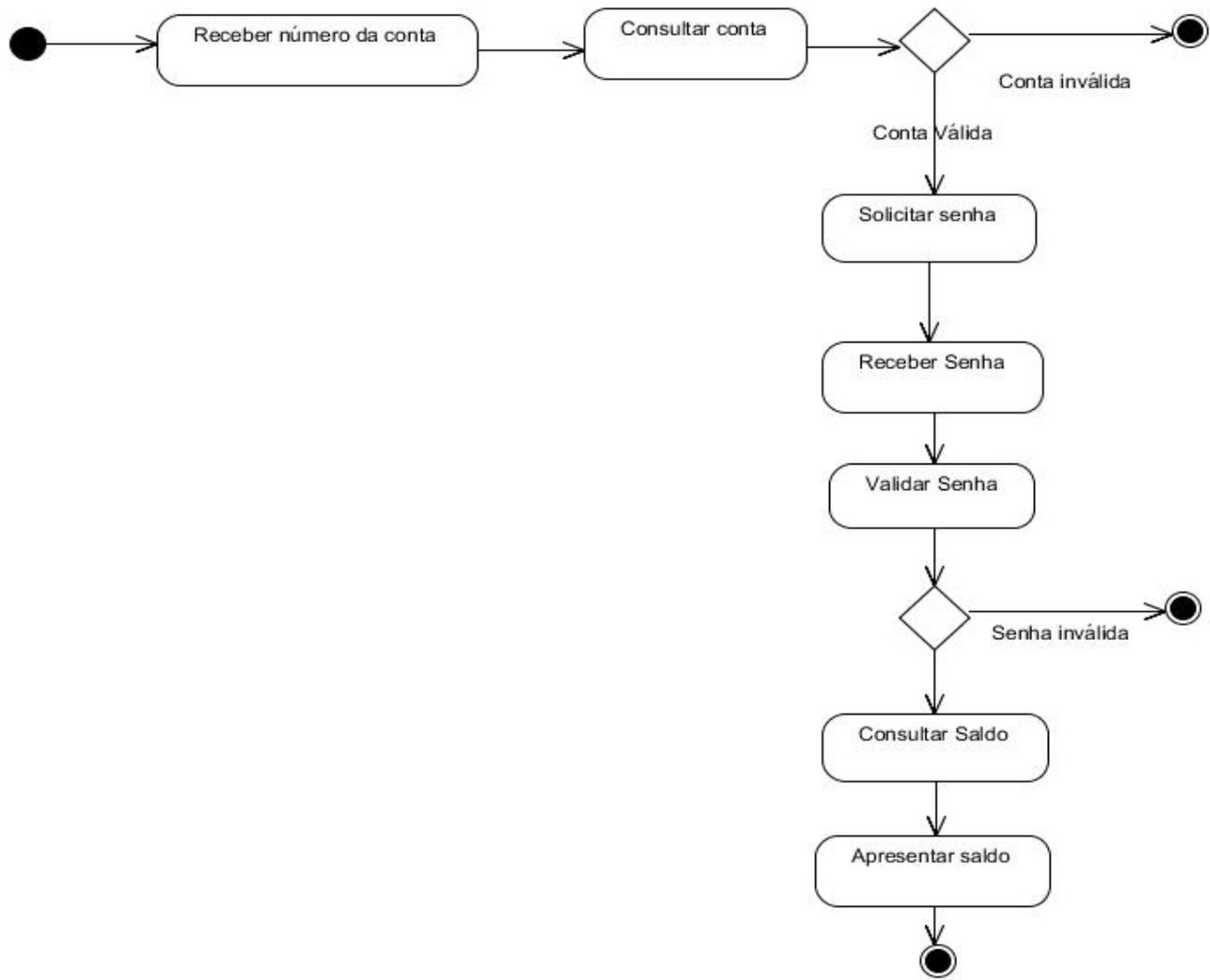
-  Pertence ao grupo de nós de controle.
-  Usado para representar o início do fluxo quando a atividade é invocada.
-  Representado por um círculo preenchido.

Nó de Final de Atividade:

-  Pertence ao grupo de nós de controle.
-  Representa o fim do fluxo de uma atividade.
-  Representado por um círculo preenchido dentro de um círculo vazio.

Nó de decisão:

-  Pertence ao grupo de nós de controle.
-  Representa uma escolha entre dois ou mais fluxos possíveis.
-  Em geral é acompanhado por condições de guarda, ou seja, textos entre colchetes que determinam a condição para que um fluxo possa ser escolhido.
-  Pode ser utilizado para unir um fluxo dividido por um nó de decisão anterior, quando passa a chamar-se nó de união.

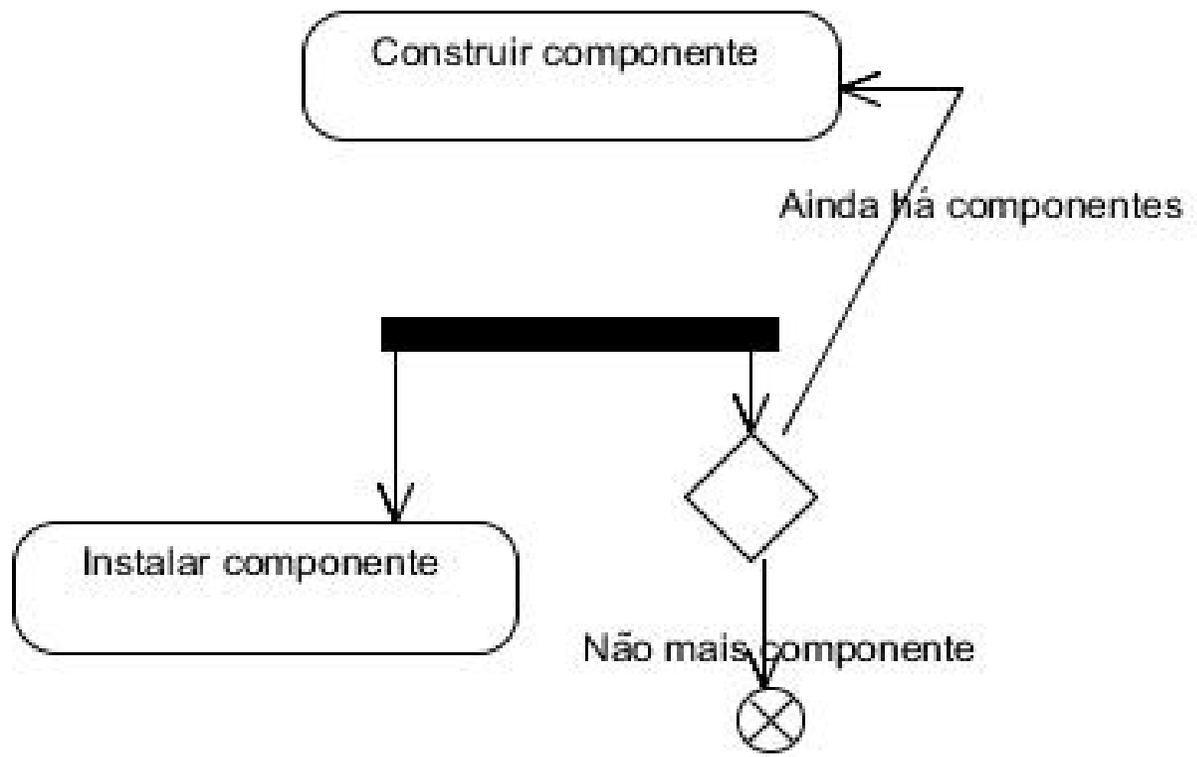


Nó de bifurcação/união.

-  Pode tanto dividir um fluxo em dois ou mais fluxos concorrentes, quando é chamado de nó de bifurcação, como mesclar dois ou mais fluxos concorrentes em um único fluxo de controle, quando é chamado nó de união.
-  Representado por uma barra que pode estar tanto na horizontal como na vertical.

Final de fluxo.

-  Representa o encerramento de uma rotina representada pelo fluxo, mas não de toda a atividade.
-  Representado por um círculo com um “X”.



Exceções.

-  Bastante comuns na maioria das linguagens de programação.
-  Seta em forma de raio que aponta para a rotina de tratamento da interrupção ou exceção.

Ação de envio de sinal.

-  Representa o envio de um sinal para um objeto ou ação.
-  Representada por um retângulo com uma protuberância triangular em seu lado direito.

Ação de evento de aceitação.

-  Representa a espera de ocorrência de um evento de acordo com determinadas condições.
-  Representada por um retângulo com uma reentrância triangular em seu lado esquerdo.

Nó de buffer central.

-  É um nó de objeto cuja função é gerenciar fluxos de múltiplas fontes e destinos.
-  Age como um buffer de memória para múltiplos fluxos de entrada e fluxos de saída de outros nós de objeto.
-  Representado como um nó de objeto com o estereótipo <<centralBuffer>>.

Nó de repositório de dados.

-  Tipo especial de nó de buffer central.
-  Representado com o estereótipo <<datastore>>. (um retângulo com pontas).
-  Usado para armazenar dados ou objetos permanentemente.

Conectores.

 Atalhos para fluxo.

 Utilizados quando da existência de uma distância relativamente grande entre os nós que o fluxo precisa ligar.

 Representado por um círculo contendo uma letra ou número exemplo apenas.

 Deverá sempre haver pares de conectores com a mesma nomenclatura.

Ação de chamada de comportamento.

 Ação que invoca a execução de um comportamento, em geral uma atividade.

 Apresenta um símbolo de ancinho apontando para baixo em seu canto inferior direito.

Ação de chamada de operação.

 É uma chamada indireta a um comportamento, na qual é invocada a execução de uma operação (método) em um objeto de uma classe.

Partição de atividade.

 Permitem representar um fluxo de um processo que passa por diversos setores ou departamentos de uma empresa, ou mesmo um processo que é manipulado por diversos atores.

 São formadas por retângulos representando divisões que identificam as zonas de influência de um determinado setor sobre parte de um determinado processo.

 Podem ser horizontais ou verticais.

Região de atividade interrompível.

-  Engloba os elementos da atividade que podem sofrer uma interrupção, todo o processo envolvido pela região poderá ser interrompido.
-  Região delimitada por um retângulo tracejado com as bordas arredondadas.

[Questão]

(SANASA Campinas – ATI - 2019)

Atenção: Para responder à questão, utilize o diagrama abaixo.



[Questão]

(SANASA Campinas – ATI - 2019)

Trata-se de um Diagrama UML de

- A) Sequência que descreve a criação de casos de uso.
- B) Comunicação que define a coleta de requisitos na forma de casos de uso.
- C) Atividades que descreve etapas do levantamento de requisitos.
- D) Sequência que descreve etapas da coleta coletiva de requisitos.
- E) Colaboração que detalha a Engenharia de Requisitos em modelos ágeis.

[Questão]

(Prefeitura de Manaus –AM – Tecnologia - FCC - 2019)

Um programador necessita fazer uma representação em UML 2.0, de um diagrama de atividades. Nesse tipo de diagrama, os nós inicial e final têm, respectivamente, as seguintes notações.

- A)  e 
- B)  e 
- C)  e 
- D)  e 
- E)  e 

[Questão]

(UFC – Técnico – TI – Desenvolvimento – CCF - 2019)

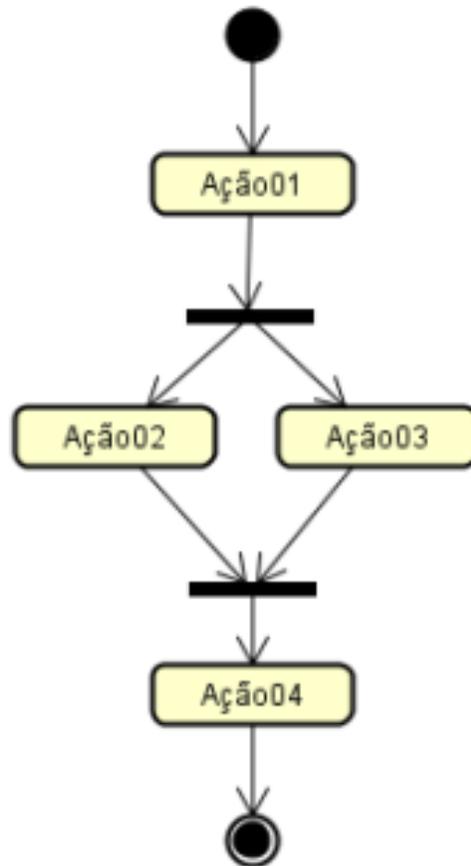
Sobre a UML 2.5 - (Unified Modeling Language), assinale a alternativa correta.

- A) O diagrama de caso de usos é composto de três elementos: atores, casos de uso e classes.
- B) Os diagramas UML são divididos em três grandes categorias: Estruturais, Comportamentais e Documentais.
- C) São exemplos de diagramas UML: diagrama de sequência, diagrama de máquina de estados e diagrama de estrutura composta.
- D) Os diagrama de classes representa os objetos em um determinado instante de tempo, representando suas instâncias e seus relacionamentos.
- E) O diagrama de atividades pertence à categoria dos diagramas comportamentais, enquanto o diagrama de sequência pertence à categoria dos diagramas documentais.

[Questão]

(IF-PA – ATI – Desenvolvimento de Sistemas IF-PA - 2019)

Com o diagrama de atividades da UML abaixo:



[Questão]

Assinale a alternativa CORRETA.

- A) O elemento que está entre a ação “Ação01” e as ações “Ação02” e “Ação03” é uma barra de sincronização denominado de Join que possui a finalidade de dividir o fluxo do processo em vários fluxos que podem ser executados de forma paralela.
- B) A barra de sincronização que está entre as ações “Ação02” e “Ação03” e a ação “Ação04” é denominada de Fork e representa a situação em que a “Ação04” só poderá ser executada após o término das ações “Ação02” e “Ação03”.
- C) As ações “Ação02” e “Ação03” são mutuamente exclusivas, ou seja, serão executadas de acordo o estado da barra de sincronização que está após a ação “Ação01”.
- D) A barra de sincronização do tipo Fork que está após as ações “Ação02” e “Ação03”, fará com que a ação “Ação04” seja executada imediatamente após o encerramento da “Ação02” ou da “Ação03”.
- E) A barra de sincronização que está entre a ação “Ação01” e as ações “Ação02” e “Ação03” é denominado de Fork que possui a finalidade de dividir o fluxo do processo em vários fluxos que podem ser executados de forma concorrente.

[Questão]

(UFRGS – ATI – Faurgs – 2018)

Considere as afirmações abaixo sobre diagramas de atividade da UML.

_____ indicam ações que são executadas no sistema.

_____ são representadas por setas contínuas e são usadas para indicar o fluxo de trabalho entre elementos do diagrama.

Uma ramificação lógica é indicada por _____ e representa desvios do fluxo de controle.

Um retângulo com bordas arredondadas identifica _____ e representa marcos de processamento.

Assinale a alternativa que preenche, correta e respectivamente, as lacunas das afirmações acima.

- A) Mensagens – Transições – ações – componentes
- B) Eventos – Dependências – mensagens – ações
- C) Atividades – Transições – pontos de decisão – ações
- D) Atividades – Dependências – mensagens – eventos
- E) Eventos – Dependências – pontos de decisão – componentes

UML

Diagrama de Visão Geral de Interação

Diagrama de visão geral de interação.

- Variação do diagrama de atividade.
- Fornece uma visão geral do controle de fluxo.
- Utiliza quadros no lugar dos nós de ação.
- **Dois tipos:**
 - **Quadros de interação**, que contêm qualquer tipo de diagrama de interação UML.
 - **Quadros de ocorrência de interação**, que fazem uma referência a um diagrama de interação, mas não apresentam seu detalhamento.

UML

Diagrama de Componentes

Diagrama de componentes.

- ☞ Identifica os componentes que fazem parte de um sistema, um subsistema ou mesmo os componentes ou classes internas de um componente individual.
- ☞ Pode ser utilizado como uma forma de documentar como estão estruturados os arquivos físicos de um sistema, permitindo assim uma melhor compreensão do mesmo.
- ☞ Facilita a reutilização de código.
- ☞ Pode identificar os componentes utilizados no desenvolvimento de sistemas baseados em componentes.

UML

Diagrama de Implantação

Diagrama de implantação.

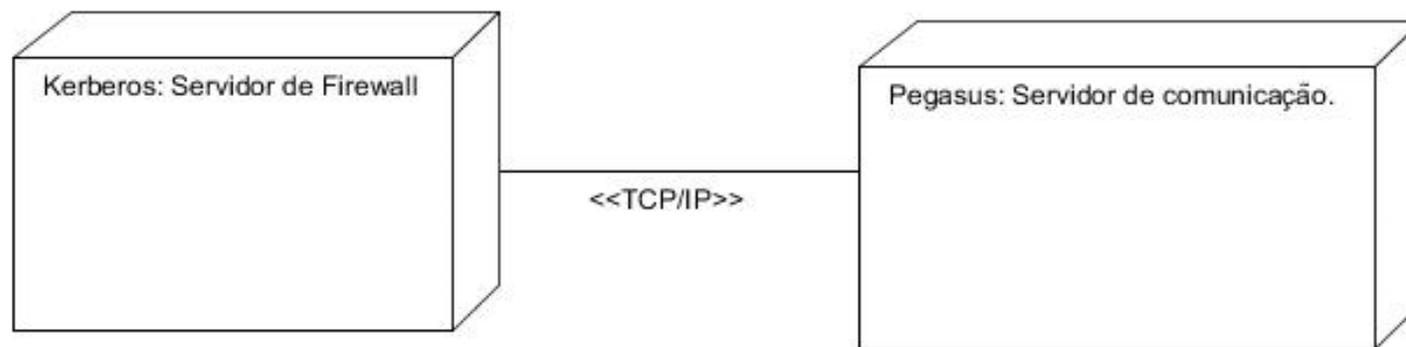
- ☞ Possui a visão mais física da UML.
- ☞ Enfoca a questão da organização da arquitetura física sobre a qual o software será implantado executado em termos de hardware e suas conexões.
- ☞ Útil quando o sistema modelado for executado sobre múltiplas máquinas.
- ☞ **Nós.**
 - ☞ Componentes básicos de um diagrama de implantação.
 - ☞ Pode representar um item de hardware.
 - ☞ Pode representar também um ambiente de execução que suporta o sistema de alguma forma.
 - ☞ Representado por um cubo.
 - ☞ Pode conter detalhes de configuração do hardware que ele representa por meio de etiquetas (tags).

Estereótipos.

-  **Devide:** aplicado a itens de hardware.
-  **Computer:** representa um computador simples.
-  **Secure:** representa um hardware de segurança responsável por impedir invasões à rede interna da organização.
-  **Server:** representa um servidor.
-  **Storage:** representa um hardware cuja função é armazenar informações, como SBDs ou arquivos.

Associações.

-  Representadas por linhas ligando um nó a outro.
-  Determinam uma ligação física por onde poderá haver comunicação, incluindo o protocolo de comunicação a ser utilizado.



Artefatos.

-  Entidade física, elemento concreto que existe realmente no mundo real, assim como os nós que o suportam.
-  Pode ser um arquivo fonte, um arquivo executável, um arquivo de ajuda, um documento de texto.
-  Deve estar implementado em um nó.

Especificação da implantação.

-  Especifica um conjunto de propriedades que determinam parâmetros de execução de um artefato implementado em um nó.

Nós contendo pacotes.

-  Um nó pode suportar subsistemas ou mesmo sistemas inteiros que são representados como pacotes.

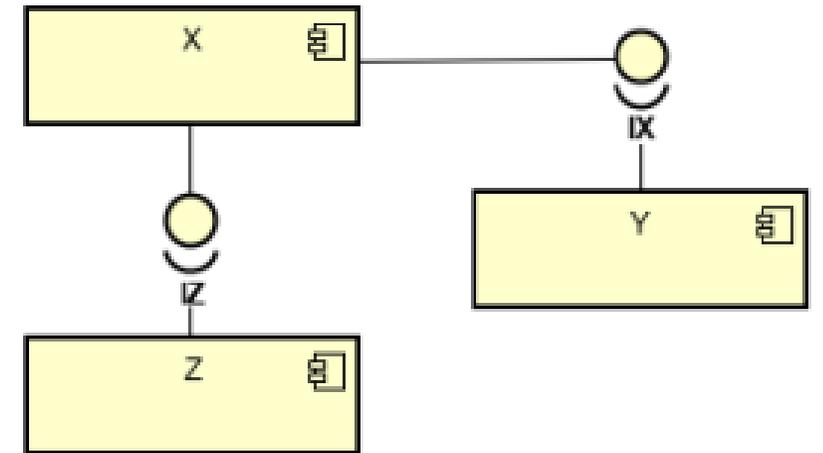
[Questão]

(UFRN – Técnico – TI – 2019 – Conperve)

A Linguagem de Modelagem Unificada (UML) é utilizada para a visualização, a especificação, a construção e a documentação de artefatos de software. Nesse contexto, observe a figura abaixo.

Essa figura representa um diagrama UML de

- A) classes.
- B) componentes.
- C) implantação.
- D) distribuição.



UML

Diagrama de Estrutura Composta

Diagrama de estrutura composta.

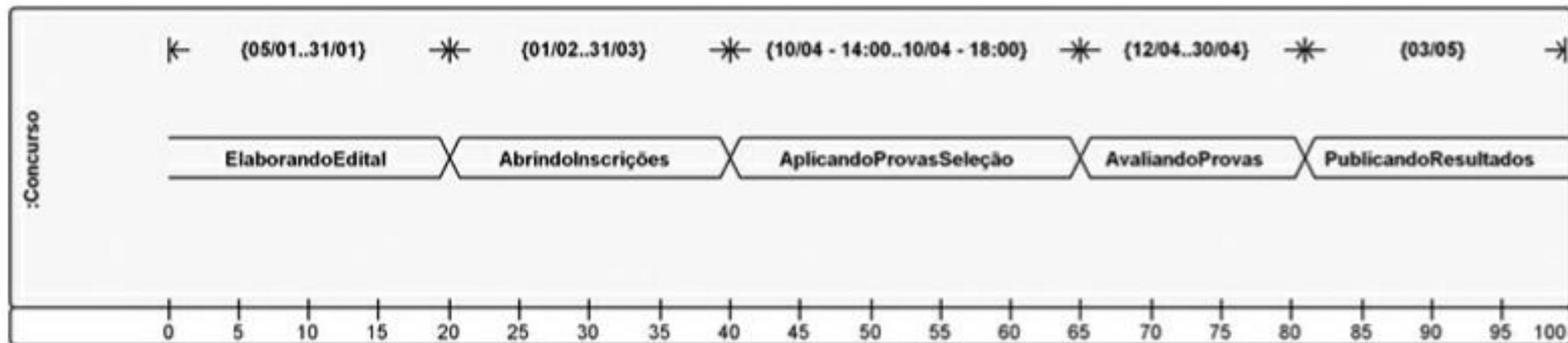
- É novidade da UML 2.0.
- Pode ser utilizado para modelar colaborações.
- Descreve uma visão de um conjunto de entidades cooperativas interpretadas por instâncias que cooperam entre si para executar uma função específica.
- Semelhante ao diagrama de classes.

UML

Diagrama de Temporização

Diagrama de Tempo ou de Temporização

- Semelhante ao diagrama de máquina de estado.
- Mas enfoca as mudanças de estado de um objeto ao longo do tempo.



Dúvidas

Professor Gabriel Pacheco

http://www.provasdeti.com.br/profile-teacher.php?professor_info_id=6

<https://www.instagram.com/professor.gabrielpacheco/>

<https://www.youtube.com/profgabrielpacheco>

<https://www.facebook.com/coachgabrielpacheco>

<http://www.professogabrielpacheco.com.br>



PROVAS DE TI
TUDO PARA VOCÊ PASSAR